CENTERS FOR MEDICARE & MEDICAID SERVICES

FFE Eligibility API Integration Manual

GENERAL AND TECHNICAL FREQUENTLY ASKED QUESTIONS

UPDATED: JULY 22, 2020

Contents

Figures

Introduction

This document is intended to expand on the other documentation for the Federally Facilitated Exchange (FFE) Application Programming Interfaces (APIs) by capturing Frequently Asked Questions (FAQ) and lessons learned on how external User Interfaces (UIs) should interact with the FFE's eligibility APIs.

This FAQ document is organized into the following sections:

1. **SES Documentation:** Summarizes the documentation available on how to integrate external UIs with the FFE's Application Services.
2. **API Services:** Describes the high-level purpose, inputs and responses of the FFE's eligibility APIs.
3. **Application Sections:** This section provides examples of how requestors can provide or use the data that is returned from the APIs within the following sections of the UI, which generally follows the flow of the data collected in the UI for backend processing.
    a. Household Contact Information
    b. Household Composition
    c. More about this household (Sex, SSN, Race/Ethnicity, Non-MAGI, Medicaid Block)
    d. Citizenship/Immigration
    e. Income
    f. Preliminary Eligibility
    g. Sign and Submit
    h. Eligibility Results
    i. Change in Circumstance
4. **General Question:** This section provides answers to general and technical questions for Enhanced Direct Enrollment (EDE) partners. Audit and compliance-related topics are not covered in this document; please refer to the "EDE Compliance FAQs" document on zONE for audit and compliance-related topics.
    a. **Access to Resources:** This section provides resources that are available to the CMS web portal and zONE.
    b. **API Questions:** This section provides additional information that is available on the FFE's APIs.
    c. **Testing Questions:** This section answers technical questions related to build/system issues in the testing and production environments.

Audience

The intended audience of this document includes both technical and functional EDE partners and other external UI requestors who are integrating with the FFE's Application Services.

SES Documentation

As documented in **APIary**, the FFE's Application Services are exposed as APIs to allow the retrieval and manipulation of application data through orchestration of the back-end eligibility logic.

In addition to APIary, the **UI Application Principles Document** contains the following attachments with supporting documentation on integration with the FFE's Application Services:

- **Application Service Companion Guide**: Provides validation logic and eligibility impacts of inputs to the application services.
- **UI Question Companion Guide**: Provides policy background and requirements on how to collect attestations in the application.
- **UI and API Interaction Flows:** Provides guidance on which application services to call in different consumer scenarios.
- **App 3.0 Section Flows**: Provides an example of UI navigation and recommended calls to the application services.

These documents include a variety of technical and policy requirements, such as:

- **Inputs:** Examples of how requestors should pass inputs to the application services.
- **Responses:** Explanation of how requestors can use the computed responses.
- **Validations:** Validation logic and recommended error handling responses.
- **Clearing:** Clarification on how requestors should clear attestations that are no longer applicable.
- **Reference Data:** System and state configuration data available through the Reference Data API.
- **API Interaction:** Examples of how the requestor should call and use the different APIs.

Provided in the table below is a high-level summary of where requestors can find documentation on the following areas of requirements:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI and API Interaction Flows	UI Application Principles Document
Inputs	X	X	X	X		X
Responses	X	N/A	X	X		X
Validations	N/A	X	N/A	N/A	N/A	X
Clearing	N/A	N/A	N/A	N/A	N/A	X
Reference Data	X	N/A	N/A	N/A	N/A	X
API Interaction	X	N/A	N/A	X	X	X

API Services

When determining the question flow within the UI, requestors should consider when the UI will need to call the different APIs and the order of the event dependencies within the backend eligibility logic.

Inputs:

1. Which APIs should the UI call to complete an application?
    a. Create Application
        i. **Purpose:** This API creates an initial application.
        ii. **Inputs:** To create an initial application, the requestor must provide the required inputs at the application and member level; else the requestor must provide the application ID to create an application from the prior coverage year.
        iii. **Response:** The response from Create App contains application metadata (such as application ID, version number, creation date, etc.) and basic member-level information (including member ID and name).

```
{
        "application": {
                "coverageYear": 2018,
                "coverageState": "DE",
                "contactInformation": {
                        "email": "t2123@yopmail.com",
                        "primaryPhoneNumber": {
                                "number": "2222222222",
                                "type": "WORK"
                        }
                }
        },
        "applicationMembers": [{
                        "householdContactIndicator": true,
                        "firstName": "First",
                        "lastName": "Last",
                        "birthDate": "1908-01-01",
                        "mailingAddress": {
                                "streetName1": "Street",
                                "cityName": "City",
                                "stateCode": "DE",
                                "zipCode": "19805",
                                "countyName": "County",
                                "countyFipsCode": "10003"
                        }
                }
        ]
}
```

Figure 1: Minimum inputs required to create an application using the Create App Service

    b. Create Application from Prior Year
        i. **Purpose**: This API creates a new application from the prior coverage year.
        ii. **Inputs**: To create an application from the prior year, the requestor must provide the application ID to create an application from the prior coverage year.

iii. **Response**: The response from the Create Application from Prior Year contains application metadata including the new insurance application identifier, version number, creation date, application type and the creating DE partner identifier (if created by a DE partner). Note: The coverage year for the newly created application is coverage year + 1 from the input application coverage year. For example, if the service is requested with an application with a coverage year of 2017, the service will create a new application with a coverage year of 2018.

c. Update Application

i. **Purpose:** This API allows for modifications of an existing application based on the inputs provided by the requestor. This service updates in-progress and submitted applications. Based on the inputs to the service, specific verification and eligibility logic is re-computed and returned to the requestor.

ii. **Inputs:** The SES Update Application API accepts attestations and attestation changes including member-level attestations, application-level attestations, and member relationships. Members cannot be added to or removed from the application through Update Application; there are separate APIs to add and remove members. However, once a member is added through the Add Members API and a member identifier is created, all updates for the member are accepted through the Update Application API including establishing relationships between other members on the application.

iii. **Response:** The response from Update App contains application metadata and the computed member and household-level results (including household composition, income, verification, and preliminary eligibility) that can inform certain UI interactions.

d. Add Members

i. **Purpose:** This API allows a requestor to add members from an application.

ii. **Inputs:** The Add Members API only accepts member-level attestations for added member(s) in the request. The requestor must call the Update Application Service to update existing member attestations and define the relationships of the added member(s) to existing members.

iii. **Response:** The response from Add Members contains application metadata and the computed member-level results (including verification). This service also returns the member attestations.

e. Remove Members

i. **Purpose:** This API allows a requestor to remove members from an application.

ii. **Inputs:** This API accepts one or more member identifiers, removal reason and removal date per member in the request and removes the member(s) from the application by deleting all of the members' data (including the members' attestations, relationships, eligibility results) and stores the member's removal details (PTN, member identifier, name, birth date, removal reason and removal date) to the removedMembers segment of the application. The service re-computes eligibility for remaining members. The Remove Members API validates that a member who is currently the application contact cannot be removed. In order to remove the application contact, SES Update Application Service must be called first to assign a new application contact.

iii. **Response:** The response from Remove Members contains application metadata only (including application ID, version number, and last modified date).

f. Submit Application

i. **Purpose:** This API collects the application signature, finalizes the submission date of the application and finalizes the eligibility results.

ii. **Inputs:** The requestor should enter all attestations via the Update Application API prior to calling the Submit Application API. The Submit Application API computes and/or re-computes verifications/eligibilities to ensure that all required fields are present and valid to submit the application. Upon successful submission, the service generates a submitted application that is immutable and can be used to process enrollment and downstream processes.

iii. **Response:** The response from Submit App contains application metadata and the computed member, household, and application-level results (including final eligibility).

2. When should the UI call the APIs to update the application?

   a. Requestors must use the APIs to enable dynamic interaction between the UI and backend eligibility logic, so that subsequent questions are only displayed when needed based on the prior attestations and verification/eligibility results of the application members. Therefore, the UI will need to call the APIs to complete the following high-level steps within the application. Note that to return the necessary responses from the APIs, the order of the questions in the UI must align with the order of the flow dependencies in the backend eligibility logic.

      i. After collecting the *household contact information*, call the Create App API to **create the application**.

         1. For *existing applications*, call the Get App API before Create App to **return the latest application data**.

      ii. After collecting *relationships* between the members for applications requesting financial assistance (FA), call the *Update App API* to **determine household composition**.

         1. After collecting the *name and birth date* of additional members (applicants and related non-applicants), call the Add Members API to **add members** to the application.

      iii. After collecting the *SSN and citizenship or lawful presence attestations*, call the Update App API to **verify the SSN and citizenship or immigration status** of each applicant

      iv. After collecting the *income attestations* for FA applications (current income attestations for each member and annual income attestations for all members), call the Update App API to **verify the attested income** for the household and **provide initial preliminary eligibility** with responses that specify which program-specific questions to ask.

      v. After collecting the *program-specific questions for Medicaid, CHIP, APTC, and QHP*, call the Update App API to **determine the final preliminary eligibility** for each applicant.

      vi. After collecting the *legal agreements for Medicaid/CHIP or APTC/QHP*, call the Update App API and then Submit App API to **determine and display the final eligibility** on the eligibility results page.

      vii. After determining the *final eligibility* for all applicants, call the DSRS API to **display the EDN** on the eligibility results page.

         1. After collecting a *full determination request* from a QHP-eligible applicant, call the Submit App API to **send the full determination request** to the state Medicaid agency.

3. How can the requestor clear prior inputs from the application?

a. The Update Application API supports partial updates. To distinguish between a new value for a field and the removal of a value for a field, the requestor must provide a field value of "null" to indicate the removal of the value, while any other input will be interpreted as a new value. Note the FFE's Application Services do not allow empty string ("") as valid value for most fields and will throw a validation error in response to this input.

4. When should the requestor clear prior inputs from the application?
    a. Conflicting attestations may negatively impact the eligibility results in the following scenarios:
        i. The consumer may backwards navigate to change a prior answer within the application. The new attestation may invalidate or alter downstream responses that, as a result, may not re-prompt and display subsequent questions within the UI that were previously stored in the API.
        ii. During CiC or Pre-pop applications, the consumer may change an answer that sends them through a different pathway in the UI which does not re-display questions for attestations that were previously returned in the Get App API.
        iii. During CiC or Pre-pop applications, data from the Get App API may return validation errors that will block submission if not resolved while the consumer updates the application through the UI.

In these scenarios, the requestors must clear the answers to any questions that are not on the final pathway by entering the value of 'null' for those attestations into the API.

For example, in the following diagram, member 1 attests to being a tax filer and member 2 attests to being a tax dependent whose tax filer is not on the application.

```
PUT /applications/123
{
  "application": {
    "accountHolderIdentityProofedIndicator": true,
    "requestingFinancialAssistanceIndicator": true
  },
  "members": {
    "1": {
      "requestingCoverageIndicator": true,
      "demographic": {
        "maritalStatus": "UNMARRIED"
      },
      "family": {
        "taxDependentIndicator": false,
        "taxFilerIndicator": true,
        "taxReturnFilingStatusType": "HEAD_OF_HOUSEHOLD",
        "claimsDependentIndicator": true
      }
    },
    "2": {
      "requestingCoverageIndicator": true,
      "family": {
        "taxDependentIndicator": true,
        "taxFilerIndicator": false,
        "taxFilerNotProvidedIndicator": true,
        "claimingTaxFilerNotOnApplicationIndicator": true
      }
    }
  },
  "household": {
    "familyRelationships": [],
    "taxRelationships": []
  }
}
```

Figure 2 Clearing Attestations Request 1 Diagram

As a result, the API computes the tax household for member 1 and is unable to compute the tax household for member 2 – which is the correct outcome with this combination of inputs.

```
200 OK
{
  "taxHouseholds": {
    "1182335275184698393": {
      "taxHouseHoldComposition": {
        "taxHouseholdStatus": "NO",
        "taxHouseholdStatusReason": "662_CLAIMING_TAX_FILER_NOT_ON_APPLICATION"
      }
    },
    "1182335275184698392": {
      "taxHouseHoldComposition": {
        "taxHouseholdStatus": "YES",
        "taxHouseholdStatusReason": "999_N_A",
        "taxHouseholdMemberIdentifiers": [
          "1182335275184698392"
        ]
      }
    }
  }
}
```

Figure 3 Clearing Attestations Result 1 Diagram

Then, while completing a CIC or after backwards navigating within the UI, the consumer changes their attestations to say member 1 is the parent and tax filer of member 2 – but the requestor *fails to clear the "taxFilerNotProvidedIndicator"* in the subsequent call to the Update App API.

```
PUT /applications/123
{
  "application": {
    "accountHolderIdentityProofedIndicator": true
  },
  "household": {
    "familyRelationships": [
      ["1", "PARENT", "2", { "resideTogetherIndicator": true }]
    ],
    "taxRelationships": [
      ["1", "TAX_FILER", "2"]
    ]
  }
}
```

Figure 4 Clearing Attestations CiC Request 2 Diagram

As a result, the back-end is unable to compute the tax household for member 2, even though the tax household for member 1 now includes both members – which produces an incorrect eligibility result.

```
200 OK
{
 ...,
 "result": {
  "computed": {
   ...,
   "taxHouseholds": {
    "1182335275184698393": {
     "taxHouseHoldComposition": {
      "taxHouseholdStatus": "NO",
      "taxHouseholdStatusReason": "662_CLAIMING_TAX_FILER_NOT_ON_APPLICATION"
     }
    },
    "1182335275184698392": {
     "taxHouseHoldComposition": {
      "taxHouseholdStatus": "YES",
      "taxHouseholdStatusReason": "999_N_A",
      "taxHouseholdMemberIdentifiers": [
       "1182335275184698392",
       "1182335275184698393"
      ]
     }
    }
   }
  }
 }
}
```

Figure 5: Clearing Attestations CiC Result 2 Diagram

5. Can a requestor prepopulate an application using data from 2 years prior?
    a. Yes, the requestor will need to retrieve the desired application using the Get Application API and create a new application using the attestations from the prior application and follow the steps below:
        i. Call the Get Application API with the desired application to prepopulate from and cache/store the application data.
        ii. Call the Create Application API (not Create from Prior API) and include all members in the request
        iii. Call the Get Application API to retrieve the member identifiers for each application member
        iv. Call Update Application API with the new year application identifier and include all the member attestations
        v. Send a request to the Update Application API with the attestation data
6. When can an EDE partner create a completely new application when Creating from Prior year?
    a. EDE Partners are permitted to create a new application when:
        i. The previous year application is not the correct coverage state, and there is no current year application.
        ii. The current year application is not the correct coverage state.
        iii. There is no current year application, and the previous year application is version 1, and is in-progress (note that the coverageYearNumber, latestApplicationVersion,

latestApplicationVersionConsumerIsOn, and applicationProcessStatus will be returned in the Person Search Response).

1. Note, if a prior coverage year application is returned with the correct coverage state, and it is version 2 or greater, then EDE Partners (or agents/brokers) are still required to prepopulate an application from the prior year application using the Create App from Prior Year App API, even if the most recent version is in "in-progress" and was not submitted.
    a. When sending the request to the Create App from Prior Year App API in this scenario, EDE Partners will need to either: 1) send the request without an application version number, or 2) send the request with the application version that is in a "complete" or "submitted" state.
2. Note also, there is additional information in the EDE API Companion Guide that addresses what to do when Person Search returns more than one application for a given coverage year.

7. Which other application members/fields need to be modified when removing a member?
    a. If the attestations are not updated during the normal course of UI questioning, the following indicators should be updated to prevent validation errors.
        i. claimsDependentIndicator: If all dependents are removed from the application, the tax filer's claimsDependentIndicator attestation should be set to 'false' prior to submitting the application. If the claimsDependentIndicator is not updated, the requestor will receive a validation error upon submission.
    b. If a consumer chooses to remove an applicant from the application, the requestor should determine whether there are any non-applicants who should be removed as well because they are no longer relevant to any applicant's eligibility. The requestor can call the Remove Member service to remove applicants as well as to remove non-applicants. If the requestor does not remove non-relevant non-applicnats (defined below) it will not cause a validation error and will not result in incorrect eligibility findings. However, it is helpful to remove irrelevant information so that a subsequent version of the application is populated only with relevant information for eligibility.
        i. For example, if an applicant tax filer is removed who claims non-applicant dependent(s) that are not claimed by any other member, then the non-applicant dependent(s) should be removed as well if they are no longer relevant to any applicant's eligibility, as defined in paragraph (ii) below.
        ii. A non-relevant non-applicant application member is defined as an application member where all of the following are true:
            1. Not seeking coverage AND
            2. Does not have a taxRelationship to any other applicant AND
            3. Does not live with any other applicant AND
            4. Is not married to any other applicant AND
            5. Is not the application filer
8. How often must a requestor call the State and System Reference Data APIs to refresh "Time to Live" data?
    a. It is recommended that requestors call the Reference Data API at most daily at 12pm EST. In general, the Reference Data API is not updated more often than once per month, so less frequent calls would also be acceptable.

Responses:

1. Which APIs can the UI call to retrieve the current application and reference data?
    a. Get Application
        i. This API provides a detailed and summary view of a consumer's application.

   ii. The summary view may be requested via Application ID or Person Tracking Number (PTN) and the detailed view may be requested via Application ID or Application ID + Version Number (no method of requesting via PTN).

b. Reference Data

   i. This API provides state and system configuration data for the requestor to display content within the UI.

   ii. UI systems are expected to load the eligibility reference data during application startup. Requestors will need to cache and refresh the reference data quarterly.

c. Address Validation

   i. The requestor should run address validations against a third-party address validation system.

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on API Services:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	Request	Input Matrix: Create, Update, Add, & Submit App	Phase 1 & 2 Questions, UI Questions, Eligibility Results, Requirements	Get Started, Household, Income, Additional Questions, Review & Sign	3.1 – Eligibility Event Dependencies 3.2 – Application Services 3.3.4 – Option for Consumer Not to Provide Information 3.3.6 – Setting Tax Return Filing Status Type
Responses	Response		Backend Response for UI	Eligibility Results	2.- Application Policy Principles 3.3.1 – Branch for Financial Assistance 3.3.2 – Dynamic Questions 3.3.3 – Backend Interaction
Validations		Input Matrix: Service & Flow Validations			2.5 – Application Operational Assistance 5.2.4 – Exception Handling
Clearing					3.3.5 – Backward Navigation

Reference Data	State and System				3.4 – Application State and System Configuration
API Interaction	Service Guiding Principles			Financial Assistance Flow	4. – Sample UI Interactions with Application Services 5. Technology Principles

Application Sections

Household Contact Information

Each API includes service-level validations to ensure the UI collects the minimum required attestations before calling the backend logic. For example, the minimum required fields to create an initial application include first name, last name, date of birth, primary phone number, and mailing address of the household contact. If the service-level validations fail, the API returns the following high-level responses indicating the request was unsuccessful The API also returns an errorCode and JSON path for any inputs that return a validation error.

- httpStatusCode = 400
- resultType = error
- errorType = client_error
- errorCode = invalid_request
- apiMessage = Request is malformed, please correct the data and try again

```
{
        "application": {
                "coverageYear": 2018,
                "coverageState": "DE",
                "contactInformation": {
                        "email": "t2123@yopmail.com",
                        "primaryPhoneNumber": {
                                "number": "2222222222",
                                "type": "WORK"
                        }
                }
        },
        "applicationMembers": [{
                        "householdContactIndicator": true,
                        "firstName": "First",
                        "lastName": "Last",
                        "birthDate": "1908-01-01",
                        "mailingAddress": {
                                "streetName1": "Street",
                                "stateCode": "DE",
                                "zipCode": "19805",
                                "countyName": "County",
                                "countyFipsCode": "10003"
                        }
                }
        ]
}
```

Figure 6 Unsuccessful Request for Create Application

```
{
   "resultType": "ERROR",
   "error": {
      "errorType": "client_error",
      "apiMessage": "Request is malformed, please correct the data and try again",
      "errorCode": "invalid_request",
      "errors": [
         {
            "errorCode": "required",
            "path": "$.applicationMembers[0].mailingAddress.cityName"
         }
      ]
   }
}
```

Figure 7 Unsuccessful Create Application HTTP Response

In addition to service-level validations, the backend logic contains flow-level validations to ensure the UI collects the minimum required attestations for determining eligibility. Since the UI should only ask questions that are necessary for eligibility, the flow-level validations are dependent on the inputs and verification/eligibility results of the application members. The API returns the following high-level responses indicating the request was successful with validation errors that will prevent submission. The API also returns an errorCode and JSON path for any inputs that return a validation error.

- httpStatusCode = 200 for all APIs *except* Submit Application, else httpStatusCode = 409 for Submit Application
- resultType = success_with_errors
- errorType = client_error
- errorCode = validation_errors
- apiMessage = Data Validation issue, please correct the data and try again

```
{
    "application" : {
        "requestingFinancialAssistanceIndicator" : true
    },
    "members" : {
        "123456789" : {
            "requestingCoverageIndicator" : true,
            "demographic" : {
                "birthDate" : "1995-01-01",
                "sex" : "FEMALE",
            },
            "family" : {
                "claimsDependentIndicator" : false,
                "parentCaretakerIndicator" : false,
                "absentParentIndicator" : false,
                "taxDependentIndicator" : false,
                "taxFilerIndicator" : true,
                "taxReturnFilingStatusType" : "SINGLE_FILER"
            }
        }
    }
}
```

Figure 8 Sample Request

```
{
  "resultType": "SUCCESS_WITH_ERROR",
  "error": {
    "errorType": "CLIENT_ERROR",
    "apiMessage": "Data Validation issue, please correct the data and try again",
    "errorCode": "validation_errors",
    "errors": [
      {
        "errorCode": "required",
        "path": "attestations.members[\"1252483273063875433\"].family.pregnancyIndicator"
      }
    ]
  }
}
```

Figure 9 Success with errors HTTP Response when pregnancy indicator was not provided

Inputs:

1. What is the minimum data required to create an initial application? If the inputs below are not provided, the Create App API returns httpStatusCode = 400.
    a. The following inputs are required at the *application-level*:
        i. coverageYear
        ii. coverageState
        iii. primaryPhoneNumber
            1. number
            2. type
    b. The following inputs are required at the *member-level*. Note the requestor must provide the **household contact** while creating the application and has the *option* of adding additional members.
        i. firstName
        ii. lastName
        iii. birthDate
        iv. householdContactIndicator = true for exactly one member
        v. mailingAddress where householdContactIndicator = true
            1. streetName1
            2. cityName
            3. stateCode
            4. zipCode
            5. countyFipsCode

2. How do the screener questions map to the API inputs?
    a. The screening questions are asked prior to the application to screen out consumer circumstances that are unsupported by the application. Please refer to the Screening Question Mapping SES tab in the UI Question Companion Guide for mapping guidance.

3. How can EDE partners process attestations from a Phase 3 application that are not supported by the Phase 1 or 2 Screeners?
    a. EDE partners must clear or default attestations that are not supported by Phase 1 or Phase 2 screeners. The clearing logic can be found in the "EDE Phase Clearing" column of the Application Services Companion Guide which, defines the clearing rules for each input data element. The following are the high-level rules for the EDE Phase clearing logic:
        i. Keep, do not clear - Data element must be retained
        ii. Clear - Data element must be set with a 'null' value
        iii. Default - Data element must be set with the defaulted value as defined in the Application Services Companion Guide
        iv. Conditional logic – Data elements that require conditional clearing.

4. Should the EDE Phase 1 and 2 screeners allow a transient address?
    a. If the partner models their screener after the App 2.0 screener, then we do not need to include a “no home address” option for a Phase 1 or 2 launch, as that’s covered in the screener question “Does everyone live together at the same home address.” Phase 1 and 2 applications should include a UI validation that does not allow the consumer to enter a home address outside of the application state.

5. Are there differences between the Phase 1 and Phase 2 questions for applications that are requesting financial assistance (FA) vs. non-FA?
    a. Applications that are requesting Financial Assistance require significantly more information than non-FA applications. For FA applications, it’s important to note the income estimate is based on the household size and not the number of people applying for coverage. Non-FA applications are good examples of more simple applications that can be used to begin integration testing with the FFE’s APIs.

6. Which inputs are applicable to non-FA applications?
    a. The following verification and eligibility results are applicable to non-FA applications[1]:
        i. Household Composition
            1. Marital status and tax relationship are not applicable to non-FA applications, however legal relationships may be required depending the eligibility results and family relationships between the applicants
        ii. SSN
        iii. Citizenship
        iv. Immigration
        v. Residency

[1] The names of the eligibility and verification results are based on the Event Dependencies within the UI Application Principles Document and Application Services Companion Guide

vi. Incarceration
vii. SEP
viii. QHP

7. What is the minimum data required to update an application? If the inputs below are not provided, the Update App API returns httpStatusCode = 400. If the Update App API returns an errorCode with httpStatusCode = 200, the requestor must add/modify the inputs returned in the path before the application can be submitted.
   a. The following inputs are required at the *application-level*:
      i. accountHolderIdentityProofedIndicator
   b. There are no inputs required at the *member-level.*

8. What data is always required to submit an application?
   a. If the inputs below are not provided, the Submit App API returns httpStatusCode= 400.
      i. The following inputs are required at the *application-level*:
         1. accountHolderIdentityProofedIndicator
         2. applicationSignatureName
         3. applicationSignatureDate
         4. applicationSignatureType
   b. If the inputs below are not provided, the Submit App API returns httpStatusCode = 409.
      i. The following inputs are required at the *application-level*:
         1. nonIncarcerationAgreementIndicator
         2. applicationAssistorType
         3. requestingFinancialAssistanceIndicator
      ii. The following inputs are required at the *member-level for applicants*:
         1. homeAddress or noHomeAddressIndicator
9. How should a requestor communicate which household member is filling out the application?
   a. Set the householdContactIndicator = true
   b. On an Update App API request, set a "SELF" relationship attestation for the member ID that is set as the contactMemberIdentifier.
   c. Note that in upcoming R12 for the API service, setting this "SELF" attestation will no longer be necessary

```
"household": {
        "familyRelationships": [
                ["1234","SELF","1234"]
},      ]
"application":{
        contactMemberIdentifier: 1234
}
```

Figure 10 Attesting Household Contact as Subscriber

Responses:

The error responses from the APIs are intended for defect triage only. If the APIs return an error, the UI should display a generic message to the user. The UI should not use the error responses from the APIs for site navigation or dynamic clearing logic.

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on the household contact:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	attestations	Input Matrix: Create App	Phase 1 & 2 Questions, UI Questions, Requirements	Get Started	3.2 - Application Services
Responses	computed	N/A	Backend Responses for UI	N/A	N/A
Validations		Input Matrix: Service & Flow Validations	N/A	N/A	2.5 – Application Operational Assistance 5.2.4 – Exception Handling
Clearing	N/A	Input Matrix: EDE Phase Clearing	N/A	N/A	N/A
Reference Data	N/A	N/A	N/A	N/A	N/A
API Interaction	Service Guiding Principles	N/A		Get Started	4 - Sample UI Interactions with Application Service

Household Composition

For those requesting financial assistance (FA), we need to calculate their tax and Medicaid family/household size based on a combination of their tax/familial relationships and the relevant members of their family who live together. We'll use each applicant's household size, along with other information we collect in the application to determine their eligibility for different programs (Medicaid, CHIP, APTC, CSR). Tax households include the tax filer(s) and their tax dependents without evaluating familial relationships or whether members of the tax household live together. Medicaid households are the same as the tax household unless exceptions apply. When exceptions apply, familial relationships and whether family members reside together determine the Medicaid household. Different household income amounts for different household sizes will qualify an applicant for a different program, as the federal poverty line data steps up with the size of a household. For example, if you earn $50,000/year and you have a tax household of 1 (single filer, no spouse, no dependents), your income would be too high for APTC eligibility. But if you're married (tax household of 2 b/c you are filing jointly) and your household income is $50,000/year, you may be eligible for APTC.

Inputs:

1. Which tax relationships need to be collected?
    a. If the applicant is filing taxes, we need to collect tax relationships for the people who appear on their tax return, including their spouse if applicable and any dependents.
        i. For APTC, a tax filer's household includes:
            1. The tax filer
            2. Their spouse IF filing jointly
            3. Anyone claimed as a dependent
        ii. For Medicaid/CHIP, a tax filer's household includes:
            1. The tax filer
            2. His/her spouse If filing jointly or filing separately but they live together
            3. Individuals the tax filer claims as tax dependents

2. Which family and legal relationships need to be collected?
    a. For Medicaid/CHIP, familyRelationships between household members are used to determine whether an exception to the tax household applies for Medicaid and CHIP eligibility. For example, if an applicant is a tax dependent, then they will have an exception to the tax household rules if the claiming tax filer is not the dependent's parent.
        i. Examples of Family Relationships
            1. Parent
            2. Spouse
            3. Son/Daughter
            4. Stepson/Stepdaughter
            5. Grandchild
            6. Sibling
            7. Domestic Partner

    b. For QHP, familyRelationships are required between all QHP-eligible members (FA and non-FA) to determine enrollment groups during plan selection. When the family relationship between QHP-eligible applicants is not sufficient to allow enrollment in a plan together under all issuer business rules, then the UI also needs to ask whether the applicants have a "legal relationship." You could collect legal relationships directly after collecting family relationships, or later, post prelim, after you know who will be

QHP eligible. Legal relationships are not required for household composition but should be collected before the end of the application. Relationships are not required for household composition but should be collected before the end of the application.

Here are the rules for collecting legal relationships between QHP-eligible applicants:

- If applicant is the grandparent, grandchild, uncle/aunt, niece/nephew, first cousin, brother/sister, domestic partner, parent's domestic partner, or child's domestic partner of someone else – display group 1
- Else if applicant is the other relative of someone else OR is the son/daughter or stepson/stepdaughter of someone else and is at least 25 years old* OR is the parent or stepparent of someone else who is at least 25 years old* – display group 2
  - *age calculated as of current date
- Else don't display any legal relationships

Group 1:

- Collateral dependent
- Court appointed guardian
- Guardian
- Sponsored dependent
- Ward
- None of the above (if selected, then return a null response for legal relationship to the API)

Group 2:

- Collateral dependent
- Court appointed guardian
- Former spouse
- Foster child
- Guardian
- Sponsored dependent
- Ward
- None of the above (if selected, then return a null response for legal relationship to the API)

3. How should the requestor provide family vs. tax relationships?
   a. When a familial relationship is attested to in the application then the family relationship should be provided between the two application members. There is no need to provide a family relationship for the inverse or mirror relationship. In the example below it is read as “1234 is the Son/Daughter of 5678”. The backend logic of SES will also recognize this attestation as “5678 is the Parent of 1234”.
      i. Please note that for hierarchical relationships, the Get Application will return the relationship as it relates to the member with the higher order of the relationship. For example, if a requestor attests to a son/daughter relationship in Update App, the Get App response will return the relationship as it relates to the ‘Parent’. The same approach is true for grandchild, stepson/stepdaughter and daughter-in-law/son-in-law relationships which will return grandparent, step-parent and mother-in-law/father-in-law in the Get App response respectively.

```
{
        "household": {
                "familyRelationships": [
                        ["1234", "SON_DAUGHTER", "5678"]
                ]
        }
}
```

Figure 11 Setting family relationships

b. Refer to 7b to see how to set taxRelationships

4. What is the difference between the tax filer and a household contact and how they should be set?
    a. The household contact is also known as the application filer. The contactIndicator is set for this person during Create App. They could be identified as a tax filer later on in the application but the tax filer and household contact are not synonymous for a Phase 3 EDE application. However, on a Phase 1 or Phase 2 EDE application, a tax filer will indeed be the application filer/household contact.

5. How should the requestor indicate when spouses are married filing jointly vs. separately?
    a. The requestor should to provide familyRelationship = SPOUSE and separate taxRelationship = TAX_FILER for the two members and set the taxReturnFilingStatusType = MARRIED_FILING_JOINTLY or MARRIED_FILING_SEPARATELY *for both spouses*.

```
"household": {
        "familyRelationships": [
                ["1234","SPOUSE","5678"]
        ]
        "taxRelationships": [
                ["1234","TAX_FILER","1234"],
                ["5678","TAX_FILER","5678"]
        ]
},
“members”: {
        “1234”: {
                “family”: {
                        taxReturnFilingStatus = MARRIED_FILING_JOINTLY
                }
        },
        “5678”: {
                “family”: {
                        taxReturnFilingStatus = MARRIED_FILING_JOINTLY
                }
        }
}
```

Figure 12 Two Applicants attesting to Married Filing Jointly

```
"household": {
        "familyRelationships": [
                ["1234","SPOUSE","5678"]
        ]
        "taxRelationships": [
                ["1234","TAX_FILER","1234"],
                ["5678","TAX_FILER","5678"]
        ]
},
"members": {
        "1234": {
                "family": {
                        taxReturnFilingStatus = MARRIED_FILING_SEPARATELY
                }
        },
        "5678": {
                "family": {
                        taxReturnFilingStatus = MARRIED_FILING_SEPARATELY
                }
        }
}
```

Figure 13 Two Applicants attesting Married filing separately

6. How does married filing jointly vs. separately affect APTC eligibility?
    a. If married, the filing status that the tax filers plan to use on their federal income tax return for the coverage year impacts eligibility for APTC. Married tax filers may be eligible for APTC if they file jointly or if they use a head of household filing status. However, only a select few married tax filers are eligible to use the head of household status, and consumers may not be familiar with the tax rules. The UI should only allow a married consumer to attest to a head of household status if he or she claims at least one dependent and does not live with their spouse. Other married tax filers who plan to file separately from their spouse will be found ineligible for APTC (as will their dependents) but may be found eligible for Medicaid or CHIP.
    b. The application may provide an educational warning message to consumers about regarding the impact on eligibility when requesting information about filing status. Note that the application should not collect tax filing status from unmarried tax filers.

7. When should a requestor set the attestedHeadOfHouseholdIndicator?
    a. Requestors should not set the attestedHeadOfHouseholdIndicator = true for a household member who is single or for two spouses who are married filing jointly. This will prevent successful submission of the application and an error will be thrown on the Submit App Response. Phase 3 requestors must however set the attestedHeadOfHouseholdIndicator to either true or false when 1) the taxReturnFilingStatusType = MARRIED_FILING_SEPARATELY; 2) the tax filer is not living with their spouse; and 3) the tax filer attests

to claiming a tax dependent. If the indicator is null or missing from an Update App request in this scenario, it will not be possible to determine APTC eligibility and thus a validation error will be triggered.

8. How should the requestor provide the taxDependentIndicator vs. claimsDependentIndicator? If married filing jointly, should the requestor provide the taxFilerIndicator and claimsDependentIndicator for both spouses?
    a. The taxDependentIndicator is required for all applicant members on FA applications as well as non-applicants that are in the tax households of applicants. When taxfilerIndicator = true and claimsDependentIndicator is provided and the tax dependent question is not asked, the UI should default taxDependentIndicator = false. Otherwise the taxDependentIndicator should be set based on attestation.
    b. The taxDependentindicator should be set to true for the application members who will be claimed as dependent for the coverage year. If this application member's tax filer has been identified then the taxRelationship should be established as well.
    c. The claimsDependentIndicator should be set to true or false for attested tax filers who are not claimed as a dependent. It should be set to true when the tax filer attest to claiming a dependent for the coverage year.
        i. This indicator should also be set to the same value for the spouse of the tax filer if on the application and if attesting to Married Filing Jointly.
    d. In the JSON relationships are "read" left to right: in the example below, memberID 1234 is the tax filer of memberId 5678. The numbers represent the member IDs that are assigned by SES for the members.

```
"household": {
        "taxRelationships": [
                ["1234","TAX_FILER","5678"]
        ]
},
“members”: {
        “1234”: {
                “family”: {
                        taxFilerIndicator = true,
                        taxDependentIndicator = false,
                        claimsDependentIndicator = true
                }
        }
        “5678”: {
                “family”: {
                        taxDependentIndicator = false
                }
        }
}
```

Figure 14 Tax filer claiming a dependent

9. How does residency affect household composition?

a. State residency is a requirement for QHP eligibility (FA or non-FA) as well as Medicaid and CHIP eligibility. Where consumers live can also impact the household composition rules for Medicaid and CHIP. For example, if you live with your spouse, they will be included in your household composition for Medicaid and CHIP eligibility purposes regardless of tax filing status. When an applicant does not file taxes or meets an exception to the tax household rules, the household will be calculated based on who lives with them.

   i. The requestor should set the residesTogetherIndicator = TRUE when establishing relationships between all household members who live together.

```
"household": {
   "familyRelationships": [
      [ "1234", "SON_DAUGHTER", "5678", { "resideTogetherIndicator": true, "caretakerRelativeIndicator": true }]
   ]
}
```

Figure 15 Applicant attesting to being a Parent Caretaker Relative

10. What’s the difference between custodial vs. non-custodial parent?

    a. A parent is only considered a non-custodial parent if he or she doesn’t live with the child AND the child does live with another parent. This matters on Phase 3 applications for determining whether or not an exception to the tax household applies for a child applicant, which influences UI logic on whether to ask “non-filer” questions, meaning questions about the family members who live with the child but may not be on the tax return. For example, if the application filer claims their son/daughter on their tax return, but does not live with that child, then to determine whether the application filer is a non-custodial parent the UI would need to ask whether the child lives with any parent or step-parent. If not, the application filer is not a non-custodial parent and no further questions are needed regarding who the child lives with.

    b. If the son/daughter is above Medicaid child age, then the parent is never considered a non-custodial parent, regardless of whether living with their adult child.

    c. If parents split custody of the child, then the parent with whom the child spends the most nights is the custodial parent, and the other parent is the non-custodial parent.

    d. If the child is claimed by two parents/step-parents who are married filing jointly, but the spouses do not live together and only one spouse lives with the child, it is not considered a non-custodial parent scenario and there is no exception to the tax household.

```
{
        "members" : {
                "1253483612323209260" : {
                        "demographic" : {
                                "homeAddress" : {
                                        "streetName1" : "456 Main Street"
                                }
                        }
                },
                "1253483612323209261" : {
                        "demographic" : {
                                "homeAddress" : {
                                        "streetName1" : "123 ABC Street",
                                }
                        }
                }
        },
        "household" : {
                "familyRelationships" : [
                        ["1253483612323209260","SELF","1253483612323209260"],
                        ["1253483612323209260","PARENT","1253483612323209261", {"resideTogetherIndicator" : false}]
                ],
                "taxRelationships" : [
                        ["1253483612323209260","TAX_FILER","1253483612323209260"],
                        ["1253483612323209260","TAX_FILER","1253483612323209261"]
                ]
        }
}
```

Figure 16 Child with Non-Custodial Parent Request

```
{
    "members" : {
        "1253483612323209260" : {
            "demographic" : {
                "homeAddress" : {
                    "streetName1" : "456 Main Street"
                }
            }
        },
        "1253483612323209261" : {
            "demographic" : {
                "homeAddress" : {
                    "streetName1" : "456 Main Street ",
                }
            }
        }
    },
    "household" : {
        "familyRelationships" : [
            ["1253483612323209260","SELF","1253483612323209260"],
            ["1253483612323209260","PARENT","1253483612323209261", {"resideTogetherIndicator" : true}]
        ],
        "taxRelationships" : [
            ["1253483612323209260","TAX_FILER","1253483612323209260"],
            ["1253483612323209260","TAX_FILER","1253483612323209261"]
        ]
    }
}
```

Figure 17 Child with Custodial Parent Request

11. What are the exceptions which will cause the household size/composition for Medicaid and CHIP to be different from the household size for APTC eligibility (note the household size for APTC eligibility is always based on the tax household).
    a. An applicant who is claimed as a **tax dependent** may meet one of *three* exceptions:
        i. If they are a tax dependent of someone other than a parent/parent’s spouse (biological/step/adopted).
        ii. If they are below the Medicaid child age (this means under 19 or a full-time student aged 20-21 in select states per a “Y” for the “under21AndFTSOption” indicator in the Reference Data API) living with both parents but their parents do not file jointly.

iii. If they are an applicantunder the Medicaid child age who expects to be claimed as a tax dependent by a non-custodial parent. Note that a tax filer is only considered a non-custodial parent if there is another parent with whom the child lives. If parents split custody of the child, then the parent with whom the child spends the most nights is the custodial parent, and the other parent is the non-custodial parent.

iv. If tax dependents meet any of the above exceptions, they follow the **non-filer household rules***

b. There are two exceptions that may apply to **tax filers** (a tax filer is someone who files taxes and is not claimed as a dependent on someone else's taxes.)

   i. If the tax filer is unmarried, under the Medicaid child age, and does not claim a dependent. This applicant will follow the **non-filer household rules***

   ii. If the tax filer is married filing separately and live with their spouse. This applicant will **not** follow the non-filer household rules. Information about the spouse needs to be collected but we do not need to ask about other family members.

c. The exceptions described above are determined on an individual basis for each applicant on a financial assistance application, and the household composition for Medicaid and CHIP eligibility may be different for applicants on the same application. In contrast, for APTC determinations, if two members are in the same tax household (a tax filer and the dependent that they claim, for example) then their household composition used for APTC eligibility will be exactly the same.

12. What are the ***non-filer household rules*** for Medicaid and CHIP?

    a. The **adult** non-filer household rules are used for applicants over the Medicaid child age who meet the exceptions described above, as well as applicants over the Medicaid child age who neither file their own tax return nor are claimed as someone else's tax dependent.

       i. The adult's Medicaid and CHIP household must include:

          1. The adult applicant themselves
          2. If the applicant is married, their spouse if living with the applicant
          3. The adult applicant's biological, adopted, and step children under the Medicaid child age if living with the applicant

    b. The **child** non-filer household rules are used for children under the Medicaid child age

       i. The child's Medicaid and CHIP household must include:

          1. The child applicant themselves
          2. The child's biological, adopted, or step parents IF living with the child
          3. The child's biological, adopted, or step siblings who are underthe Medicaid child age IF living with the child
          4. If the child is married, the spouse if living with the child
          5. If the child has their own child(ren), the children and stepchildren if living with the child

13. What are the scenarios where the UI should provide an option to consumers about whether the provide information about certain non-applicant household members, resulting in the household size/composition not being derived by SES?

In most cases the requestor needs to provide sufficient information to the APIs to be able to construct both the Medicaid household and the tax household, but there are limited situations where it is okay for the consumer to provide enough household information for determining only APTC/CSR or only Medicaid/CHIP. This only happens when it is not reasonable to expect the application filer to have the information for both types of households.

Scenario	Type of Applicant	Set Data Element	Interaction	Sample Json Reference
Applicant claimed by non- parent tax filer provides tax filer information but not family information	**Medicaid Child Age**	**liveWithParentOrSiblingindicator**	Set to true when consumer attests to living with a parent or sibling. This should also trigger a question about whether the consumer wants to provide information about the parent/sibling—that answer should set the **medicaidFamilyNotProvidedIndicator** Set to false when consumer attests to not living with parent or sibilings	Figure 18: Example JSON request when the dependent is claimed by non-parent tax filer, lives with parent or sibling, and does not provide Medicaid family Figure 19: Example JSON response when the dependent is claimed by non-parent tax filer, lives with parent or sibling, and does not provide Medicaid family
Applicant claimed by non- parent tax filer provides tax filer information but not family information	**Medicaid Child Age**	**medicaidFamilyNotProvidedIndicator**	Set to true when consumer attests to living with a parent or sibling but chooses not to provide family information. In addition to setting to true, the UI should display a warning message that the system will not be able to determine Medicaid/CHIP eligibility without this information. Set to false when consumer agrees to provide family information. The UII shouldproceed to collect said information	See above
Applicant claimed by non-custodial parent on the applicaiton	**Medicaid Child Age**	**livesWithCustodialParentNotOnApplicationIndicator**	Set to true when a consumer attests to living with a parent not on the application and display a warning message that the system will not be able to determine Medicaid/CHIP eligibility without this information. Set to false when the consumer attests that they do not live with a parent. In this scenario, both Medicaid/CHIP and APTC will follow tax household rules.	Figure 20: Example JSON request when the dependent is claimed by non-custodial parent tax filer and the custodial parent is not on the application Figure 21: Example JSON response when the dependent is claimed by non-custodial parent tax filer and the custodial parent is not on the application

Scenario	Type of Applicant	Set Data Element	Interaction	Sample Json Reference
Applicant claimed by non- parent or non-custodial parent (not the application filer) tax filer	**Medicaid Child Age and Adult**	**taxFilerNotProvidedIndicator**	Set to true when the consumer attests to not providing information about a non-parent or non-custodial parent(that's not on the app) claiming tax filer.This option would only be provided when this tax filer is not the application filer and not an applicant on the application.In addition to setting the indicator, the UI should display a warning message that the system will not be able to determine APTC eligibility without this information. Set to false when consumer providesinformation about the claiming tax filer	Figure 22: Example JSON request when the dependent is claimed by non-parent tax filer that is not provided on the application Figure 23: Example JSON request when the dependent is claimed by non-parent tax filer that is not provided on the application

a. The applicant's Medicaid household cannot be computed when the requestor provides certain responses for the following indicators, in which case the medicaidHouseholdStatus = N in the response. The UI should provide an educational warning message to the consumer to explain that the custodial parent can file a separate application in order to receive an eligibility determination for Medicaid and CHIP.

i. **liveWithParentOrSibling indicator**: When a Medicaid child age applicant is claimed by a non-parent or a non-custodial parent tax filer and that tax filer is the application filer, then application should ask whether the tax dependent applicant also lives with immediate family members such as a parent or sibling. In some cases and applicant might not live with their parents or siblings so the Medicaid household would just be themselves, and we will have enough information complete both Medicaid/CHIP and APTC eligibility. However, if the child applicant does live with their parent or sibling (or spouse or own son/daughter) then the livewithParentOrSibling indicator is set to true and another question is asked to find out whether the application filer would like the provide information about the family members the child lives with or not. Depending on the answer, one of two indicators should be set:

ii. **medicaidFamilyNotProvidedIndicator:** If the child applicant is claimed by a non-parent tax filer, and the livewithParentOrSibling indicator is set to true, then this MedicaidFamilyNotProvidedIndicator is set to either true or false depending on the application filer opts not to provide the family information. Note that if this indicator is set to true, then a determination for Medicaid/CHIP will not be available for the child applicant. The application should inform the application filer that Medicaid and CHIP eligibility cannot be determined without this information

```
{
        "members": {
                "1253474600522039365": {
                        "family": {
                                "claimsDependentIndicator": false,
                                "parentCaretakerIndicator": false,
                                "absentParentIndicator": false,
                                "taxFilerNotProvidedIndicator": false,
                                "taxDependentIndicator": true,
                                "liveWithParentOrSiblingIndicator": true,
                                "medicaidFamilyNotProvidedIndicator": true,
                                "taxFilerIndicator": false,
                                "fosterCareIndicator": false
                        }
                }
        },
        "household": {
                "familyRelationships": [
                        ["1253474600522039364","SELF","1253474600522039364"],
                        ["1253474600522039364","GRANDPARENT","1253474600522039365","resideTogetherIndicator" : true}]
                ],
                "taxRelationships": [
                        ["1253474600522039364","TAX_FILER","1253474600522039364"],
                        ["1253474600522039364","TAX_FILER","1253474600522039365"]
                ]
        }
}
```

Figure 18: Example JSON request when the dependent is claimed by non-parent tax filer, lives with parent or sibling, and does not provide Medicaid family

```
{
    "result": {
        "computed": {
            "members": {
                "1253474600522039365": {
                    "preliminaryMedicaidStatusReason": "562_NO_MEDICAID_HOUSEHOLD",
                    "preliminaryChipStatus": "NO",
                    "preliminaryChipStatusReason": "562_NO_MEDICAID_HOUSEHOLD",
                    "medicaidHouseholdComposition": {
                        "medicaidTaxRoleType": "DEPENDENT_ON_OTHER",
                        "medicaidHouseHoldStatus": "NO",
                        "medicaidHouseholdSize": 0
                    }
                }
            }
        }
    }
}
```

Figure 19: Example JSON response when the dependent is claimed by non-parent tax filer, lives with parent or sibling, and does not provide Medicaid family

iii. **livesWithCustodialParentNotOnApplicationIndicator:** This is similar in impact as the indicator described above, but it should be set to true by the requestor when the non-custodial parent is the application filer for their applicant child and the application filer opts not to provide information about the custodial parent with whom the child lives.

```
{
    "members": {
        "1253474600522039463": {
            "family": {
                "claimsDependentIndicator": false,
                "parentCaretakerIndicator": false,
                "absentParentIndicator": false,
                "taxFilerNotProvidedIndicator": false,
                "taxDependentIndicator": true,
                "livesWithCustodialParentNotOnApplicationIndicator": true,
                "medicaidFamilyNotProvidedIndicator": true,
                "taxFilerIndicator": false,
                "fosterCareIndicator": false
            }
        }
    },
    "household": {
        "familyRelationships": [
            ["1253474600522039462","SELF","1253474600522039462"],
            ["1253474600522039462","PARENT","1253474600522039463", {"resideTogetherIndicator": false}]
        ],
        "taxRelationships": [
            ["1253474600522039462","TAX_FILER","1253474600522039462"],
            ["1253474600522039462","TAX_FILER","1253474600522039463"]
        ]
    }
}
```

Figure 20: Example JSON request when the dependent is claimed by non-custodial parent tax filer and the custodial parent is not on the application

```
{
        "members": {
                "1253474600522039463": {
                        "family": {
                                "claimsDependentIndicator": false,
                                "parentCaretakerIndicator": false,
                                "absentParentIndicator": false,
                                "taxFilerNotProvidedIndicator": false,
                                "taxDependentIndicator": true,
                                "livesWithCustodialParentNotOnApplicationIndicator": true,
                                "medicaidFamilyNotProvidedIndicator": true,
                                "taxFilerIndicator": false,
                                "fosterCareIndicator": false
                        }
                }
        },
        "household": {
                "familyRelationships": [
                        ["1253474600522039462","SELF","1253474600522039462"],
                        ["1253474600522039462","PARENT","1253474600522039463", {"resideTogetherIndicator":
false}]
                ],
                "taxRelationships": [
                        ["1253474600522039462","TAX_FILER","1253474600522039462"],
                        ["1253474600522039462","TAX_FILER","1253474600522039463"]
                ]
        }
}
```

Figure 21: Example JSON response when the dependent is claimed by non-custodial parent tax filer and the custodial parent is not on the application

b. The applicant' tax household cannot be computed when the requestor sets one of the following indicators to true, in which case the taxHouseholdStatus = N in the response. The UI may provide an educational warning message to the consumer to explain that the claiming tax filer can file a separate application in order to receive an eligibility determination for a premium tax credit.

    i. **taxFilerNotProvidedIndicator:**– This indicator is set to true when the applicant is claimed by a non-parent tax filer and the application filer chooses not to provide the tax filer's information.

        1. In order to set these fields appropriately, the requestor would need to prompt users with a question asking if they'd like to provide more information about the claiming tax filer in order to determine their eligibility for APTC. This question is only prompted in the event

that the claiming fax filer is not an applicant, not the application filer, and not the parent of the tax dependent applicant.

```
{
        "members" : {
                "1253521196748091439" : {
                        "family" : {
                                "claimsDependentIndicator" : false,
                                "parentCaretakerIndicator" : false,
                                "absentParentIndicator" : false,
                                "taxFilerNotProvidedIndicator": true,
                                "taxDependentIndicator" : false,
                                "livesWithCustodialParentNotOnApplicationIndicator": false,
                                "taxFilerIndicator" : false,
                                "fosterCareIndicator" : false
                        }
                }
        },
        "household" : {
                "familyRelationships" : [
                        ["1253521196748091438","SELF","1253521196748091438"],
                        ["1253521196748091438","GRANDPARENT","1253521196748091439", {"resideTogetherIndicator" :true}]
                ],
                "taxRelationships" : [
                        ["1253521196748091438","TAX_FILER","1253521196748091438"]
                ]
        }
}
```

Figure 22: Example JSON request when the dependent is claimed by non-parent tax filer that is not provided on the application

```
{
  "result": {
    "computed": {
      "taxHouseholds": {
        "1253521196748091439": {
          "taxHouseHoldComposition": {
            "taxHouseholdStatus": "NO",
            "taxHouseholdStatusReason": "662_CLAIMING_TAX_FILER_NOT_ON_APPLICATION"
          }
        }
      }
    }
  }
}
```

Figure 23: Example JSON request when the dependent is claimed by non-parent tax filer that is not provided on the application

14. Which household combinations are supported by EDE Phase 1 and 2?
    a. The combinations of the questions on the screener make household composition very simple for a Phase 1 launch. We confirm that the application filer is filing taxes and if he/she is married, that they're filing jointly with their spouse. We also confirm through the screener that any other applicants are either a son or daughter claimed as a tax dependent of the application filer or the spouse of the application filer. We also confirm via the screener that they all live together. The screener answers also tell us that there are no exceptions that could cause the household size used for Medicaid and CHIP to differ from the tax household used for APTC. That means we don't need to collect information about anyone other than the tax filer(s) and their dependents.

Responses:

1. Where can the requestor find the results of the tax and Medicaid households?
    a. The computed.taxHouseholds and computed.members.medicaidHouseholdComposition is a map from each tax filer or applicant to their associated household. If there are multiple tax households, there will be multiple keys within computed.taxHouseholds where the taxHouseholdStatus = YES. Member IDs for the associated household members can be found within the taxHouseholdMemberIdentifiers and medicaidHouseholdMemberIdentifiers objects. Note the medicaidHouseholdMemberIdentifiers does not include unborn children which are counted in the medicaidHouseholdSize for pregnancies, but the back-end will update the size of the Medicaid household accordingly.
    b. The results of the tax and Medicaid households can be used to identify "non-relevant non-applicants". If a non-applicant is not included in any applicant's computed.taxHouseholds or computed.members.medicaidHouseholdComposition then they should not be asked questions on the

application after the household composition section. For example, on a single member application where the application filer is married but does not live with their spouse and does not file a joint tax return with their spouse, the SSN and income of the spouse should not be requested.

c. This response of multi-tax households should only be used in the eligibility results page or to identify tax households for income discrepancy questions.

```
taxHouseholds": {
   "1283826392975753443": {
      "taxHouseHoldComposition": {
         "taxHouseholdStatus": "YES",
         "taxHouseholdStatusReason": "999_N_A",
         "taxHouseholdMemberIdentifiers": ["1283826392975753443"]
      },

   },
   "1283826392975753442": {
      "taxHouseHoldComposition": {
         "taxHouseholdStatus": "YES",
         "taxHouseholdStatusReason": "999_N_A",
         "taxHouseholdMemberIdentifiers": ["1283826392975753442"]
      },

      }
   }
}
```

Figure 24: Example JSON Response when there is a mutli taxhousehold in an application(MFS or 2 Individual Filers)

```
"taxHouseholds": {
  "1283827598913679620": {
    "taxHouseHoldComposition": {
      "taxHouseholdStatus": "YES",
      "taxHouseholdStatusReason": "999_N_A",
      "taxHouseholdMemberIdentifiers": ["1283827598913679620",
      "1283827598913679621"]
    },
```

Figure 25: Example JSON Response when there is a single taxhousehold in an application (MFJ)

```
"taxHouseholds": {
  "996279987396202194": {
    "taxHouseHoldComposition": {
      "taxHouseholdStatus": "NO",
      "taxHouseholdStatusReason": "664_NON_APPLICANT_FAMILY_MEMBER_OF_NON_FILER_NO_TAX_HH"
    },

  },
  "996279987396202198": {
    "taxHouseHoldComposition": {
      "taxHouseholdStatus": "YES",
      "taxHouseholdStatusReason": "999_N_A",
      "taxHouseholdMemberIdentifiers": [
        "996279987396202198"
      ]
    },
  }
```

Figure 26: Example JSON Response when there is a tax filer with a non-filer on the application

2. How to read relationships when in the response from the API?
   a. After the requestor provides the Family Relationship between two members in one direction, the API determines the mirror or inverse relationships between the members as follows in the response:
      i. "PARENT" <-> "SON_DAUGHTER"
      ii. "STEP_PARENT" <-> "STEPSON_STEPDAUGHTER"
      iii. "PARENTS_DOMESTIC_PARTNER" <-> "CHILD_OF_DOMESTIC_PARTNER"
      iv. "AUNT_UNCLE" <-> "NEPHEW_NIECE"
      v. "GRANDPARENT" <-> "GRANDCHILD"
      vi. "SIBLING" <-> "SIBLING"
      vii. "UNRELATED" <-> "UNRELATED"
      viii. "DOMESTIC_PARTNER" <-> "DOMESTIC_PARTNER"
      ix. "SELF" <-> "SELF"
      x. "FIRST_COUSIN" <-> "FIRST_COUSIN"
      xi. "OTHER_RELATIVE" <-> "OTHER_RELATIVE"
      xii. "SPOUSE" <-> "SPOUSE"
   b. After the request provides the tax relationship between two members in one direction the API determines the mirror or inverse relationships between the members as follows in the response:
      i. “TAX_FILER” <-> “TAX_DEPENDENT”

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on household composition:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	attestations.household member.family	Input Matrix: Household Composition	Phase 1 Questions, Phase 2 Questions, UI Questions, & Requirements	Household	3.1 – Eligibility Event Dependencies 3.3 – Application Navigation
Responses	computed.taxHouseholds & computed.members.medicaidHouseholdComposition		Backend Responses	Household	
Validations		Input Matrix: Service & Flow Validations			5.2.4 – Exception Handling
Clearing					3.3.5 – Backwards Navigation
Reference Data	under21AndFTSOption medicaidChildAgeThreshold medicaidChildAgeStudentThreshold				3.4 Application State and System Configuration
API Interaction					

Citizenship/Immigration:

One of the requirements for eligibility for a QHP as well as for Medicaid and CHIP is US Citizenship, US National status or eligible immigration status. (Applicants who do not have an eligible immigration status may still be eligible for emergency Medicaid services). Applicants for health coverage are asked questions that the Social Security Administration (SSA) and the Department of Homeland Security (DHS) use to verify attested citizenship or immigration status.

Based on the applicant's attestations and their Social Security Number (SSN), SES calls SSA to verify an applicant's SSN and US citizenship. The response from SES will instruct the requestor on whether to re-ask the SSN question and/or whether to ask the naturalized or derived citizenship question. If an applicant attests to being a non-citizen, they must be asked to attest to additional information that may be used to initiate a call to DHS. Additional information may include an applicant's immigration documentation, immigration status, how long they've been in the country and/or whether they are personally related to or are an active duty member of the US military or honorably discharged veteran. After SES calls DHS, DHS' response could prompt the requestor to ask the applicant for clarifying information or even necessitate that the requestor initiate additional verification procedures.

When a consumer provides information related to SSN, Citizenship, or Immigration status, the requestor should populate the appropriate inputs and call the Update App API.

Inputs:

1. What should the requester send to the API if an application member does not provide their SSN?
   a. The requester should leave the SSN indicator as null.
   b. SSN = null

2. What should the requestor send to the API if a non-citizen applicant chooses not to answer the question regarding eligible immigration status?
   a. If the applicant prefers not to answer the question regarding eligible immigration status, then the requestor can set lawfulPresenceStatusIndicator = null. As long as citizenshipStatusIndicator for this applicant has been set to false, then the null response for the lawful presence indicator will not cause any error. The back-end eligibility logic has validations to ensure that both the citizenshipStatusIndicator and lawfulPresenceStatusIndicator <> true or null for an applicant.

      The UI should require an answer to a question that populates the citizenshipStatusIndicator first. If the applicant is a non-citizen, then the lawfulPresenceStatusIndicator can be set as well. The applicant must be able to leave the question about their eligible immigration status blank and the UI should not include a response option for "no". The requestor should provide the following inputs to the API for the lawful presence indicator to avoid a validation error from the back-end logic:
      i. If the citizenshipStatusIndicator = true, the lawfulPresenceStatusIndicator = null
      ii. If the citizenshipStatusIndicator = false, the lawfulPresenceStatusIndicator = true or null

3. Can the applicant choose to not provide an immigration or citizenship document type attestation?
   a. Yes, if the applicant attests to lawfulPresenceStatusIndicator = true the UI may include radio buttons for the applicant to attest 'None of these' or 'I don't know' regarding their lawfulPresenceDocumentation or allow the consumer to continue without selecting a document option.
   b. If an applicant chooses 'None of these' or 'I don't know' and does not provide an alien number or I-94 number, the requestor should set lawfulPresenceDocumentation = null. SES will be unable to gather enough

information to make a call to DHS, and the applicant will not be able to have their attestation automatically verified by the Department of Homeland Security. If the consumer is QHP eligible, a data matching issue will be generated and paper documentation requested.

```
{
        "application": {
                "accountHolderIdentityProofedIndicator" : true
        },
        "members": {
                "123456789": {
                        "lawfulPresence": {
                                "citizenshipIndicator": false,
                                "lawfulPresenceStatusIndicator": true
                        }
                }
        }
}
```

Figure 27: Example JSON request when the applicant attests to 'None of these' or 'I don't know.' And does not provide I-94 or alien number

4. If an applicant provides an Alien Number or I-94 number without a documentType, what should the requestor set as the documentType?
    a. The requestor should set lawfulPresenceDocumentation = OTHER and provide otherTypeText
        i. This will provide enough information to call DHS and potentially verify lawful presence status.

```
{
        "application": {
                "accountHolderIdentityProofedIndicator": true
        },
        "members": {
                "123456789": {
                        "lawfulPresence": {
                                "citizenshipIndicator": false,
                                "lawfulPresenceStatusIndicator": true,
                                "lawfulPresenceDocumentation": {
                                        "OTHER": {
                                                "alienNumber": "823156229",
                                                "otherTypeText": "Other Document"
                                        }
                                }
                        }
                }
```

Figure 28: Example JSON request when the applicant attests to 'None of these' or 'I don't know.' And ***provides*** *alienNumber*

5. If an applicant provides an Alien Number less than 9 digits, how should the requestor send it?
    a. The requestor must adhere to the SES validation requiring alienNumber to be 9 digits and numbers only.
    b. For consumers that have an alienNumber that is less than 7 or 8 digits, the requestor should add leading zeros prior to submitting the alien number in the API request, but this can be done invisibly to the consumer

6. What should the requestor send to the API if the applicant provides the document type but chooses not to provide the fields required (numbers and country) for an immigration document to verify immigration status with DHS?
    a. The required data elements in the Immigration Status Reference Table (Appendix X) should be optional fields in the UI. The requestor should populate the lawfulPresenceDocumentation with the document numbers provided by the user, or if the user does not provide any document numbers, set the noIdentifiersProvidedIndicator = true.
    b. Note the UI should include a warning messaging to the consumer not providing full citizenship or immigration document information ras described in item 74 of the UI Question Companion Guide.

7. When should the requestor populate the livedinUS5yearIndicator?
    a. Despite the name, this data element is related to whether the non-citizen applicant has lived in the US since 1996 (not for 5 years, though the question relates to an exception to the five year bar for non-citizens' Medicaid and CHIP eligibility). Therefore, the UI does not need to populate the livedinUS5yearindicator for applicants who were born on or after August 22, 1996. For older non-citizen applicants, the indicator should be populated as true, false or null.

8. When should the requestor populate the veteran indicator?
    a. The veteranIndicator is required by the back-end when the applicant meets the criteria to call DHS and they attest to moving to the U.S. after 8/22/1996 (livedInUS5yearIndicator = false or null). We recommend that the requestor set the veteranIndicator for every non-citizen applicant. Since the question is optional the requester should set the indicator to false if the consumer chooses not to answer the question

9. If a non-citizen applicant is not a veteran but their spouse or parent is a veteran, should the veteranIndicator be set to true for the non-veteran applicant and/or for their veteran family member?
    a. Both the veteranIndicator and veteranSelfIndicator should be set to true for a non-citizen applicant if that applicant attests to being an active duty service member or veteran. Only the veteranIndicator should be set to true for a non-citizen applicant if the non-citizen applicant's spouse (or former spouse) is an active duty service member or veteran. Similarly, if the non-citizen applicant is a tax dependent claimed by their parent and the claiming tax filer is an active duty service member or veteran, the veteranIndicator should be set to true for the child and the veteranSelfIndicator should be set to true for the parent. If the parent is not on the application, then it won't be possible to set the veteranSelfIndicator for the parent. Instead, the nonMemberVeteranRelationshipTypes should be set for the child as well as the veteranIndicator.
        i. Here is an example: George (member ID: 123456789) is applying for coverage along with his wife, Jane (member ID: 456789012), and teenage daughter, Judy (member ID: 678901234), whom he claims as a dependent. All three are non-citizens. If Jane is a US Army veteran, then the

veteranIndicator should be set to true for George as well as Jane and Judy. If Jane, the veteran, is not applying for coverage or is a US citizen, then the veteranIndicator does not need to be set to true for her, but should still be set to true for George and Judy.

```
{
    "application": {
        "accountHolderIdentityProofedIndicator": true
    },
    "members": {
        "123456789": {
            “requestingCoverageIndicator: true,
            “lawfulPresence”: {
                "citizenshipIndicator": false,
                " lawfulPresenceStatusIndictor": true
            },
            "other": {
                “veteranIndicator: true
            }
        },

        "456789012": {
            “requestingCoverageIndicator: true,
            “lawfulPresence”: {
                "citizenshipIndicator": false,
                " lawfulPresenceStatusIndictor": true
            },

            "other": {
                “veteranIndicator: true,
                “veteranSelfIndicator: true
            }
        },

        "678901234": {
            “requestingCoverageIndicator: true,
            “lawfulPresence”: {
                "citizenshipIndicator": false,
                " lawfulPresenceStatusIndictor": true
            },
            "other": {
                “veteranIndicator: true
            }
        }
    }
}
```

ii.

iii. While only the veteranIndicator is ever required in an API request, in order to prepopulate the veteran indicator UI question on CiC and backward navigation, the requestor may wish to provide additional information in the API request to reflect how the consumer answered the UI question. The following indicators may be provided via Update App API:

iv. veteranSelfIndicator - Indicates if the household member attests to being a veteran, even when that household member is not an applicant or is not a non-citizen. This is an optional indicator, and should only be provided when relevant based on the response to the existing UI question asked for non-citizen applicants in Item 73 of the UI Question Companion Guide.

v. nonMemberVeteranRelationshipTypes - Indicates the relationship to a person not on the application who is a veteran, when the non-citizen applicant attests in item 73 that someone not on the application (such as a parent or deceased spouse) is a veteran.

10. How should the requestor update their API requests if an applicant changes their attestation from a non-citizen to an attested citizen?
    a. If an applicant:
        i. Attests to being a non-citizen and provides documentation to verify their lawful presence status
        ii. Later backwards navigates, performs a CIC or re-enrolls the application and their citizenship status changes (ie. they attest to being a US Citizen and provide documentation regarding their naturalization)

    When the UI runs Get App to pre-populate the application, the UI should null out the previou non-citizenship attestation and document information before calling update app with the new naturalization attestations.
11. How should a requestor populate non-SAVE verifiable documents?
    a. The following documents are considered non-SAVE verifiable and may be attested per item 71 of the UI Q CG:
        i. ORR_ELIGIBILITY_LETTER
        ii. CUBAN_HATIAN_ENTRANT
        iii. NS1_MEMBERS_OF_A_FEDERALLY_RECOGNIZED_INDIAN_TRIBE
        iv. NS4_NON_CITIZEN_WHO_IS_LAWFULLY_PRESENT_IN_AMERICAN_SAMOA
        v. VAWA_SELF_PETITIONER
    b. An applicant may attest to multiple non-SAVE documents in addition to one SAVE document. To populate a non-SAVE document in the request, the requestor must include the "noIdentifiersProvided" = True under the non-SAVE document.

```
{
    "application": {
        "accountHolderIdentityProofedIndicator": true
    },
    "members": {
        "123456789": {
            "lawfulPresence": {
                "citizenshipIndicator": false,
                "lawfulPresenceStatusIndicator": true,
                "lawfulPresenceDocumentation": {
                    "VAWA_SELF_PETITIONER": {

                        “noIdentifiersProvided”: “true”},
                    "FOREIGN_PASSPORT": {
                        "passportNumber": "12345689080",
                        "passportIssuingCountry": “CAN”
                        "documentExpirationDate": "2017-01-01"}
                }
            }
        }
    }
}
```

Figure 29: Example JSON request when the applicant attests to a non-SAVE document

Responses:

The UI uses the API responses to prompt the following questions within the Citizenship/Immigration section of the application:

- **Social Security Number (SSN) retry** is needed when SSA has a partial match on data submitted for verification.
- **Naturalized citizen follow-up attestation** is asked when SSA cannot verify citizenship for an attested citizen.
- **Grant Date of the applicant’s current immigration status** is needed when the applicant has an immigration status where the 5-year bar is applicable and the Hub indicates their 5-year bar met status is pending.

1. When has the requestor collected enough data to receive the information about SSN and citizenship status verification in order to prompt follow-up questions if needed?
    a. The Hub call to SSA can occur after the following data has been collected:
        i. firstName <> null
        ii. lastName <> null
        iii. birthDate <> null
        iv. ssn <> null
        v. accountHolderIdentityProofedIndicator = true
        vi. requestingFinancialAssistanceIndicator <> null

vii. citizenshipIndicator <> null (applicants only)

2. When has the requestor collected enough data to receive the information about immigration status verification in order to prompt follow-up questions if needed?
    a. The Hub call to DHS can occur after the following data has been collected:
        i. firstName <> null
        ii. lastName <> null
        iii. birthDate <> null
        iv. accountHolderIdentityProofedIndicator = true
        v. requestingCoverageIndicator = true
        vi. requestingFinancialAssistanceIndicator <> null (conditional)
        vii. lawfulPresenceStatusIndicator = true OR naturalizedCitizenIndicator = true (conditional)
        viii. lawfulPresenceDocumentation <> null (including the required data elements per the Hub's BSD for VLP)
        ix. veteranIndicator <> null (conditional)
    b. Please see the Application Services Companion Guide for when conditional data elements are required.

.

3. What response from the API should prompt the question for SSN retry?
    a. Display the SSN retry question when the ssnStatusReason = 634_ SSA_DATA_MISMATCH.
    b. For a US citizen applicant where the response received from Update App is a 634, the UI should display the SSN retry question and do another Update App call with the consumer's revised attestations, if any. If the consumer with the 634 response is a non-citizen or a non-applicant, then the retry opportunity should be presented to the applicant prior to the income section (because a verified SSN is required to make any subsequent hub calls for other verifications).
    c. The API will return ssnStatus = Y and ssnStatusReason = 573 when the requestor exceeds the maximum limit to call SSA within 24 hours (3 times). In this case, if the consumer is a US citizen applicant, the UI should ask about naturalized citizenship and move forward with the application flow.

4. What response from the API should prompt the question for Naturalized/Derived Citizenship?
    a. Display the Naturalized/Derived Citizenship question when the **citizenshipIndicator** = true and **citizenshipStatusReason** <> 999
    b. The naturalizedCitizenIndicator is required by the API when the **ssnStatusReason** = 999_N_A and the **citizenshipStatusReason** = 177_CITIZENSHIP_NOT_VERIFIED.
    c. If an applicant receives a citizenshipStatusReason = **999**, then no further citizenship or immigration questions are necessary for that member.

5. How should the UI respond if an applicant changes birthdate in response to a UI prompt based on DHS or SSA response and the changed age means they should have been asked different household composition questions or other questions based on age?

a. The UI should determine if the change in birthdate changes the age to above or below one of the age cutoffs that determine when to prompt a question. Age cutoffs are either hardcoded or provided in system state configuration data. The SES Get States and System Reference Data API contains all of the state and system configuration age cutoffs.

b. If an applicant changes birthdate in response to a UI prompt based on DHS or SSA, the UI must consider whether any of the following questions are impacted by the birthdate change and, if so, return the consumer to the beginning of the relevant section. Any prior answers based on the original question sequence should be cleared before submit if the questions are not in the new question sequence as a result of the age change.

c. If birthdate/age is not used to prompt different questions in the UI, or if the applicant's new age does not require different questions then the applicant does not have to re-attest.

6. How should the UI requestor proceed when receiving an SSN or Citizenship status reason of 573_SSA_HUB_CALL_HELD_CALL_COUNTER_3_OR_MORE?
   a. This status reason indicates that SES has reached the limit (3 tries) for requesting SSA data within a given day. UI requesters can call Update App up to 3 times within a session OR until statusReason 573_SSA_HUB_CALL_HELD_CALL_COUNTER_3_OR_MORE is returned in the API response. At this point, the UI should proceed to the next question in the flow. This status will not prevent the application from being successfully submitted. After 24 hours the UI requestor may call Update App to receive SSA verification for SSA data e.g SSN.

Citizenship/Immigration Questions and Age Change Impacts

The following question is impacted if the applicant's birthdate changes to before or after 8/22/1996 (hard coded date):

i. Has [FNLNS] lived in the U.S. since 1996?

The following question is impacted if the applicant's birthdate changes to under 17 or 17 and over OR if applicant is unmarried and age changes to under 15 or 15 and over (hard coded ages):

ii. Are any of these people an honorable discharged veteran or active duty member of the military?

1. What response from the API should prompt the question for Step 1b (SEVIS)?
   a. If the API returns qhpLawfulPresenceStatusReason = 566_SEVIS_ID_REQUIRED, the UI should prompt the user to provide/edit the SEVIS ID. The requestor should set the SEVISid field to the number that the consumer provides in response to this question.

2. What response from the API should prompt the question for Grant Date?
   a. If the API returns qhpLawfulPresenceStatusReason =
      i. 682_ FIVE_YEAR_BAR_ PEND_NEED_GRANT_DATE_AND_LP_EXPIRE_AFTER_90_DAYS

   ii. 683_FIVE_YEAR_BAR_PEND_NEED_GRANT_DATE_AND_TEMP_LP_EXPIRE_WITHIN_90_DAYS

b. the UI should ask the consumer for the approximate date when he or she received their current immigration status. The requestor should set this date in MM-YYYY form for the lawfulPresenceGrantDate field.

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on SSN and citizenship/immigration:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	attestations.members.lawfulPresence members.demographic	Input Matrix: SSN, Citizenship, Immigration, & Preliminary Eligibility	UI Questions & Requirements	Household	3.1 – Eligibility Event Dependencies 3.2 – Application Services 3.3.4 – Option for Consumer Not to Provide Information
Responses	computed.members. and qhpLawfulPresence and fiveYearBar and medicaidLawfulPresence and emergencyMedicaid		Backend Responses for UI: Items 1-5	Household	3.3.2 – Dynamic Questions 3.3.3 – Backend Interaction
Validations		Input Matrix: Service & Flow Validations			2.5 – Application Operational Assistance 5.2.4 – Exception Handling
Clearing					3.3.5 – Backwards Navigation
Reference Data	*See Immigration Status Reference Table in FAQ				*See Immigration Status Reference Table in FAQ
API Interaction	N/A			Household	4 – Sample UI Interactions with Application Services 5.2.3 – Technical Assumptions

More about this household

This section in the UI collects the SSN, Sex, Race/Ethnicity, and Disability/Long-Term Care attestations.

In addition, the UI must include questions about whether the applicant was recently denied Medicaid or CHIP by the state, and if not, then whether the applicant had Medicaid or CHIP coverage that recently ended or will end soon. The purpose of collecting these attestations is to block the applicant's Medicaid/CHIP eligibility when the state has recently determined the applicant is ineligible, so the applicant is not referred back to the state through Account Transfer (AT) unless there has been a change in the applicant's household income or size since the determination. If an applicant was not denied coverage but had Medicaid/CHIP coverage that recently ended or will end soon, we also want to determine whether the applicant is eligible for a Special Enrollment Period (SEP) based on termination of Medicaid or CHIP coverage.

Inputs:

1. When should the appliedDuringLifeChangeOrOeIndicator be set?
    a. If a consumer attests that they were denied Medicaid or CHIP by the state agency in the last 60 days, then they may be eligible for the Medicaid Denial SEP. In order to determine SEP eligibility, we also need to know when they applied for coverage. The UI should ask whether the consumer applied during the most recent OE period or applied following a life change. When OE ends each December, the application question should be updated with the dates of the most recent open enrollment period.
    b. If the UI splits the Open Enrollment and
    c. The questions about when the applicant applied for coverage do **not** need to be asked (and the indicator does not need to be set) if the applicant was denied Medicaid/CHIP more than 60 days ago or if the applicant is attesting to a recent loss of Medicaid/CHIP rather than a denial.

2. How should the request be structured if the applicant indicates that they had Medicaid or CHIP coverage that ended or have coverage ending soon, rather than having been denied coverage?
    a. Requestors have two options for how to set the indicators for the Medicaid denial question flow. While either option should work from an eligibility perspective, we recommend implementing the new indicators as described in Option 2 (paragraph ii) because it will allow each attestation to be saved using a unique indicator, which facilitates pre-population upon backwards navigation and CiC.
    b. For UI requestors that choose to <u>not implement</u> the new Medicaid Block fields:
        i. If the applicant indicates their household income or size has changed since notice of termination:
            1. Set medicaidDeniedIndicator = false AND
            2. Do not set the medicaidDeniedDate
        ii. If the applicant indicates their household income and size have not changed since notice of termination:
            1. Set medicaidDeniedIndicator = true AND
            2. Do not set medicaidDeniedDate
    c. For UI requestors that choose to implement the new Medicaid Block fields:
        i. If the applicant indicates their household income or size has changed since notice of termination:
            1. Set medicaidDeniedIndicator = false AND
            2. Set informationChangeSinceMedicaidEndedIndicator = true AND
            3. Set medicaidEndIndicator=true AND
            4. Set medicaidEndDate
        ii. If the applicant indicates their household income and size have not changed since notice of termination:
            5. Set medicaidDeniedIndicator = false AND

6. Set informationChangeSinceMedicaidEndedIndicator = false AND
7. Set medicaidEndIndicator=true AND
8. Set medicaidEndDate

d. In each case above (in paragraphs b and c), the loss of MEC change in circumstance fields should also be set:

i. If the applicant's attested coverage end date is on or before the current date and within the last 60 days, set:

1. members[memberID].attestations.other.changeInCircumstance["LOSS_OF_MEC"] and the respective changeDate = consumer's attested end date of coverage.

ii. If the applicant's attested coverage end date is in the future from current date and is in the next 60 days, set:

1. members[memberID].attestations.other.changeInCircumstance["FUTURE_LOSS_OF_MEC"] and the respective changeDate = consumer's attested end date of coverage.

2. How should the UI set the medicaidDeniedDueToImmigrationIndicator in the request?

a. For all non-citizen applicants with lawfulPresenceStatusIndicator=true on financial assistance applications, the requestor should determine whether the applicant has been denied Medicaid due to immigration status in the past 5 years and is likely still ineligible for Medicaid/CHIP on that basis, by following the flow of questions as outlined in the UI question companion guide and the App 3.0 Medicaid block flow.

3. When should indicators related to former foster care be set?

a. The fosterCareIndicator only needs to be set to true or false for applicants aged 18-25; it should be left null for all other consumers.

b. For applicants aged 18-25 (inclusive) who attest to being in foster care the fosterCareIndicator should be set to true

i. The requestor also must set fosterCareState equal to the state in which the applicant was in foster care. The partner should include all 50 states (plus DC) in the answer options for Item 150 which asks in which state the consumer was in foster care. The partner does not need to use State Reference Data API to determine the state names to display. The policy around Medicaid due to former foster care depends on the state where the consumer is currently applying for coverage, not the state where they had foster care. For example, if a consumer now lives in Pennsylvania and is applying on a Pennsylvania application, when we ask where he or she was in foster care the consumer might say they had foster care in Maryland until age 21. When determining whether this applicant will be Medicaid eligible, we will need to look only at the state options only for Pennsylvania. Because Pennsylvania does not require in-state foster care and has an age out at 21, this applicant will likely be eligible for Pennsylvania Medicaid when they submit their application.

ii. If the applicant had Medicaid while in foster care the requestor should set medicaidDuringFosterCareIndicator to true and then should set fosterCareEndAge equal to the age in which the applicant left foster care. The requestor will receive an error if the fosterCareEndAge is a number greater than 26.

4. How should the incarceration status be set for each member on the application?
    a. If any applicant attests to being incarcerated then at an application level the **nonIncarcerationAgreementIndicator** should be set to false as well as the individual **incarcerationType** for each applicant at the member level.

5. When should the pregnancyIndicator be set by the requester?
    a. Provide the indicator for all female applicants and non-applicants that meet the age requirement (age 9 and up) on a financial assistance application. If the pregnancyIndicator is set to true for a male application member then the requestor will receive an error.
    b. If the pregnancy question is optional in the UI, and a household member does not attest to being pregnant then the requestor should default the pregnancyIndicator to false.
        i. If a household member attests to being pregnant but does not provide the number of expected babies it will be defaulted to 1 by SES.

6. When and how should the two parentWeeklyWorkHourQuantity fields be set?
    a. Overall this information should be sent to the API in a state where deprivationRequirementRetained = Y, where an adult applicant is preliminarily eligible for Medicaid and is a parent or caretaker relative of a child who lives with both parents.
    b. The parentWeeklyWorkHourQuantity fields should be set under the child's member object.
    c. For whose parents should parent1WeeklyWorkHourQuantity and parent2WeeklyWorkHourQuantity information be collected from?
        i. **Non-Applicant PCR Children** – if the child was added to the application with no identified parents (such as when the child was only added because an adult applicant attests to being their caregiver on a Phase 2 or 3 application) then the UI should determine if this child resides with both of its parents and set the **resideWithBothParentIndicator**. The scenario only occurs when this child is added as a PCR child with non-parent relationship identified between them and their caretaker. If the application filer attests that the child lives with both parents (who are not on the application) then their work hours should be collected since they will not be collect in Income Section.
        ii. **Applicant PCR Children –** if the child is on the app with both parents and live with them (identified through the familyRelationship attestation) the work hours will first be collected through the Income section. If these hours are not reported in the income section then there should be a second chance to collect them if necessary later on in the application. (See item 4 of Medicaid Specific Questions)
        iii. The parent1WeeklyWorkHourQuantity and parent2WeeklyWorkHourQuantity will be set under the child.

7. When should the UI collect other addresses and transient addresses?
    a. In the event that not all applicants live together at the same address (Phase 2 and 3 only), we need to collect the address of each applicant to store in the homeAddress fields. If an applicant (other than the application filer) attests to having no home address, then the noHomeAddressIndicator should be set to true and the mailing address for that applicant must be collected and set in the mailingAddress field. In

the event that an applicant enters an address outside of the application state, a requestor will need to determine if they're moving back to the state and if so, collect the address they're moving back to and store that in transientAddress.

b. It is also recommended to ask all non-applicants whether they live at the same address as the household contact so that the resideTogetherIndicator can be appropriately set within each family relationship between applicants and non-applicants. Setting the resideTogetherIndicator is important for purposes of household composition for applicants claimed by parents; for household composition of applicants following non-filer rules; and for parent/caretaker relative eligibility for applicants who claim or take care of children. However, if the application is able to collect sufficient information to always set the resideTogetherIndicator appropriately without asking for the home address of non-applicants in this address collection question flow, then it is not required to include all non-applicants in this question flow. However, the address of a tax filer must always be collected.

8. When should the UI ask the full-time student question?
    a. As we've referenced a few times already in the document, in certain states, some 19 and 20 year-olds who are full-time students can be considered children for the purposes of Medicaid/CHIP household composition (see Medicaid Child age in glossary and Under21AndFTS in state reference data API). In addition, student status impacts Medicaid eligibility for a parent or caretaker of an 18 year-old, and student status impacts residency rules for 18-22 year old applicants in some states (see OptionStudentResidency in the state reference data API). Therefore, Phase 2 and 3 applications need to add a student status question for all applicants 18-22 years old, and non- applicants 18-20 If the 18-22 year-old is a student, the requestor must set the fullTimeStatusIndicator to true).
        i. This attestation should be collected prior to running the household composition to set the Medicaid child age correctly according to the state configuration in the SES logic.
        ii. Additionally, in a subset of states we need to add follow up questions, see additional context in the App 3 question flows. In states with optionStudentResidency = Yes the requester should set the parentOfStudentLiveInStateIndicator

Responses:

The UI uses the API responses to prompt the following questions within the More about this household section of the application:

1. Parent/Caretaker Relative
    a. We need to give every adult applicant the opportunity to attest to being a parent or caretaker relative of a child in order to accurately determine their eligibility for Medicaid. SES helps requestors to derive whether based on the relationships already collected, each adult applicant has or has not been determined a parent or caretaker relative of a child.
        i. Our recommendation is that a requestor use the Update App response after household composition to determine whether or not you need to ask the Parent/Caretaker Relative questions. For each applicant over 18, check if the parentCaretakerCategoryStatus is equal to YES,

and if so, don’t ask the Parent/Caretaker relative questions of those applicants as they’ve already been determined by SES to be parents/caretaker relatives

ii. In the event that not all adult applicants have that status equal to yes, we add two questions to determine if any adult applicant is the primary person taking care of a child on the application, or a child not yet added to the application and use those attestations to set the parentCaretakerIndicator = TRUE on the adult’s member record, and to add both the caretakerRelativeIndicator=TRUE and resideTogetherIndicator=TRUE when establishing the family relationship between the two members

```
{
"household": {
"familyRelationships": [

      [ "1234", "SON_DAUGHTER", "5678", { "resi
deTogetherIndicator": true, "caretakerRelativeIndicator": true }]
      ]
   }

}
```

Figure 30 Attesting to ParentCaretaker

b. What relationships should be identified between an attested Parent Caretaker and the child they attest to taking care of ?

i. Parent
ii. Stepparent
iii. Parent’s Domestic Partner
iv. Brother/Sister
v. Uncle/Aunt
vi. Nephew/Niece
vii. First Cousin
viii. Grandparent
ix. Other Relative
x. Other unrelated

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on sex, race/ethnicity, Medicaid block, and non-MAGI:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	attestations.household member.family application.legalAttestations members.other members.medicaid	Input Matrix: Household Composition, Preliminary Eligibility, Incarceration, Medicaid CHIP Standard	UI Questions & Requirements	Household	3.1 – Eligibility Event Dependencies 3.2 – Application Services
Responses	result.computed.members.medicaidChipStandard			Household	3.3.2 – Dynamic Questions
Validations		Input Matrix: Service & Flow Validations			2.5 – Application Operational Assistance 5.2.4 – Exception Handling
Clearing					3.3.5 – Backwards Navigation
Reference Data	fosterCareAgeThreshold inStateFosterCareRequired under21AndFTSOption deprivationRequirementRetained				3.4 Application State and System Configuration
API Interaction	N/A			Household	

Income

The income attestations of all household members, including non-applicants, are required to determine eligibility for financial assistance. First in this section, the application requests each household members' current month income and deduction sources individually. The requestor sends SES the current income attestations through the Update App API, including both frequency and amount for each income type and SES calculates the projected monthly and annual net income. SES also verifies the income attestation against external data sources (ie. Equifax, IRS, SSA) and uses that information to determine whether consumers may subsequently need to provide paper documentation proving their income attestation. Sometimes, a consumer can prevent the need to send paper documentation by explaining a discrepancy between attested income and external data sources. SES will return responses to indicate whether there is an income discrepancy that may addressed by the user through an answer on the application.

Inputs:

1. How should the requestor update the API after collecting the currentIncome attestations for a member, and before the variableIncomeIndicator or unknownIncomeIndicator is collected?
    a. The requestor should provide all required fields for each currentIncome object based on the incomeSourceType and incomeFrequencyType.
    b. Note, per item #181 of the UI Questions tab of the UI Questions Companion guide, the SES calculated value must then be displayed for the consumer to agree or disagree with.
        i. computed.members.[memberid].income.attestedAnnualizedAptcIndividualIncomeAmt
    c. The requestor can populate the annualTaxIncome object after the consumer has a chance to agree or disagree with the calculated annual income and the applicant has attested to their own annual income number if they disagreed.

2. Which fields are required within the currentIncome object? Are the required fields conditional or always the same?
    a. All currentIncome records must include the incomeAmount, incomeFrequencyType, and incomeSourceType.
    b. In addition, the employerName is required if the incomeSourceType = Job, averageWeeklyWorkHours is required if the incomeFrequencyType = Hourly, and averageWeeklyWorkDays is required if the incomeFrequencyType = Daily.
    c. The incomeAmount may be negative or positive values for a subset of incomeSourceTypes that can result in a net profit of loss: self-employment, farming/fishing, capital gains, rental/royalty.
    d. For deductions, the incomesourceTypes should always be negative: Alimony_Payment, Other_Deduction, Student Loan Interest.

```
"members" : {
        "123456789" : {
                "income" : {
                        "currentIncome" : {
                                "currentIncome1" : {
                                        "incomeAmount" : 25000,
                                        "incomeSourceType" : "JOB",
                                        "incomeFrequencyType" : "ANNUALLY",
                                        "jobIncome" : {
                                                "employerName" : "Example Company"
                                        }
                                }
                        }
                }
        }
}
```

Figure 31 Required fields for currentIncome

3. How should the requestor provide scholarship income in the request?
    a. If any household member reports scholarship income, the application should allow the application filer to attest to how much of the scholarship income is used to pay educational expenses. ‘Scholarship’ income used to pay educational expenses is collected as a portion of the gross amount for ‘Other’ income. Therefore, to prevent under/over counting scholarship income, requestors should provide two separate currentIncome objects with the same incomeFrequencyType; one where the incomeSourceType = OTHER_INCOME (which should include all scholarship income both used for educational expenses or not) and one where incomeSourceType = SCHOLARSHIP (which should just include the amount used for educational expenses). The reported scholarship income should not exceed the reported other income.

```
"members" : {
            "123456789" : {
                        "income" : {
                                    "currentIncome" : {
                                                "currentIncome1" : {
                                                            "incomeAmount" : 9000,
                                                            "incomeSourceType" : "OTHER_INCOME",
                                                            "incomeFrequencyType" : "ANNUALLY",

                                                },
                                                "currentIncome2" : {
                                                            "incomeAmount" : 8000,
                                                            "incomeSourceType" : "SCHOLARSHIP",
                                                            "incomeFrequencyType" : "ANNUALLY",

                                                }
                                    }
                        }
            }
}
```

Figure 32 Sending in Scholarship Income

In the above scenario the member is attesting to having a 9000 scholarship but only using 8000 for educational purposes

4. If this same customer comes back to make a life change, how should we pre-populate the income details? When we call the Get Application Details service, the response is going to include at least two entries. How do we display the non-scholarship income on the UI?
    a. There are a few different scenarios of what could be provided in Get App and how we would recommend the UI prepopulate the information when the consumers reports a life change. Unfortunately the information available through Get App API will not always be sufficient to know how to pre-populate the UI so in one of the scenarios below the UI could either ask a tailored question to confirm the type of “other income”, or the UI could clear the previous “other income” attestation and require the consumer to re-attest.

        i. When the Get App response includes a current income object where incomeSourceType=scholarship and only one current income object where incomeSourceType=other_income, then the UI can populate the amount for incomeSourceType=other_income in the UI field for scholarship income and populate the amount for incomeSourceType=scholarship in the UI field that asks for the amount of the scholarship which is used to pay for educational expenses.

ii. When the Get App response includes a current income object where incomeSourceType=scholarship and includes more than one current income object where incomeSourceType=other_income, then the UI can populate the amount value for the incomeSourceType=scholarship in the UI field which asks for the amount of the scholarship used for educational expenses. However, we recommend that the UI clear out the line above it (where the UI collects the total scholarship amount), as well as the UI field for general "other income" because it won't be clear from the Get App response what is "other income" and what is scholarship

iii. When there is no current income object that says scholarship, then the UI can pre-populate any incomeSourceType=other_income amount on the "other income" UI field and assume that it's not scholarship income

5. How should the requestor update the API when the applicant's projected annual income **is NOT variable**?

   a. After collecting all of a household member's current income, the application should call the Update App API to return a calculated annual income amount which will be returned via the field computed.member.income.attestedAnnualizedAptcIndividualIncomeAmt. The application UI should present this amount to the consumer and ask whether they agree with the amount as a projection of their total income (minus deductions) for the coverage year. If the consumer agrees, then it means their current monthly income times 12 essentially equals their annual income, so we consider that to be the situation when the consumer's income is "not variable".

   b. The requestor should set the variableIncomeIndicator = false and unknownIncomeIndicator = false, in which case the back-end logic uses the computed.member.income.attestedAnnualizedAptcIndividualIncomeAmt as the member's projected annual income.

```
"members": {
          "123456789": {
                    "income": {
                              "annualTaxIncome": {
                                        "incomeAmount": 25000,
                                        "unknownIncomeIndicator": false,
                                        "variableIncomeIndicator": false
                              },
                    }
          }
}
```

Figure 33 Sample JSON when Annual Tax Income is Not Variable

6. How should the requestor update the API when the applicant's projected annual income ***is*** **variable and predictable**?

a. When the application filer attests that they do not agree with the calculated projected annual income, we need to ask them whether their income for the coverage year is “hard to predict”. If they say no, then we can consider their annual income to be “predictable”.

b. The requestor should set the variableIncomeIndicator = true and unknownIncomeIndicator = false and annualTaxIncome.incomeAmount <> null.

```
"members" : {
        "123456789" : {
                "income" : {
                        "annualTaxIncome" : {
                                "incomeAmount" : 25000,
                                "unknownIncomeIndicator" : false,
                                "variableIncomeIndicator" : true
                        },
                }
        }
}
```

Figure 34 Sample JSON when Annual Tax Income IS Variable

7. How should the requestor update the API when the applicant’s projected annual income ***is* variable and NOT predictable (i.e., estimated)**?

   a. When the application filer attests that they do not agree with the calculated projected annual income, we need to ask them whether their income for the coverage year is “hard to predict”. If they say yes, then we can consider their annual income to be not “predictable”. When that is the case, we still need to collect a projected annual income from the consumer—we call this an “estimated” income.

   b. When the UI collects an annual estimated income amount for a consumer with income that is hard to predict, the requestor needs to set this in the Update App request by creating an additional current income record with incomeSourceType = OTHER_INCOME where the estimatedForAptcIndicator = true and the amount is set to the attested estimated annual income.

```
"members": {
    "123456789": {
        "income": {
            "annualTaxIncome": {
                "unknownIncomeIndicator": true,
                "variableIncomeIndicator": true
            },
            "currentIncome": {
                "currentIncome1": {
                    "incomeAmount": 1300,
                    "estimatedForAptcIndicator": false,
                    "incomeSourceType": "JOB",
                    "incomeFrequencyType": "MONTHLY",
                    "jobIncome": {
                        "employerName": "Example Company"
                    }
                },

                "currentIncome2": {
                    "incomeAmount": 25000,
                    "estimatedForAptcIndicator": false,
                    "incomeSourceType": "JOB",
                    "incomeFrequencyType": "ANNUALLY",
                    "jobIncome": {
                        "employerName": "Example Company"
                    }
                },

                "currentIncome3": {
                    "incomeAmount": 50000,
                    "estimatedForAptcIndicator": true,
                    "incomeSourceType": "OTHER_INCOME",
                    "incomeFrequencyType": "ANNUALLY",
                    "jobIncome": {
                        "employerName": "Example Company"
                    }
                }
```

Figure 35 Sample JSON when Annual Tax Income is Variable and NOT Predictable (aggregate)

8. Is there a valid business scenario where variableIncomeIndicator = false and unknownIncomeIndicator = true?
    a. No, this is not a valid combination of attestations and may produce incorrect income eligibility results.

9. Does the requestor need to provide inputs to the API when a user attests to no or $0 income?
    a. The back-end logic defaults the member’s attested income amount to $0 if no currentIncome attestations are provided for the individual and variableIncomeIndicator = false. Therefore, if the applicant does not

attest to income in the UI, the requestor does not need to provide currentIncome for that member in the API.

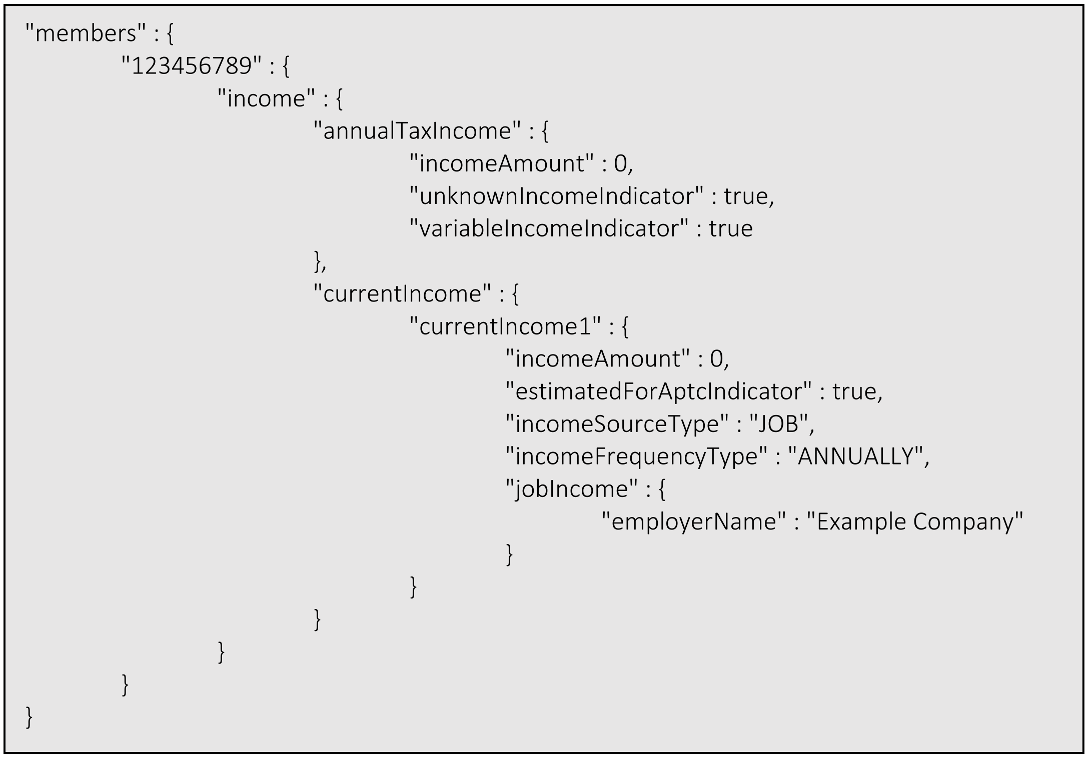
b. However, if the applicant attests to currentIncome <> null and variableIncomeIndicator = true, then the applicant must estimate their annual income for the coverage year. If the application filer attests that income is predictable and is $0 for the coverage year, then the requestor should provide annualTaxIncome.incomeAmount = 0. If the application filer attests that income is not predictable, the requestor should set unknownIncomeIndicator = true and provide the API a currentIncome record where the estimatedForAptcIndicator = true and incomeAmount = 0 to prevent a validation error.

```
"members": {
        "123456789": {
                "income": {
                        "annualTaxIncome": {
                                "incomeAmount": 0,
                                "unknownIncomeIndicator": false,
                                "variableIncomeIndicator": true
                        },
                        "currentIncome": {
                                "currentIncome1": {
                                        "incomeAmount": 1200,
                                        "estimatedForAptcIndicator": false,
                                        "incomeSourceType": "JOB",
                                        "incomeFrequencyType": "MONTHLY",
                                        "jobIncome": {
                                                "employerName": "Example Company"
                                        }
                                },
                                "currentIncome2": {
                                        "incomeAmount": 5000,
                                        "estimatedForAptcIndicator": false,
                                        "incomeSourceType": "JOB",
                                        "incomeFrequencyType": "ONE_TIME",
                                        "jobIncome": {
                                                "employerName": "Example Company"
                                        }
                                }


                        }
                }
        }
}
```

Figure 36 Income Member attesting to predictable variable income

Member 123456789 has attested a predicable variable income of $0 with a current income present



```
"members" : {
        "123456789" : {
                "income" : {
                        "annualTaxIncome" : {
                                "incomeAmount" : 0,
                                "unknownIncomeIndicator" : true,
                                "variableIncomeIndicator" : true
                        },
                        "currentIncome" : {
                                "currentIncome1" : {
                                        "incomeAmount" : 0,
                                        "estimatedForAptcIndicator" : true,
                                        "incomeSourceType" : "JOB",
                                        "incomeFrequencyType" : "ANNUALLY",
                                        "jobIncome" : {
                                                "employerName" : "Example Company"
                                        }
                                }
                        }
                }
        }
}
```

Figure 37 Income Attesting to non-predictable estimated income

Member 123456789 has attested their income in not predictable, thus the API is provided a current income record with a value of $0 that is estimated for APTC

10. How should the requester provide guidance when the application filer reports income totaling more than $1 million?
    a. The requester could display a message that instructs the consumer to go back and double check their individual income amounts and reduce any as needed, or change their application to non-FA.
11. How should tribal income be provided?
    a. The tribal income is the portion of an attested current income that is received from tribal sources. It must meet all the following conditions or a validation error will be returned.
        i. must be a positive numeric amount AND
        ii. must not exceed the total income amount AND
        iii. the incomeType must not be "SOCIAL_SECURITY_BENEFIT" or "UNEMPLOYMENT"

Responses:

The UI uses the API responses to prompt the following questions within the Income section of the application:

- Job Income Discrepancy page is needed when the applicant has attested current month job income which prompts an inconsistency that the back-end system determines could be potentially resolved by a reasonable explanation.
- Annual Income Discrepancy page is needed when the household has an annual income inconsistency which the back-end system determines can be resolved by a reasonable explanation.

1. When has the requestor collected enough data to receive the information about income inconsistencies?
    a. The Hub call to Equifax can occur after the following data has been collected:
        i. firstName <> null
        ii. lastName <> null
        iii. birthDate <> null
        iv. ssn <> null
        v. accountHolderIdentityProofedIndicator = true
        vi. requestingFinancialAssistanceIndicator = true
    b. The Hub call to IRS can occur after the following data has been collected:
        i. firstName <> null
        ii. lastName <> null
        iii. ssn <> null
        iv. accountHolderIdentityProofedIndicator = true
        v. requestingFinancialAssistanceIndicator = true
        vi. computed.taxHouseholds.taxHouseholdComposition <> null
    c. Current and annual income attestations and family/tax relationships have been collected for all household members and provided to the Update App API.

2. When should the UI display the Job Income Discrepancy page?
    a. The UI should display the Job Income Discrepancy page – before the Annual Income Discrepancy page – when the computed.members.income.employer.jobIncomeExplanationRequiredIndicator = true. Note the jobIncomeExplanationRequiredIndicator can be set for multiple employers under the same member.

```
{
        "result": {
                "computed": {
                        "members": {
                                "1250270556856992473": {
                                        "income": {
                                                "employer": [{
                                                                "employer": "ABC Corp.",
                                                                "jobIncomeExplanationRequiredIndicator": true
                                                        }
                                                ]
                                        }
                                }
                        }
                }
        }
}
```

Figure 38 Sample JSON result for when Job Income Explanation is required

b. Based on the applicant’s answer to the question in the UI, the requestor should set the attestations.members.income.currentIncome.jobIncome.incomeDifferenceReason where the attestations.members.income.currentIncome.jobIncome.employerName matches the computed.members.income.employer.employerName.

```
{

        "members": {
                "123456789" : {
                        "income": {
                                "currentIncome": {
                                        "currentIncome1": {
                                                "sequenceNumber": 1,
                                                "incomeAmount": 12000,
                                                "incomeSourceType": "JOB",
                                                "incomeFrequencyType": "ANNUALLY",
                                                "jobIncome": {
                                                        "incomeDifferenceReason": "STOP_WORKING",
                                                        "employerName": "ABC Corp."
                                                }
                                        }

                                }
                        }
                }
        }
}
```

Figure 39 Sample JSON request provided when Job Income Explanation is required

c. If the applicant selects “A reason not listed above,” the requestor should send:

  i. attestations.members.income.currentIncome.jobIncome.incomeDifferenceReason = “OTHER”

  ii. attestations.members.income.currentIncome.jobIncome.otherIncomeDifferenceReason = applicant’s free-text answer

```
{

        "members": {
                "123456789" : {
                        "income": {
                                "currentIncome": {
                                        "currentIncome1": {
                                                "sequenceNumber": 1,
                                                "incomeAmount": 12000,
                                                "incomeSourceType": "JOB",
                                                "incomeFrequencyType": "ANNUALLY",
                                                "jobIncome": {
                                                        "incomeDifferenceReason": "OTHER",
                                                        "employerName": "ABC Corp."
                                                        "otherIncomeDifferenceReason”: “”
                                                }
                                        }

                                }
                        }
                }
        }
```

Figure 40 Sample JSON for when applicant selects "A reason not listed above" for job income discrepancy

2. When should the UI display the Annual Income Discrepancy page?
    a. The UI should display the Annual Income Discrepancy page when at least one of the following conditions are true:
        i. Tax household-level discrepancy is flagged: computed.taxHouseholds[*tax household anchor* member ID*].annualIncome. incomeExplanationRequiredIndicator = true
            1. When this type of discrepancy is flagged, the backend flags whether the discrepancy is due to attested tax household income being higher or lower than the external data sources at the following path: computed.taxHouseholds[*tax household anchor member ID*].annualIncome.incomeExplanationRequiredReasonType = "INCOME_HIGHER_THAN_SOURCE" or "INCOME_LOWER_THAN_SOURCE"
            2. The tax household anchor member ID’s corresponding name should be displayed in the question. If this member has attestations.members[tax household anchor member ID].family.taxReturnFilingStatusType = “MARRIED_FILING_JOINTLY,” then their attested spouse’s name would also be included.
        ii. Member-level discrepancy is flagged: any member has computed.members[member ID 123].income.annualIncomeExplanationRequired = true
            1. There is no flag for whether attested income is higher vs. lower because member-level discrepancies are always set by the backend based on attested income being LOWER than the external data source.

2. The name of the tax filer(s) of the member with the discrepancy should be displayed in the question.
    a. Logic to identify tax filers: For given member ID "123," select *tax household anchor member ID* where computed.taxHouseholds[*tax household anchor member ID*].taxHouseholdComposition.taxHouseholdMemberIdentifiers includes "123."
3. Note that it is possible for there to be both tax household and member-level discrepancies flagged.

```
{
    "result": {
        "computed": {
            "members": {
                "593829573": {
                    “annualIncomeExplainationRequired”: true
                }
            },
            "taxHouseholds": {
                "123456789": {
                    "annualIncome": {
                        "incomeExplanationRequiredIndicator": true
                         "incomeExplanationRequiredReasonType":
"INCOME_LOWER_THAN_SOURCE"
                    }
                }
            }
        }
```

Figure 41 Sample JSON result for when tax household anchor has both tax household and member-level discrepancy

```
{
    "result": {
        "computed": {
            "taxHouseholds": {
                "123456789": {
                    "annualIncome": {
                        "incomeExplanationRequiredIndicator": true
                         "incomeExplanationRequiredReasonType":
"INCOME_HIGHER_THAN_SOURCE"
                    }
                }
            }
        }
    }
}
```

Figure 42 Sample JSON result for when only tax household-level discrepancy is flagged

```
{
        "result": {
                "computed": {
                        "members": {
                                "593829573": {
                                        “annualIncomeExplainationRequired”: true
                                }
                        }
                }
```

Figure 43 Sample JSON result for when only member-level discrepancy is flagged

b. The requestor should set the answer(s) about the reason(s) for the discrepancy to the question based on the type of discrepancy flagged. Note that multiple answers can be selected except for “OTHER.”

   i. If the tax household-level discrepancy is flagged, set members[*tax household anchor member ID*].income.annualTaxIncome.taxHouseholdIncomeDifferenceReasonType = selected answer(s).

      1. If the selected answer is not “OTHER,” then the requestor should also set members*[tax household anchor member ID*].income.annualTaxIncome.incomeLessExplainedIndicator = true

      2. If the selected answer is “OTHER,” then the requestor should set members[*tax household anchor member ID*].income.annualTaxIncome.incomeLessExplainedIndicator = false AND set members[*tax household anchor member ID*].income.annualTaxIncome. taxHouseholdIncomeDiscrepancyDescriptionText = consumer’s free-text answer.

```
{
        "members": {
                "123456789": {
                        "income": {
                                "annualTaxIncome": {
                                        "incomeAmount": 12000,
                                        "incomeLessExplainedIndicator": true,
                                        "unknownIncomeIndicator": false,
                                        "variableIncomeIndicator": false,
                                        "taxHouseholdIncomeDifferenceReasonType": [/*List of enums*/],
                                }
                        }
                }
        }
}
```

Figure 44 Sample JSON request for when tax household-level discrepancy is flagged and consumer selects reason(s) that are not "OTHER"

```
{
        "members": {
                "123456789": {
                        "income": {
                                "annualTaxIncome": {
                                        "incomeAmount": 12000,
                                        "incomeLessExplainedIndicator": false,
                                        "unknownIncomeIndicator": false,
                                        "variableIncomeIndicator": false,
                                        "taxHouseholdIncomeDifferenceReasonType": “OTHER”,
                                        "taxHouseholdIncomeDiscrepancyDescriptionText": "Free text"
                                }
                        }
                }
        }
```

Figure 45 Sample JSON request for when tax household-level discrepancy is flagged and consumer selects "OTHER"

ii. If a member-level discrepancy is flagged, then for ALL members flagged, set members[member ID].income.annualTaxIncome.taxHouseholdIncomeDifferenceReasonType = selected answer(s). For **each** member:

1. If the selected answer <u>is not</u> “OTHER,” then the requestor should set members[member ID].income.annualTaxIncome.incomeLessExplainedIndicator = true
2. If the selected answer is “OTHER,” then the requestor should set members[member ID].income.annualTaxIncome.incomeLessExplainedIndicator = false AND set members[member ID].income.annualTaxIncome. variableIncomeDescriptionText = consumer’s free-text answer.

```
{
        "members": {
                "123456789": {
                        "income": {
                                "annualTaxIncome": {
                                        "incomeAmount": 12000,
                                        "incomeLessExplainedIndicator": false,
                                        "unknownIncomeIndicator": false,
                                        "variableIncomeIndicator": false,
                                        "taxHouseholdIncomeDifferenceReasonType": “OTHER”,
                                        "variableIncomeDescriptionText ": "Free text"
                                }
                        }
                }
        }
```

Figure 46 Sample JSON request for when member-level discrepancy is flagged and consumer selects "OTHER"

```
{
        "members": {
                "123456789": {
                        "income": {
                                "annualTaxIncome": {
                                        "incomeAmount": 12000,
                                        "incomeLessExplainedIndicator": false,
                                        "unknownIncomeIndicator": false,
                                        "variableIncomeIndicator": false,
                                        "taxHouseholdIncomeDifferenceReasonType": “OTHER”,
                                        "taxHouseholdIncomeDiscrepancyDescriptionText": “Free text”
                                }
                        }
                },
                "593829573": {
                        "income": {
                                "annualTaxIncome": {
                                        "incomeAmount": 1000,
                                        "incomeLessExplainedIndicator": false,
                                        "unknownIncomeIndicator": false,
                                        "variableIncomeIndicator": false,
                                        "taxHouseholdIncomeDifferenceReasonType": “OTHER”,
                                        "variableIncomeDescriptionText ": "Free text"
                                }
                        }
```

Figure 47 Sample JSON request for when both tax household and member-level discrepancies are flagged and consumer selects "OTHER”

3. What should the UI do when an application filer attest to multiple incomes from the same employer and there is an income discrepancy?
    a. The system will only flag for income discrepancy (incomeDiscrepancyIndicator = Y) if the backend can match one for one employer name for the discrepancy with an external source. If there are multiple attested or external job records with the same employer name or if the frequency of the attested and external income records do not match, then set incomeDiscrepancyIndicator = X. Therefore, for this scenario your UI will not be expected to handle this situation as it is not applicable.

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on income:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	attestations.members.income	Input Matrix: Income	Phase 1 & 2 Questions, UI Questions, Requirements	Income	3.1 – Eligibility Event Dependencies 3.2 – Application Services 3.3.4 – Option for Consumer Not to Provide Information
Responses	computed.members.income and computed.taxHouseholds.annualIncome		Backend Responses for UI	Income	3.3.1 – Branch for Financial Assistance 3.3.2 – Dynamic Questions 3.3.3 – Backend Interaction
Validations		Input Matrix: Service & Flow Validations			2.5 – Application Operational Assistance 5.2.4 – Exception Handling
Clearing					3.3.5 – Backwards Navigation
Reference Data	N/A				N/A

API Interaction	N/A			Income	4 – Sample UI Interactions with Application Services 5.2.3 – Technical Assumptions

Preliminary Eligibility

Preliminary eligibility results allow the UI to prompt only the questions relevant to the program(s) for which each applicant may be eligible. After requestors have collected and sent to SES all the attestations collected from the preceding income section, a call should be made to the Update App API so that requestors receive each application member's initial preliminary eligibility results for Medicaid, CHIP and APTC.

The information collected in this section of program-specific questions is used to determine each applicant's final eligibility. The attestations collected may also change the result for applicants initially preliminarily eligible for APTC, Medicaid or CHIP to **ineligible**. After the program-specific questions, another call should be made to the Update App API after collecting to determine whether program eligibility has changed. If the Update App API initially included a response of preliminaryMedicaidStatus = Y or preliminaryChipStatus = Y, then a subsequent Update App call may change those indicators to N and change preliminaryAptcStatus for that applicant to Y. If so, the UI should then prompt APTC and QHP-specific questions.

Cross-Program Questions

There are several questions that are required program-specific questions for more than one program:

- For applicants that are preliminarily eligible for Medicaid, CHIP, or APTC, the UI must ask about current enrollment in health coverage. These questions are not applicable for members that are only preliminarily eligible for QHP and/or non-FA applications. The request structure in the API differs based on program eligibility; however, the UI may ask about current enrollment within one question.
- For applicants that are preliminarily eligible for Medicaid, Emergency Medicaid, or CHIP – and who attest to being American Indian/Alaskan Native (AI/AN) – the UI should ask whether the applicant is eligible for and receives Indian Health services.

Inputs

1. How does the request structure differ for the current health coverage enrollment attestations based on program eligibility?
    a. If the member is preliminarily eligible for **Medicaid or CHIP**, the requestor should use the *members.medicaid structure to* set the *insuranceMarketType* list based on the consumer's selection(s) and set the *enrolledInHealthCoverageIndicator* = true.
    b. If the member is preliminarily eligible for **APTC**, the requestor should use the *members.insuranceCoverage* structure.
    c. The requestor may set the fields described in (a) and (b) for each applicant who attests to current coverage. For example, the API will not return an error if the *members.insuranceCoverage* object is populated as well as the *enrolledinHealthCoverageIndicator* for an applicant who has prelimMedicaidStatus = Y. This is especially useful for applicants who switch from being preliminarily Medicaid/CHIP eligible to being preliminarily APTC eligible.

2. What enums should the requestor provide when an applicant selects current enrollment in Marketplace coverage or other private market non-group coverage?

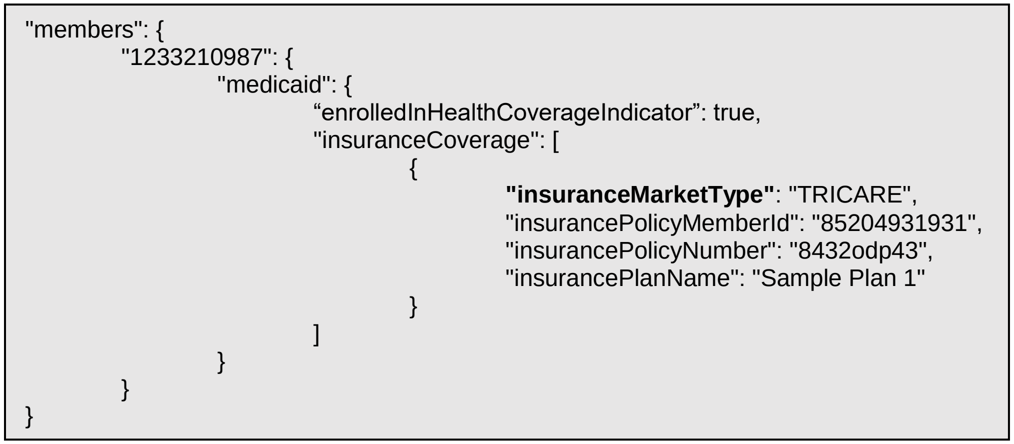
a. If the applicant is preliminarily eligible for **Medicaid, Emergency Medicaid, or CHIP**, the requestor should use the following *insuranceMarketType* enums in the *medicaid.insuranceCoverage* structure:
  i. OTHER_FULL_BENEFIT_COVERAGE
  ii. OTHER_LIMITED_BENEFIT_COVERAGE
  iii. MARKETPLACE_COVERAGE

b. If the applicant is preliminarily eligible for **APTC**, the requestor should select *members.insuranceCoverage.enrolledCoverages[index].insuranceMarketType* = INDIVIDUAL_INSURANCE if 2ai or 2aii are selected. If Marketplace coverage is attested, the requestor should select MARKETPLACE_COVERAGE.

3. What are the minimum required fields a requestor needs to provide if an applicant attests to current enrollment in public Minimum Essential Coverage (MEC)?

a. For applicants who attests to enrollment in **Medicare**, **Medicaid**, **CHIP**, **TRICARE**, **Peace Corps**, and/or **VA Health Program**:

i. And are preliminarily **Medicaid** or **CHIP** eligible, set *members.medicaid.insuranceCoverage[index].insuranceMarketType = MEDICARE, MEDICAID, CHIP, TRICARE, VETERAN_HEALTH_PROGRAM, and/or PEACE_CORPS* and set the *enrolledInHealthCoverageIndicator* = true.



```
"members": {
        "1233210987": {
                "medicaid": {
                        “enrolledInHealthCoverageIndicator”: true,
                        "insuranceCoverage": [
                                {
                                        "insuranceMarketType": "TRICARE",
                                        "insurancePolicyMemberId": "85204931931",
                                        "insurancePolicyNumber": "8432odp43",
                                        "insurancePlanName": "Sample Plan 1"
                                }
                        ]
                }
        }
}
```

Figure 48 Sample JSON for Preliminary Medicaid/CHIP applicants enrolled in coverage

ii. And are preliminarily **APTC** eligible, set *members.insuranceCoverage.enrolledCoverages[index].insuranceMarketType* = MEDICARE, MEDICAID, CHIP, TRICARE, VETERAN_HEALTH_PROGRAM, and/or PEACE_CORPS.

```
"members": {
        "1233210987": {
                "insuranceCoverage": {
                        "enrolledCoverages": [
                                {
                                        "insuranceMarketType": "MEDICARE",
                                        "insurancePolicyMemberId": "53202135943",
                                        "insurancePolicyNumber": "4923xdt24",
                                        "insurancePlanName": "Sample Plan 1"
                                }
                        ]
                }
        }
}
```

Figure 49 Sample JSON for Preliminary APTC applicants enrolled in non-employer sponsored coverage

b. For any applicant who attests to enrollment in **Employer Sponsored Coverage, Retiree**, and/or **COBRA** coverage, the following fields must be provided:
   i. If the applicant is preliminarily eligible for **Medicaid or CHIP and enrolled in ESC or Retiree**, set the following under *members.medicaid.insuranceCoverage:*
      1. *enrolledInHealthCoverageIndicator* = true
      2. *insuranceCoverage[index].insuranceMarketType* = EMPLOYER_SPONSORED
   ii. If the applicant is preliminarily eligible for **CHIP and enrolled in COBRA**, set the following under *members.medicaid.insuranceCoverage:*
      1. *enrolledInHealthCoverageIndicator* = true
      2. *insuranceCoverage[index].insuranceMarketType* = OTHER_LIMITED_BENEFIT_COVERAGE
      3. create an *escOffer**
         a. *escEnrolledIndicator* = true
         b. *employer.name* = *<consumer attestation>*
         c. *cobraAvailableIndicator* = true IF attests to being enrolled in COBRA
   iii. If the applicant is preliminarily eligible for **APTC**, create an *escOffer** object with the following:
      1. *escEnrolledIndicator* = true
      2. *employer.name* = *<consumer attestation>*

3. *cobraAvailableIndicator* = true IF attests to being enrolled in COBRA
4. *retireePlanCoverageIndicator* = true IF attests to being enrolled in Retiree coverage

```
"members": {
        "1233210987": {
                "insuranceCoverage": {
                        "offeredEmployeeCoverage": "YES",
                        "employerSponsoredCoverageOffers": {
                                "escOffer1": {
                                        "escEnrolledIndicator": true,
                                        "employer": {
                                                "name": "..."
                                        }
                                }
                        }
                }
        }
}
```

Figure 50 Sample JSON for Preliminary APTC applicants enrolled in employer-sponsored coverage

iv. When an applicant who was preliminarily eligible for APTC attests to current enrollment in one of the coverage types described in (a) or (b) , and if the attested types do not *only* include Medicaid or CHIP, then the UI need not ask about offers of employer-sponsored coverage (for more about that question sequence, see APTC specific questions section below).

4. When should the UI prompt questions on insurance policy information?
    a. Per the UI Question CG, the questions should be asked for applicants who are preliminaryMedicaidStatus or preliminaryChipStatus = Y and select current coverage in Medicare, TRICARE, VA Health Care Program, other limited or full benefit coverage, or other coverage. As outlined in the CG, the policy information asked depends on the coverage type selected.
    b. The insurance policy information is optional for consumers to provide.
    c. The questions do not need to be asked for applicants who are not preliminaryMedicaidStatus or preliminaryChipStatus = Y, but the system allows the attestations to be sent in the *members.insuranceCoverage* structure per 1c above.

5. How should requestors validate attested dates to account for SES validations and differences of time zones?
    a. The system converts untimed date attestations to the following times:

i. Beginning of Day: Attested Date at 00:00:00 Pacific Time

ii. End of Day: Attested Date at 23:59:59 Pacific Time

b. If an applicant attests to a valid date in a particular time zone it is possible that the system will throw a validation error when a date is before or after the current date.

i. if a user attests to a Birthdate of 1/1/2018 at 1/1/2018 00:31:01 EST (East Coast), the system will convert that date to 01/01/2018 at 00:00:00 PST (West Coast) and compare to the current time, which is 12/31/2017 21:31:01 PST (West Coast). According to the validation, the user has attested to a future birthdate.

ii. if a user attests to a future loss of MEC of 12/31/2017 at 12/31/2017 23:31:01 HST (Hawaii), the system will convert that date to 12/31/2017 at 23:59:59 PST (West Coast) and compare to the current time, which is 01/1/2018 02:31:01 PST (West Coast). According to the validation, the user has attested to a past loss of MEC.

c. To avoid these validation errors, requestors should have UI validations to compare attested dates based on Pacific Time.

i. To prevent future date but allow the current date the UI should convert the attested date to the beginning of the date at 00:00:00 Pacific Time and compare to current date time.

1. IF current dateTime < attested dateTime then throw a validation error
2. IF current dateTime > attested dateTime then do not throw a validation error

ii. To prevent future dates and not allow the current date the UI should convert the attested date to the end of the date at 23:59:59 Pacific Time and compare to current date time.

1. IF current dateTime < attested dateTime then throw a validation error
2. IF current dateTime > attested dateTime then do not throw a validation error

iii. To prevent past dates and allow current date the UI should convert the attested date to the beginning of the date at 00:00:00 Pacific Time and compare to current date time.

1. IF current dateTime > attested dateTime then throw a validation error
2. IF current dateTime < attested dateTime then do not throw a validation error

iv. To prevent past dates and not allow current date the UI should convert the attested date to the end of the date at 23:59:59 Pacific Time and compare to current date time.

1. IF current dateTime > attested dateTime then throw a validation error
2. IF current dateTime < attested dateTime then do not throw a validation error

Responses

1. When should the UI prompt the question for current enrollment in health coverage?
    a. Display these questions for all applicants, on an application requesting financial assistance, who are preliminarily eligible for Medicaid (preliminaryMedicaidStatus = Y) or preliminary eligible for Emergency Medicaid (preliminaryEmergencyMedicaidStatus = Y) or preliminary eligible for CHIP (preliminaryChipStatus = Y) or preliminarily eligible for APTC (preliminaryAptcStatus = Y).
    b. The questions are skipped for non-FA applications or applicants who are not preliminary eligible for Medicaid, Emergency Medicaid, CHIP, or APTC.
    c. See the *Inputs* section for more information on how the answers to these questions map to the API request based on the applicant's preliminary eligibility.

2. When should the UI prompt the questions related to American Indian/Alaska Native status?
    a. Display questions related to use and eligibility of Indian Health Services for all applicants with preliminaryMedicaidStatus = Y, preliminaryEmergencyMedicaidStatus = Y, or preliminaryChipStatus = Y and who attested to americanIndianAlaskanNativeIndicator = true. The answers to these questions map to the eligibleForItuIndicator and the receiveItuIndicator in the API request.
    b. Display questions related to membership in a federally recognized tribe for all applicants with preliminaryAptcStatus = Y and who attested to americanIndianAlaskanNativeIndicator = true. The answers to these questions map to personRecognizedTribeIndicator and federallyRecognizedTribeName in the API request.

3. When should the UI prompt the APTC and QHP-specific questions if the applicant is not initially eligible for APTC and QHP?
    a. Display these questions if the applicant is initially preliminaryMedicaidStatus = Y or preliminaryChipStatus = Y, but after asking the Medicaid or CHIP-specific questions, Update App API returns for the applicant (preliminaryMedicaidStatus = N or preliminaryChipStatus = N) and preliminaryAptcStatus = Y and/or preliminary QHP eligible (see QHP Specific Questions-Responses section for prelim QHP conditions).

APTC Specific Questions

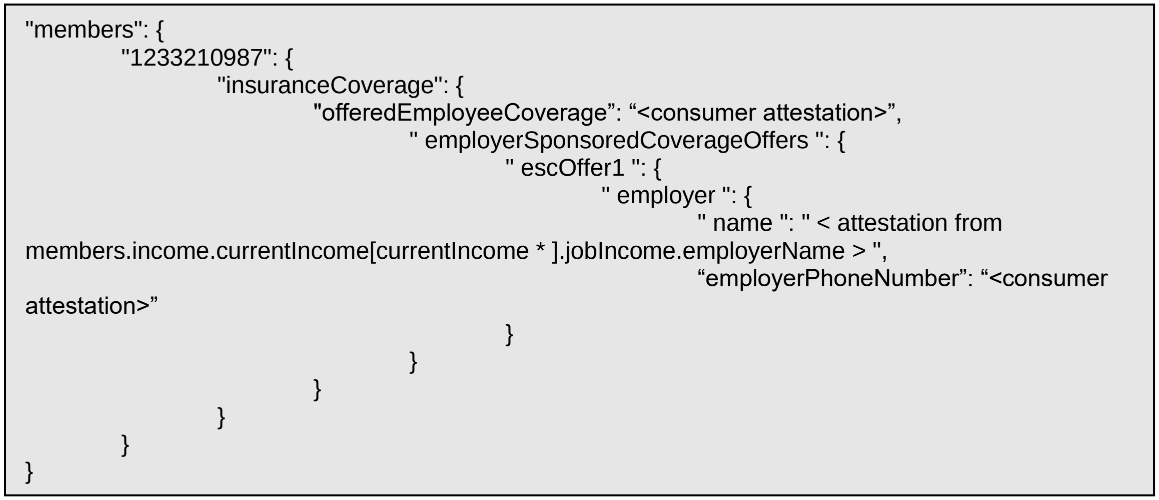
If the applicant indicates they are **not** currently enrolled in disqualifying health coverage for APTC and the API response indicates they are preliminary eligible for APTC, the UI should prompt additional questions to ask whether the applicant is **currently offered** health coverage through an employer. Please note, applicants who are preliminary APTC eligible should also be asked the ICHRA Affordability Questions in the UI. More information on the following can be found in "HRA Affordability Information" of the General HRA Information section below.

Inputs

1. What are the minimum required fields related to employer sponsored coverage that requestors need to provide?
   a. For any applicant who does not attest to enrollment in disqualifying coverage[2] for APTC eligibility, the UI must ask if the applicant is **currently offered coverage** by their employer or any other employers, collected in the income section where the income type was job, of consumers in the applicant's tax household. If the applicant attests to being offered coverage through an employer, the UI must ask additional questions to collect information about the coverage offer as well as the employer.
      i. *enrolledCoverages[index].insuranceMarketType* = MEDICAID, CHIP, or NONE
      ii. If applicant attests to having an offer of ESC, set *offeredEmployeeCoverage* = YES and create an *escOffer** object with the following fields:
         1. *employer.name*
         2. *employer.employerPhoneNumber*
         3. *escEnrolledIndicator* = false
         4. *employerOffersMinValuePlan*
         5. If applicant attests to an offer that meets the minimum value standard (*employerOffersMinValuePlan* = YES), the following fields must also be provided:
            a. *primaryInsuredMemberIdentifier*
            b. *lcsopPremium* and *lcsopPremiumFrequencyType*
      iii. Set *offeredEmployeeCoverage* = NO if applicant attests to not having an offer of coverage.

2. What employer contact information is required?
   a. The employer phone number is required for **all employers** when anyone in the tax household is eligible for APTC regardless of whether the applicant has an offer of coverage from the employer. Other employer contact information (as outlined in the UI question Companion Guide) should be requested for each employer, but is optional for the consumer to provide.
   b. At minimum, the requestor should create an *escOffer** with the following for each preliminarily APTC eligible applicant **per employer**:

[2] Disqualifying health coverage for APTC includes public and employer coverage, including: Employer Sponsored Coverage, Retiree, COBRA, Medicare, TRICARE, Peace Corps, and/or VA Health Program coverage. Note that while current enrollment in Medicaid or CHIP may ultimately disqualify an applicant from APTC eligibility as well, the back-end will apply additional logic to make that determination, so the UI should not consider applicants who only attest to enrollment in Medicaid or CHIP to be disqualified.



```
"members": {
        "1233210987": {
                "insuranceCoverage": {
                        "offeredEmployeeCoverage": "<consumer attestation>",
                                " employerSponsoredCoverageOffers ": {
                                        " escOffer1 ": {
                                                " employer ": {
                                                        " name ": " < attestation from
members.income.currentIncome[currentIncome * ].jobIncome.employerName > ",
                                                        "employerPhoneNumber": "<consumer
attestation>"
                                        }
                                }
                        }
                }
        }
}
```

Figure 51 Employer Contact Information required for applicants with an ESC Offer

3. Does the number proceeding the *escOffer** object need to follow a certain numbering scheme?
    a. No, it could be any combination of numbers as long as the number is unique.

4. How do requestors identify and send to the API the *primaryInsuredMemberIdentifier* for an offer of coverage?
    a. The UI should show the list of all members for the tax household the applicant belongs to, including non-applicants as well as the applicant themselves. Requestors should then map the selected name to the member identifier, as generated by the back-end, to send as the *primaryInsuredMemberIdentifier*. See the Household Composition Responses section above for retrieving information on tax households.

5. Is the *employerSponsoredCoverageOffers[escOffer*].employeeStatus* field required?
    a. No, this is not required for the input to the API. The reason the requestor may choose to ask about employee status (whether the applicant is a current or a former employee) is because the attestation may indicate whether the coverage offer is COBRA or retiree coverage, which is treated differently than other offers of ESC (it doesn't disqualify applicants from APTC as long as they are not currently enrolled).

6. Are attestations about future changes of a coverage offer or current enrollment through a job required?
    a. No, the UI can choose to not prompt any questions around future health coverage offers through job(s). **It is recommended that the UI focus only on current coverage offers to make the consumer's UI experience simpler**. The UI should include educational messaging to explain that consumers should return to report a life change (CiC application version) if their offer of coverage changes.
7. How should a requestor set indicators when the consumer is not enrolled in an ESC offer?
    a. For a given esc offer, if escEnrolledIndicator is false, planToEnrollEscIndicator and waitingPeriodIndicator need to be set. If planToEnrollEscIndicator and/or waitingPeriodIndicator is not collected from the consumer, set the value to false. Note this is only required until SES R12.1 deployment.

```
"members": {
        "1233210987": {
                "insuranceCoverage": {
                        "offeredEmployeeCoverage": "<consumer attestation>",
                                " employerSponsoredCoverageOffers ": {
                                        "escOffer": {
                                          "cobraAvailableIndicator": false,
                                        "lcsopPremium": 100,
                                         "lcsopPremiumFrequencyType": "WEEKLY",
                                        "lcsopName": "Name",
                                        "employerOffersMinValuePlan": "YES",
                                        "planToEnrollEscIndicator": false,
                                        "employer":
       {"name": "Pizza Hut",                  "employerPhoneNumber": "7038675309",  },

                                               "employeeStatus": "WORKING",
                                               "escEnrolledIndicator": false,
                                               "retireePlanCoverageIndicator": false,
                                               "waitingPeriodIndicator": false,
                                               "primaryInsuredMemberIdentifier": "2347128934"
                                        }
                                }
                        }
                }
        }
}
```

Figure 52 Example of attestation of ESC Offer

8. How should the requestor set ESC indicators in the API request when a member attests that the only ESC MEC they are offered is through the job of someone outside their tax household?
    a. The requestor should set employerOffersMinValuePlan = NO when ESC MEC is offered through the job of an employee who is not within the tax household of the APTC eligible applicant.

Responses

1. Is it possible for an applicant to change from preliminary APTC eligible to preliminary Medicaid/CHIP eligible after asking the APTC program-specific questions?
    a. No, this is not possible. An additional Update App API call is not needed immediately to check a preliminary APTC eligible applicant's eligibility results after collecting the attestations. Assuming an initial call was already made to check preliminary QHP eligibility, the UI would ask the QHP-specific questions next if applicable.

Medicaid Specific Questions

Below are the questions that only display when an applicant is preliminarily eligible for Medicaid:

- The **medical bills question** is asked to determine whether the applicant may be eligible for retroactive Medicaid coverage.
- The **covered dependent child question** is asked to determine whether a parent or caretaker that is preliminarily eligible for Medicaid has a non-applicant child that is receiving health coverage.
- The **absent parent question** is asked to check if the child has a parent living outside of the home when both the child and a parent or caretaker on the application are preliminarily eligible for Medicaid.
- The **deprived child question** is asked when the applicant lives in a state with a deprivation requirement and the applicant is the parent or caretaker of a child who lives with two parents.

Inputs

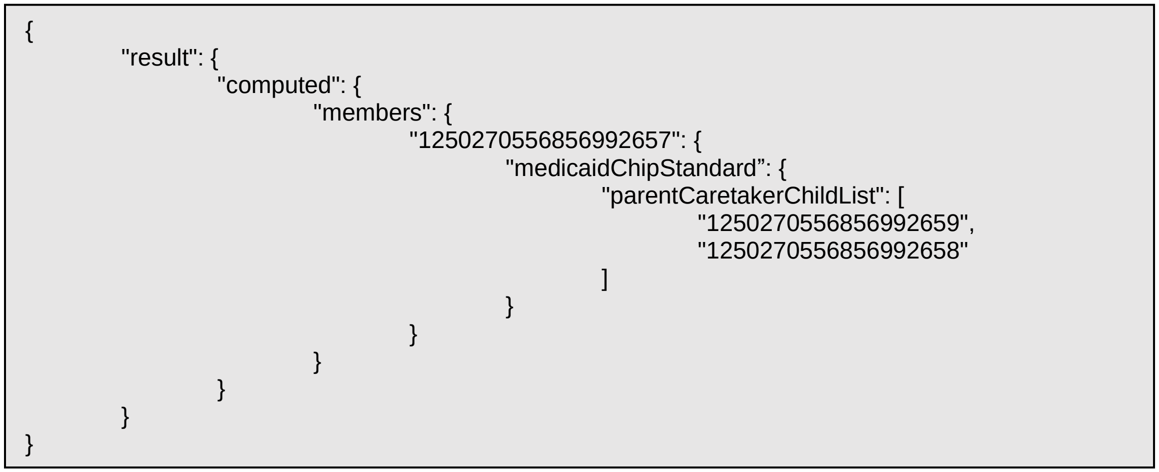
1. How should the requestor provide the answers to the Medicaid-specific questions in the API request?
    a. The answers to the Medicaid-specific questions in the UI are attestations located within the Medicaid object in the API.

Responses

1. When should the UI prompt the **medical bills question**?

   a. The medical bills question should display for all applicants where the preliminaryMedicaidStatus = YES or preliminaryEmergencyMedicaidStatus = YES.

2. When should the UI prompt the **covered dependent child question**?
   a. The covered dependent child question should display for all member where the preliminaryMedicaidStatus = YES and the dependentChildCoveredStatusReason = 670. This may happen on Phase 1, 2 or 3 applications.
   b. The list of children that should appear within the covered dependent child question depends on the child's age, attested caretaker status, and appropriate relationship based on state configuration data. Therefore, the API returns a *parentCaretakerChildList* with the list of children that should be populated by the UI within the covered dependent child question.The requester should only populate non-applicants (requestingCoverageIndicator = false) from this list



```
{
        "result": {
                "computed": {
                        "members": {
                                "1250270556856992657": {
                                        "medicaidChipStandard": {
                                                "parentCaretakerChildList": [
                                                        "1250270556856992659",
                                                        "1250270556856992658"
                                                ]
                                        }
                                }
                        }
                }
        }
}
```

Figure 53 Sample JSON of Parent Caretaker Child List

3. When should the UI prompt the **absent parent question**?
   a. The absent parent question should display if any parents or caretakers who are preliminarily eligible for Medicaid (any member with preliminaryMedicaidStatus = YES and parentCaretakerCategory = YES or T) has a child – in either the applicant's *parentCaretakerChildList* or where the caretakerRelativeIndicator = true in the familyRelationship between the parent and child – who attested to living with less than two parents.

4. When should the UI prompt the **deprived child question?**

- The deprived child question should display for all members where the requestingCoverageIndicator = true and preliminaryMedicaidStatus = Y and parentCaretakerCategoryStatus = Temporary and for all members in the *parentCaretakerChildList* with childCaretakerDeprivedStatus = Temporary. This may happen on Phase 1, 2 or 3 applications based on the state reference data API. See the DeprivationRequirementRetained indicator.

CHIP Specific Questions

Below are the questions that only display when an applicant is preliminarily eligible for CHIP:

- The **coverage through a job question** checks whether an applicant who is preliminarily eligible for CHIP and not pregnant had group health coverage that ended within the state's CHIP waiting period. This is relevant on Phase 1, 2 and 3 applications in selected states based on the state reference data API. See the StateCHIPWaitingPeriod indicator.
- The **offered state health benefits question** asks whether the applicant is offered or enrolled in health coverage for/through state employees when they are preliminarily eligible for CHIP, attests to being enrolled in employer sponsored coverage, and lives in a state that allows CHIP eligibility for applicants that receive state health benefits.. This would only be needed on a Phase 3 application based on the state reference data API. See the ChipForStateHealthBenefits indicator.

Inputs

1. How should the requestor provide the answers to the CHIP-specific questions in the API request?
    a. The answers to the CHIP-specific questions in the UI are attestations located within the CHIP object in the API.

Responses

1. When should the UI prompt the coverage ended through a job question?
    a. The coverage ended through a job question should display for all applicants where the chipWaitingPeriodStatusReason = 670.
        i. Within the question, the UI should display the number of months for the waiting period according to the stateCHIPWaitingPeriod state configuration.
    b. If the applicant attests to coverage ending, the UI should also ask about the reason why coverage ended.

2. When should the UI prompt the two different questions related to offers and and enrollment in state employee health benefits as part of the CHIP program-specific questions?

a. The question about whether a potentially CHIP eligible applicant is currently enrolled in state health benefits should display for all applicants where:
    i. (Prelim Chip eligible: <preliminaryChipStatus>=yes OR pregnant: <pregnancyIndicator>=True)AND
    ii. The application state doesn't block CHIP for state employees: < ChipForStateHealthBenefits = 03 AND
    iii. The applicant attested to being enrolled in Employer Sponsored Coverage <insuranceMarketType>="EMPLOYER_SPONSORED"

b. The question about whether a potentially CHIP eligible applicant is offered state health benefits should display for all applicants where the
    i. (Prelim Chip eligible: <preliminaryChipStatus>=yes OR pregnant: <pregnancyIndicator>=True)AND
    ii. Application state blocks CHIP for state employees and theirdependents: ChipForStateHealthBenefits = 01

QHP Specific Questions

The following QHP specific questions are applicable to both FA and non-FA applications for applicants who are preliminarily QHP eligible:

- Applicants who attest to having an AI/AN status should be prompted to answer whether they are part of a federally recognized tribe and the tribe name, if applicable, for the purposes of determining CSR and SEP eligibility.
- Applicants should also be prompted for past and/or future life changes to determine an applicant's SEP eligibility. These questions must be asked both during and outside of the annual Open Enrollment Period. The date of the change in circumstance is required for all change types, but note that the applicant would not be eligible for a SEP if the attested change date is more than 60 days in the past/future from the day of application submission.
    - Note that the adoption change in circumstance is the only type of change in which a **non**-applicant's attestation may allow the application to enroll in a SEP. Backend logic will grant the SEP to an adoptee's QHP-eligible parents if the non-applicant's attested date of adoption is within the last 60 days.

Please note, applicants who are preliminary QHP eligible should also be asked the HRA SEP Questions in the UI. More information on the following can be found in "HRA SEP Information" of the General HRA Information section below.

Inputs

1. What types of life changes could qualify an applicant for a SEP?
    a. Per the API documentation, an applicant can attest to any, all, or none of the following types:
        i. GAINING_LAWFUL_PRESENCE
        ii. RELEASE_FROM_INCARCERATION
        iii. LOSS_OF_MEC
            1. This is a loss in the past, including on the day of application update and/or submission.

iv. FUTURE_LOSS_OF_MEC
v. RELOCATION
vi. MARRIAGE
vii. ADOPTION

2. Should the UI questions for life changes be prompted for non-applicants?
    a. Yes, but only for the adoption SEP. All other life change types should only be prompted for applicants.

3. What should the requestor send if the applicant attests to losing MEC the day s/he is updating and/or submitting the application?
    a. The requestor should send this attestation as a **past** loss of MEC, using the following fields:

```
"members": {
        "1233210987": {
                "other": {
                        "changeInCircumstance"  {
                                "LOSS_OF_MEC" : {
                                        "changeDate": "...",
                                }
                        }
                }
        }
}
```

Figure 54 Sample JSON request for attesting to past loss of MEC

    b. Note that the issuer name is not required.

4. Are all the attestations around a marriage SEP required?
    a. They are conditionally required.
    b. If the applicant attests to being a member of a federally recognized tribe (attested *personRecognizedTribeIndicator* = true), then only the *changeDate* is required.

c. If the applicant does not attest to being a member of a federally recognized tribe (attested *personRecognizedTribeIndicator* = false), then the *coverage60DayBeforeMarriageIndicator* attestation is required.

d. If the applicant does not attest to being a member of a federally recognized tribe (attested *personRecognizedTribeIndicator* = false) and that s/he did not have coverage 60 days prior to getting married (attested *coverage60DayBeforeMarriageIndicator* = false), then the *liveInForeignCountry60DayBeforeMarriageIndicator* attestation is required.

5. Are all the attestations around a move/relocation change in circumstance required?
   a. They are conditionally required.
   b. If the applicant attests to having moved from a foreign country prior to moving (movedFromForeignCountryIndicator = true), the preceding address attestations and *coverage60DayBeforeMoveIndicator* are not required.
   c. If the applicant attests to being a member of a federally recognized tribe (attested *personRecognizedTribeIndicator* = true), then only the preceding address attestations are required.
   d. Otherwise, the *coverage60DayBeforeMoveIndicator* is also required.

6. Should the UI allow the consumer to select a date outside the eligibility window for a SEP?
   a. The UI should only allow consumers to select/enter a date within 60 days of the current date, including the current date.

Responses:

1. Is the question branching in the UI applicable for non-FA applications?
   a. The status indicators for preliminary eligibility in the API response are not applicable for non-FA applications. However, before displaying the SEP questions, the UI should check the following response from the API to determine whether the applicant is preliminarily eligible for QHP:
      i. preliminaryMedicaidStatus = NOT_APPLICABLE or NO and
      ii. preliminaryChipStatus = NOT_APPLICABLE or NO and
      iii. *Either* citizenshipStatus = YES *OR* qhpLawfulPresenceStatus = YES and
      iv. qhpResidencyStatus = YES and
      v. incarcerationStatus = NO

2. What is the recommended logic to pre-populate answers for consumer changes?

a. For re-enrollment and CiC applications, the requestor should call Get Application Details and pre-populate the members' attested changes for all types of life changes, **excluding an adoption/foster care/court-appointed guardian change**[3], that meet the following logic:
   i. Member has computed result for **eligibleSep** where **status** = Y, current date >= **endDate**, and **type** matches corresponding attestation of the life change.

b. The following table depicts the circumstance change attestations and the corresponding SEP type to each attested change type:

attestations.members.other	computed.memberseligibleSep.type
GAINING_LAWFUL_PRESENCE	LAWFUL_PRESENCE_SEP
RELEASE_FROM_INCARCERATION	RELEASE_FROM_INCARCERATION_SEP
LOSS_OF_MEC	LOSS_OF_MEC_SEP where currentDate >= startDate
FUTURE_LOSS_OF_MEC	LOSS_OF_MEC_SEP where currentDate < startDate
RELOCATION	MOVE_SEP
MARRIAGE	MARRIAGE_SEP
ADOPTION	ADOPTION_SEP

Table 1 Circumstance change attestation and corresponding SEP

3. What is the difference between eligibleSep and bestSep?
   a. An eligibleSep is a member level eligibility determination that states whether a member is eligible for a Special Enrollment Period by the status of "Y." A member may have zero to many eligibleSep determinations, which is based off the member's changeInCircumstance attestations and other criteria.
   b. The bestSep is an application level eligibility determination that states if an application has been granted a Special Enrollment Period by the status of "Y." An application will only have one bestSep determination, which is derived from the member level eligibleSep based on earliest startDate and other eligibility factors accounted for in the backend system logic.
   c. The bestSEP is used to populate relevant consumer messaging in the eligibility results page and eligibility determination notice.

[3] If the member who originally attested to being adopted is not SEP eligible and a QHP eligible parent is SEP eligible, UI pre-population may depict the incorrect member as having attested to adoption. However, the UI can prepopulate the fact that a consumer was adopted without specifying the name(s).

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on preliminary eligibility:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	Attestations.members.medicaid Attestations.members.chip Attestations.members.insuranceCoverage Attestations.members.other.changeInCircumstance	Preliminary Eligibility, Final Medicaid and CHIP, Attestation SEP	APTC & Medicaid & CHIP program questions - current coverage	Additional Questions	3.1 – Eligibility Event Dependencies 3.3.1 Branch for Financial Assistance Applications 3.3.4 – Option for Consumer Not to Provide Information
Responses	computed.members.[preliminaryMedicaidStatus, preliminaryChpStatus, preliminaryAPTCStatus, prelimniaryEmergencyMedicaidStatus, citizenshipStatus, qhpLawfulPresenceStatus, qhpResidencyStatus, incarcerationStatus] computed.members.medicaidChipStandard.childCaretakerDepreivedStatus computed.members.medicaidChipStandard.parentCaretakerChildList	N/A	Backend Responses for UI	Additional Questions	3.3.2 Dynamic Questions

Validations	N/A	Same as Inputs			2.5 – Application Operational Assistance 5.2.4 – Exception Handling
Clearing	N/A	N/A	N/A	N/A	3.3.5 – Backwards Navigation
Reference Data	[state].stateCHIPWaitingPeriod [state].deprivationRequirementRetained	N/A	N/A	N/A	3.4 Application State and System Configuration
API Interaction	N/A	N/A	N/A	Additional Questions	4 – Sample UI Interactions with Application Services 5.2.3 – Technical Assumptions

General HRA Information

1. What is an HRA?
    a. Health Reimbursement Arrangements (HRAs) are employer-funded group health plans from which employees are reimbursed tax-free for qualified medical expenses up to a fixed dollar amount per year. Unused amounts may be rolled over to be used in subsequent years. The employer funds and owns the arrangement. Health Reimbursement Arrangements are sometimes called Health Reimbursement Accounts.

2. What types of HRAs will the FFE ask consumers about?
    a. There are two types of HRAs which consumers can attest to on the Federally Facilitated Exchange (FFE), an individual coverage Health Reimbursement Arrangement (ICHRA) and a Qualified Small Employer Health Reimbursement Arrangement (QSEHRA).

3. What is an ICHRA?
    a. Employers can offer their employees an individual coverage Health Reimbursement Arrangement (ICHRA) instead of offering a traditional group health plan. This type of HRA is an alternative to traditional group health plan coverage to reimburse medical expenses, like monthly premiums and out-of-pocket costs such as copayments and deductibles.
    b. Individual coverage HRAs are a new type of HRA available to employers as of January 1, 2020. Therefore, consumers on the FFE should not be expected/allowed to attest to having an ICHRA offer or coverage prior to January 1, 2020.
    c. Generally, employers of any size can offer an individual coverage HRA, as long as they have one employee who isn't a self-employed owner or the spouse of a self-employed owner.
    d. If an employer offers their employee an individual coverage HRA and the employee accepts it, the employee (and their household members, if applicable) must enroll in individual health insurance coverage, or Medicare Parts A (Hospital Insurance) and B (Medical Insurance) or Part C (Medicare Advantage) that starts by the time the employee's individual coverage HRA begins. The employee can enroll in individual health insurance coverage through the Marketplace or through a private plan outside the Marketplace.

4. What is a QSEHRA?
    a. A Qualified Small Employer HRA is an alternative to traditional group health plan coverage which allows small employers to provide non-taxed reimbursement of certain health care expenses, like health insurance premiums and coinsurance, to employees who maintain minimum essential coverage, such as a plan from the Individual Marketplace. In many states, QSEHRAs allow small employers to provide their employees additional plan choices without managing group health plan coverage.
    b. Generally, employers can offer a Qualified Small Employer HRA if:
        i. They have fewer than 50 full-time employees
        ii. Provide the arrangement on the same terms to all full-time employees (reimbursement amounts may only vary based on age and the number of individuals covered)
        iii. Do not offer a group health plan, such as SHOP coverage, or a flexible spending arrangement (FSA)

5. Where on healthcare.gov can I find more information about ICHRAs and QSEHRAs?
    a. https://www.healthcare.gov/glossary/health-reimbursement-account-hra/
    b. https://www.healthcare.gov/ichra/

c. https://www.healthcare.gov/glossary/qsehra/
d. https://www.healthcare.gov/small-businesses/learn-more/individual-coverage-hra/
e. https://www.healthcare.gov/small-businesses/learn-more/qsehra/
f. https://www.healthcare.gov/qsehra/

Application Technical Sections

This section is intended to provide general guidance on the new HRA attestations and the corresponding API request fields. The UI should collect the Individual Coverage HRA (ICHRA) Affordability and HRA SEP attestations from different subsets of consumers based on their preliminary eligibility. The UI should only display the ICHRA Affordability questions to members that are preliminarily APTC-eligible, while the UI should display the HRA SEP attestations for all preliminarily QHP-eligible applicants.

Note: Only ICHRA Affordability functionality will be available to the consumer following SES R19.0. QSEHRA Employee Affordability functionality will be prioritized for a future release.

1. How does a requestor provide and differentiate the HRA SEP vs. HRA Affordability attestations in the API request?
    a. The requestor should provide the information for both an ICHRA offer for Affordability functionality and an ICHRA offer for SEP functionality using the *hraOffers* object. The requestor should send only one hraOffers object per member in the API request, which can have multiple arrays for different sets of attestations, as shown below.
    b. General guidance for differentiating between an ICHRA Affordability attestation and an ICHRA SEP attestation is based on the required inputs for each. All ICHRA offer Affordability attestations are expected to have an *employer.Name* and *primaryInsuredMemberNotInTaxHhIndicator*, which are both fields that are not collected for the HRA SEP attestations. Additionally, all ICHRA offer SEP attestations are expected to have a *noticeDate* and *enrolledInOfferFromSameEmployerIndicator*, which are both fields that are not collected for the ICHRA Affordability attestations. Note that the ICHRA SEP and ICHRA Affordability attestations should be sent as different arrays, even if the common fields (e.g. *hraType*, *startDate*) are identical for both sets of attestations.
    c. More information regarding required fields for the ICHRA Affordability and HRA SEP attestations are provided in the sections below.

```
{
    "members" : {
        "123456789" : {
            "insuranceCoverage" : {
                "enrolledInIchraIndicator": false,
                “offeredIchraIndicator”: true,
                “hraOffers”: [
                    {
                        “hraType”: “ICHRA”,
                        “startDate”: “2020-01-01”,
                        “noticeDate”: “2020-01-01”,
                        “enrolledInOfferFromSameEmployerIndicator”: true,
                        “enrolledInOfferFromSameEmployerUntilStartDateIndicator”: false
                    },
                    {
                        “hraType”: “QSEHRA”,
                        “startDate”: “2020-01-01”,
                        “noticeDate”: “2020-01-01”,
                        “enrolledInOfferFromSameEmployerIndicator”: true,
                        “enrolledInOfferFromSameEmployerUntilStartDateIndicator”: false
                    },
                    {
                        “hraType”: “ICHRA”,
                        “primaryInsuredMemberIdentifier”: “123456789”,
                        “startDate”: “2020-01-01”,
                        “endDate”: “2020-12-31”,
                        “employerSelfOnlyOfferAmount”: 4000,
                        “employeeSelfOnlyOfferFrequencyType”: “MONTHLY”,
                        “primaryInsuredMemberNotInTaxHhIndicator”: false,
                        “employer”: {
                            “name”: “ABC Corp”,
                            “employerIdentificationNumber”: “123456”,
                            “streetName1”: “123 Main St”,
                            “streetName2”: “Apt 100”,
                            “cityName”: “Wilmington”,
                            “stateCode”: “DE”,
                            “zipCode”: 12345”,
                            “plus4Code”: “1111”,
                            “countryCode”: “001”,
                            “countyName”: “Wilmington”,
                            “countyFipsCode”: “019832”,
                            “employerPhoneNumber”: “7038675309”,
                            “contact”: {
                                “name”: {
                                    “firstName”: “First Name”,
                                    “lastName” “Last Name”,
                                    “middleName”: Middle Name”,
                                    “suffix”: “Jr.”
                                },
                                “email”: “employer.contact@gmail.com”,
                                “phoneNumber”: “7038673509”
                            }
                        }
                    }
                ]
            }
        }
```

```
{
    "members" : {
        "123456789" : {
            "insuranceCoverage" : {
                “hraOffers”: [
                    {
                        “hraType”: “ICHRA”,
                        “startDate”: “2020-01-01”,
                        “noticeDate”: “2020-01-01”,
                        “enrolledInOfferFromSameEmployerIndicator”: true,
                        “enrolledInOfferFromSameEmployerUntilStartDateIndicator”: false
                    },
                    {
                        “hraType”: “QSEHRA”,
                        “startDate”: “2020-01-01”,
                        “noticeDate”: “2020-01-01”,
                        “enrolledInOfferFromSameEmployerIndicator”: false
                    }
                ]
            }
        }
    }
}
```

HRA Affordability Information

The guidance provided in this section is specific ICHRA Affordability, as this functionality will be available with SES R19.0. QSEHRA Affordability will be prioritized for a future release.

If the API response indicates a consumer is preliminarily eligible for APTC, the UI should prompt additional questions to ask whether the applicant is **currently enrolled in or offered** an Individual Coverage HRA through an employer. Note that unlike ESC, the UI should prompt the consumer for Individual Coverage HRA Affordability information even if the consumer attests to being currently covered by other health coverage. Additionally, affordability of an Individual Coverage HRA offer will not be determined until the requestor submits the application, so the UI should continue to collect attestations for ESC offer questions, if applicable, even if the consumer attests to an ICHRA offer.

1. What fields are required within the hraOffers object for Affordability-related attestations?
    a. For any applicant who is preliminarily eligible for APTC, the UI must ask if the applicant is **currently enrolled in** an Individual Coverage HRA or if they are **currently offered** an Individual Coverage HRA by their employer or any other employer collected for consumers in the income section where the income type was set equal to “Job” and within the applicant’s tax household. If the applicant attests to being currently covered by an ICHRA, the UI does not need to collect any further ICHRA affordability information (noting that Individual Coverage HRA SEP information would still need to be collected, if applicable), as the consumer will be disqualified from receiving APTC benefits. If the applicant attests to being offered coverage through an employer, the UI must ask additional questions to collect information about the offer.
        i. If the consumer attests to being covered by an Individual Coverage HRA and the consumer will still be covered beyond the next 60 days, set the *enrolledInIchraIndicator* = true; else set to false.

ii. If applicant is not currently enrolled in an Individual Coverage HRA (*enrolledInIchraIndicator* = false), the UI must ask if the applicant is offered an Individual Coverage HRA. If the consumer attests to being offered an Individual Coverage HRA, set *offeredIchraIndicator* = true and create an *hraOffers* object with the following minimally required fields:

i. *hraType* = "ICHRA"
ii. *employer.Name*
iii. *employer.contact.PhoneNumber*
iv. *primaryInsuredMemberNotInTaxHhIndicator*
v. If an applicant attests to having access to an offer by an employer where the employee is within the applicant's tax household (*primaryInsuredMemberNotInTaxHhIndicator* = false), the following fields must also be provided:

1. *primaryInsuredMemberIdentifier*
2. *startDate*
3. *endDate*

vi. If an applicant attests that their ICHRA offer is relevant to their eligibility determination (see Question #2), the following fields must also be provided:

1. *employeeSelfOnlyOfferAmount*
2. *employeeSelfOnlyOfferFrequencyType*

iii. Set *offeredIchraIndicator* = false if applicant does not attest to having an Individual Coverage HRA offer.

2. Would a requestor ever expect to set both *enrolledInIchraIndicator* = true and *offeredIchraIndicator* = true for a single consumer?

a. No. When the *enrolledInIchraIndicator* is set to true further ICHRA affordability information is not required, and the expectation is that the *offeredIchraIndicator* will be set to null.

3. Should Individual Coverage HRA affordability indicators be set for members who are not preliminarily APTC eligible?

a. No. If a consumer is **not** preliminarily eligible for APTC, which includes members who are preliminarily eligible for Medicaid or CHIP, who are not seeking coverage, or who are not requesting financial assistance, the requestor should set both *enrolledInIchraIndicator* and *offeredIchraIndicator* to null for that member.

4. What instances do not require the requestor to provide an *employeeSelfOnlyOfferAmount and employeeSelfOnlyOfferFrequencyType*?

a. A requestor does not need to provide an *employeeSelfOnlyOfferAmount and employeeSelfOnlyOfferFrequencyType* in the API request if the consumer's attested offer start and/or end dates disqualify the ICHRA offer from impacting the consumer's point in time eligibility determination on the consumer's application submission date. A consumer's ICHRA offer is only considered relevant for the consumer's eligibility determination if the consumer will still have *access* to the ICHRA (whether they will be covered by the ICHRA or will still have access to the ICHRA offer) beyond the next 60 days. Additionally, the consumer's ICHRA offer should start on the current date, before the current date, or within the next 60 days from current date; otherwise the UI should inform the consumer to return to their application at a later date to provide information on his/her ICHRA offer. Therefore, *employeeSelfOnlyOfferAmount* and *employeeSelfOnlyOfferFrequencyType* are not required if *startDate* is greater than 60 days from the date of application submission, or if endDate is 60 days or fewer from the date of application.

b. Note, the requestor can still provide this information even if the conditions stated above for *startDate* and *endDate* are not met. SES will evaluate the *startDate* and *endDate* and determine if the system will calculate affordability for the offer.

5. What employer contact information is required for Individual Coverage HRA attestations?
   a. Similar to the guidance provided for ESC, the *employer.contact.phoneNumber* field is required for all Individual Coverage HRA offer attestations. Other employer contact information (as outlined in the UI question Companion Guide) should be requested for each employer but is optional for the consumer to provide.

6. How do requestors identify the *primaryInsuredMemberIdentifier* for an Individual Coverage HRA offer?
   a. The UI should list all employers attested to in the Income section where income type is “Job,” including employers of non-applicants. Requestors should then map the selected employer to the associated employee and provide the member’s identifier as the *primaryInsuredMemberIdentifier* in the API request. It is recommended that the employer is clearly defined to a member on the UI to accurately handle scenarios where more than one applicant works for the same employer. Please refer to the Household Composition Responses section for retrieving information on tax households.

7. How should the requestor set Individual Coverage HRA indicators in the API request when a member attests that the only ICHRA they are offered is through the job of someone outside of their tax household?
   a. The requestor should set *primaryInsuredMemberNotInTaxHhIndicator* = true when the ICHRA offer available to a consumer is through the job of someone outside of their tax household. As mentioned in Question #1, this field is minimally required when the consumer attests to having access to an ICHRA offer.

8. When should the requestor provide an *employeeSelfOnlyOfferFrequencyType* of “PRORATED” in the API request?
   a. The UI requestor should provide an *employeeSelfOnlyOfferFrequencyType* = “PRORATED” when the consumer provides an *employeeSelfOnlyOfferAmount* that is the total amount available to them between the ICHRA’s attested start and end dates, rather than a weekly or monthly amount.

9. What is the recommended pre-pop guidance for HRA affordability attestations for CiC and re-enrollment scenarios?
   a. The UI requestor should pre-populate a consumer’s affordability attestations during CiC and re-enrollment scenarios. The requestor may choose not to pre-populate a consumer’s offer attestations if the end date of the previously attested ICHRA offer is fewer than 60 days from the current date.

```
{
    "members" : {
        "123456789" : {
            "insuranceCoverage" : {
                "enrolledInIchraIndicator": false,
                “offeredIchraIndicator”: true,
                “hraOffers”: [
                    {
                        “hraType”: “ICHRA”,
                        “primaryInsuredMemberIdentifier”: “123456789”,
                        “startDate”: “2020-01-01”,
                        “endDate”: “2020-12-31”,
                        “employerSelfOnlyOfferAmount”: 4000,
                        “employeeSelfOnlyOfferFrequencyType”: “MONTHLY”,
                        “primaryInsuredMemberNotInTaxHhIndicator”: false,
                        “employer”: {
                            “name”: “ABC Corp”,
                            “employerIdentificationNumber”: “123456”,
                            “streetName1”: “123 Main St”,
                            “streetName2”: “Apt 100”,
                            “cityName”: “Wilmington”,
                            “stateCode”: “DE”,
                            “zipCode”: 12345”,
                            “plus4Code”: “1111”,
                            “countryCode”: “001”,
                            “countyName”: “Wilmington”,
                            “countyFipsCode”: “019832”,
                            “employerPhoneNumber”: “7038675309”,
                            “contact”: {
                                “name”: {
                                    “firstName”: “First Name”,
                                    “lastName” “Last Name”,
                                    “middleName”: Middle Name”,
                                    “suffix”: “Jr.”
                                },
                                “email”: “employer.contact@gmail.com”,
                                “phoneNumber”: “7038673509”
                            }
                        }
                    }
                ]
            }
        }
    }
`
```

HRA SEP Information

The HRA SEP questions are applicable to both FA and non-FA applications for applicants who are preliminarily QHP-eligible. The HRA SEP functionality includes both ICHRA *and* QSEHRA SEP, which are available to the consumer following SES R19.0.

1. What fields are required to provide within the hraOffers object for SEP-related attestations?

   a. For each member that attests to being newly offered an ICHRA or newly provided a QSEHRA, where the ICHRA or QSERHA start date is within the past or future 60 days, the following attestations are required:
      i. *hraType* = either "ICHRA" or "QSEHRA"
         i. Note: based on guidance provided above, a consumer can only have one HRA SEP attestation per HRA type per member
      ii. *startDate*, which should always follow the "YYYY-MM-DD" format
         i. Note: when *hraType* = "ICHRA", the member's startDate must be set equal to or after 01/01/2020
      iii. *noticeDate*, which should always follow the "YYYY-MM-DD" format
      iv. *enrolledInOfferFromSameEmployerIndicator*, which should always follow a Boolean format
      v. If the *enrolledInOfferFromSameEmployerIndicator* = true, the enrolledInOfferFromSameEmployerUntilStartDateIndicator is also required and should always follow a Boolean format
   b. If a member does not attest to being newly offered an ICHRA or newly provided a QSEHRA, the requestor should not provide an hraOffers object with the required fields noted above for that member in the request.

2. What is the recommended pre-pop guidance for HRA SEP attestations for CiC and re-enrollment scenarios?
   a. The UI should pre-populate a consumer's Individual Coverage HRA SEP attestations and Qualified Small Employer HRA SEP attestations during CiC and re-enrollment scenarios. The requestor may choose not to pre-populate HRA SEP attestations if the start date of the previously attested offer is no longer within the prior 60 days from the current date, equal to the current date, or within the next 60 days from the current date.

```
{
        "members" : {
                "123456789" : {
                        "insuranceCoverage" : {
                                "hraOffers": [
                                        {
                                                "hraType": "ICHRA",
                                                "startDate": "2020-01-01",
                                                "noticeDate": "2020-01-01",
                                                "enrolledInOfferFromSameEmployerIndicator": true,
                                                "enrolledInOfferFromSameEmployerUntilStartDateIndicator": false
                                        },
                                        {
                                                "hraType": "QSEHRA",
                                                "startDate": "2020-01-01",
                                                "noticeDate": "2020-01-01",
                                                "enrolledInOfferFromSameEmployerIndicator": false
                                        }
                                ]
                        }
                }
        }
}
```

Sign and Submit

Before submitting the application, the consumer must provide a signature and the applicable legal attestations based on the preliminary eligibility results. If the consumer does not consent to the legal agreements, it may impact the final eligibility results.

Inputs:

1. How should the requestor provide the legal attestations in the API request?
   a. The legal attestations are only available in the Update App API at the application-level in the request. The requestor must enter the legal attestations through the Update App API before calling the Submit App API.

2. How should the requestor provide the signature in the API request?
   a. The signature name/text, type and date are available in both the Update App API and Submit App API. When calling the Submit App API, the requestor is always required to provide the signature and identity proofing indicator. If the Submit App API returns an error, the requestor should enter the signature through the Update App API, so that Medicaid/CHIP eligible applicants can receive retroactive coverage back to the lastConsumerSubmissionDate which may be different from the actual submissionDate.
   b. When the Submit App API is requested during the consumer flow, the requestor should provide the consumer's name in the signatureText field and pass "APPLICANT" in the signatureType field. If not the consumer flow, the requestor should provide the name of the Agent/Broker or Authorized Representative in the signatureText field and pass "AGENT_BROKER" or "AUTHORIZED_REPRESENTATIVE" in the signatureType field respectively.

Responses:

Below are the legal attestations that must be collected before submitting the application:

- The **renewal agreement indicator** and **renew eligibility year quantity** are asked on applications that request financial assistance.
- The **Medicaid agreement** is asked when any of the applicants are eligible for Medicaid.
- The **absent parent agreement** is asked when any child under 18 is eligible for Medicaid and attest to an absent parent.
- The **reconcile APTC indicator** is asked when any of the applicants are eligible for APTC. The answer to this question is optional.
- The **terminate coverage agreement** is asked when any of the applicants are eligible for QHP.
- The **penalty of perjury agreement** and **change in information agreement** are required for all applications. The **non-incarceration agreement** is also required for all applications; however, the requestor may derive and set this indicator in the API request after collecting the attested incarceration type from all the applicants, instead of displaying a separate legal agreement in the UI.

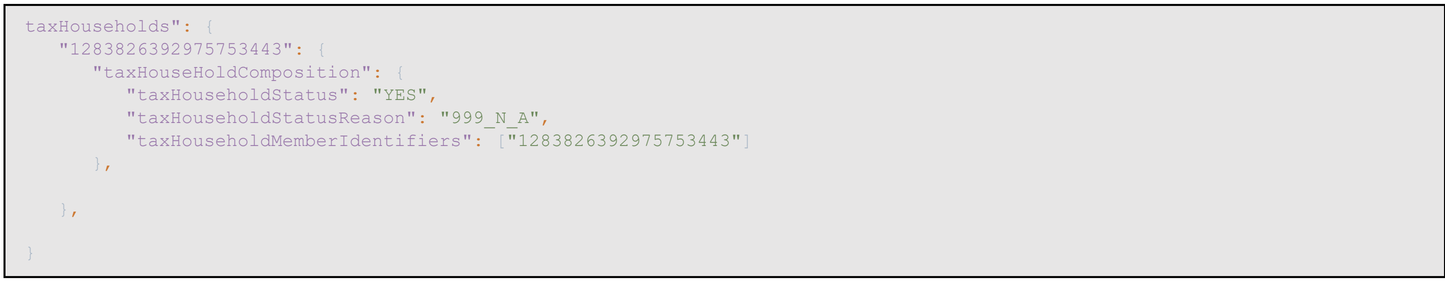
1. How should the renewal agreement indicator renewal eligibility year quantity should be set?

   a. Send true for renewal agreement indicator = true when the renewal eligibility year quantity is greater than 0
   b. Send false for renewal agreement indicator = false when the applicant denies renewal and set the renewal eligibility year quantity to 0

2. When should the UI prompt the **Medicaid agreement**?
   a. The Medicaid agreement should display when any member has a preliminaryMedicaidStatus = Yes OR (emergencyMedicaidStatus = Yes AND preliminaryMedicaidStatus = No).
   b. The requestor should provide the medicaidRequirementAgreementIndicator in the API request based on the applicant's answer to the Medicaid agreement.

3. When should the UI prompt the **absent parent agreement**?
   a. The absent parent agreement should display when any member under the age of 18 attested to absentParentIndicator = True and has a preliminaryMedicaidStatus = Yes.
   b. The requestor should provide the absentParentAgreementIndicator in the API request based on the applicant's answer to the absent parent agreement.

4. When should the UI prompt the **reconcile APTC indicator**?
   a. The reconcile APTC indicator should display when any member has a preliminaryAptcStatus = Yes. The UI should populate this question with all members with preliminaryAptcStatus = Yes that is a tax anchor.



```
taxHouseholds": {
   "1283826392975753443": {
      "taxHouseHoldComposition": {
         "taxHouseholdStatus": "YES",
         "taxHouseholdStatusReason": "999_N_A",
         "taxHouseholdMemberIdentifiers": ["1283826392975753443"]
      },

   },

}
```

Figure 55: Example JSON Response for a tax anchor

   b. The requestor should provide the reconcilePtcIndicatorType in the API request based on the applicant's answer to the reconcile PTC indicator.

5. When should the UI prompt the **terminate coverage agreement**?
   a. The terminate coverage agreement should display when any member meets the following criteria for preliminary QHP eligibility:
      i. preliminaryMedicaidStatus = No AND
      ii. preliminaryChipStatus = No AND
      iii. (citizenshipStatus = Yes OR qhpLawfulPresenceStatus = Yes) AND
      iv. qhpResidencyStatus = Yes AND
      v. incarcerationStatus = No

   b. The requestor should provide the terminateCoverageOtherMecFoundAgreementIndicator in the API request based on the applicant's answer to the absent parent agreement.

6. How should the UI handle when the Submit App API returns an error?
   a. If the Submit App API returns httpStatusCode <> 200 or resultType = ERROR, the UI should display a warning message to the consumer that collects their signature, and the requestor should use the Update App API to enter and capture the date of the signature.
   b. During times when the FFM is unable to contact the Trusted Data Sources (TDS) and all applicants were **not** pre-determined eligible for Medicaid or CHIP (detailed criteria below), EDE entities are permitted to continue collecting eligibility application data from consumers, along with any associated plan selection data. EDE entities are also permitted to provide consumers with estimated eligibility and enrollment results (premium, APTC, and CSR, etc.). Prior to providing consumers with estimated eligibility and enrollment results, EDE entities must collect an attestation from each consumer, where the consumer acknowledges that they are only being given an estimate, and that the plan selection will not be final unless the below criteria is met. EDE entities must also explain the enrollment process to the consumer, as outlined below. At a later time, EDE entities must call the Submit App service to complete the eligibility determination for the consumer's eligibility application, and if eligible based on the criteria outlined below, submit the consumer's plan selection data to HealthCare.gov using the EDE pathway.
      i. All of the below criteria must be met prior to providing the Consumer with estimated eligibility results:
         1. Submit App API returns httpStatusCode = 200 and errorCode = EDS_OR_HUB_DELAYED with error type = DATA_SOURCE_ERROR
         2. All Application members with requestingCoverageIndicator = True are also preliminaryMedicaidStatus = No and preliminaryChipStatus = No

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on sign and submit:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	TBD	TBD	TBD	TBD	TBD
Responses	TBD		TBD	TBD	TBD
Validations		TBD			TBD
Clearing					TBD
Reference Data	TBD				TBD
API Interaction	N/A			TBD	TBD

Eligibility Results

The eligibility result page provides the consumer with the final eligibility results of all the applicants and an electronic copy of the Eligibility Determination Notice (EDN). Applicants that were found ineligible for Medicaid and CHIP may also request a full Medicaid/CHIP determination from their state agency.

Inputs:

1. How does the requestor submit a full determination request through the API?
    a. The requestor should provide the list of member identifiers that are requesting a full determination in the array of requestMedicaidDeterminationMembers in the Submit App API. The Submit App API allows the requestor to submit a full determination request for a configurable number of days after the submission date (currently a max of 10 days). The list of requestMedicaidDeterminationMembers must include applicants with a medicaidStatus <> Yes.

Responses:

1. How does the requestor retrieve the final eligibility results?
    a. After submitting the application, the response in the Submit API contains the final eligibility results. The requestor may also call the Get App API to retrieve the latest data from the application.

2. Which reason codes in the API response indicate that a member's verification results are inconsistent?
    a. The reason codes in the attached spreadsheet indicate when a member's verification results are inconsistent:

       Inconsistency Reason Codes.xlsx

3. How does the requestor retrieve the EDN?
    a. After submitting the application using the Submit App API, requestors should call the DSRS API to retrieve the EDN to populate in the UI.

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on eligibility results:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	TBD	TBD	TBD	TBD	TBD
Responses	TBD		TBD	TBD	TBD
Validations		TBD			TBD
Clearing					TBD
Reference Data	TBD				TBD
API Interaction	N/A			TBD	TBD

Change in Circumstance

The consumer may return to the application to submit a Change in Circumstance (CIC). The requestor should always call the Get App API to check the latest data before updating an existing application. The requestor can also use the data returned by the Get App API to pre-populate the UI with the consumer's prior attestations – however, when developing the flow for CICs, the UI should consider how changes to upstream attestations may impact downstream inputs based on changes to the members program eligibility.

Inputs:

1. How can the requestor handle scenarios where the consumer changes an answer in the UI that invalidates downstream attestations that were previously entered/returned in the API?
   a. The requestor should cache data that is returned by the API while initiating a new version of an application, in order to retain a separate data set that can be used to pre-populate the UI with 'un-attested' answers that have not yet been entered or seen on the screen by the consumer. Then once the consumer views and confirms the previously 'un-attested' answer(s), the requestor can update and/or clear the attestations within the API to ensure an application is never submitted with 'un-attested' answers.

Household Composition Questions and Age Change Impacts

The following questions are impacted when the application member's age changes to under 19 or 19 and over (hard coded age):

The following questions are impacted when the application member's age changes to under or over the medicaidChildAgeThreshold:

i. Does [Applicant FNLNS] live with another parent and/or stepparent?
ii. Does [Applicant FNLNS] live with either of [his/her] parents and/or stepparents?
iii. Does [Applicant FNLNS] live with any other brothers or sisters who are under age [State Medicaid Age]?
iv. Does [Applicant FNLNS] live with brothers or sisters who are under age [State Medicaid Age]?
v. Who is brother or sister living with [Applicant FNLNS]?
vi. Does [FNLNS] live with [his/her] son, daughter, stepson, or stepdaughter?
vii. Does [FNLNS] live with any other sons, daughters, stepsons, or stepdaughters?
viii. Who is son, daughter, stepson, or stepdaughter living with [FNLNS]?
ix. Does [FNLNS] live with any other children under age 19?
x. Is [FNLNS] the main person taking care of this child (or children)?
xi. Tell us about this child?
xii. What's this child's relationship to [FNLNS]?
xiii. Do any of these children live with more than one parent, through birth or adoption?
xiv. About how many hours per week does each parent work?

The following questions are impacted when the application member's age changes to under or over the marriageQuestionAgeCutoff:

xv. Is [FNLNS] married?
xvi. Who is [FNLNS]'s spouse?

xvii. Enter [FNLNS]'s spouse's information.
xviii. Does [FNLNS] live with [FNLNS's spouse]?

The following questions are impacted when the application member’s age changes to under or over the medicaidChildAgeStudentThreshold:

xix. Are any of these people full time students?
xx. Does [Applicant name selected above] have a parent living in the same state where [Applicant name] goes to school?

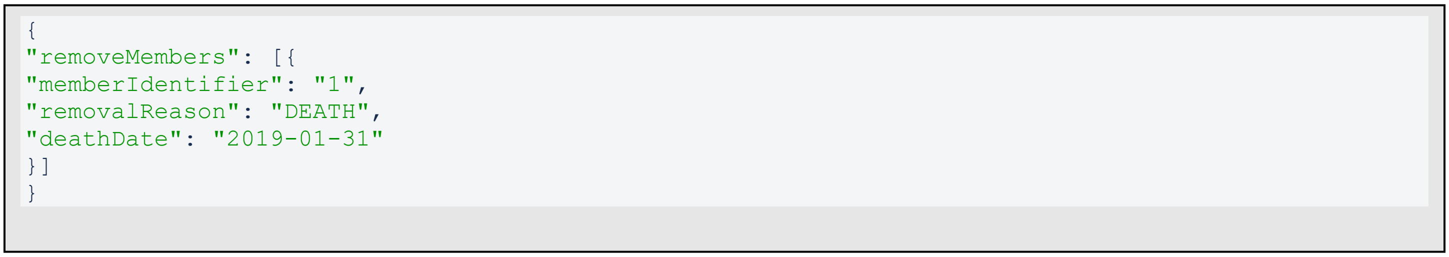
The following questions are impacted when the application member’s is female and age changes to within the ages of 9-66 or outside the ages of 9-66 (HardCoded):

xxi. Are any of these people pregnant?
xxii. How many babies is [FNLNS] expecting during this pregnancy?

The following questions are impacted when the application member’s age changes to within the ages of 18-25 or outside the ages of 18-25 (HardCoded):

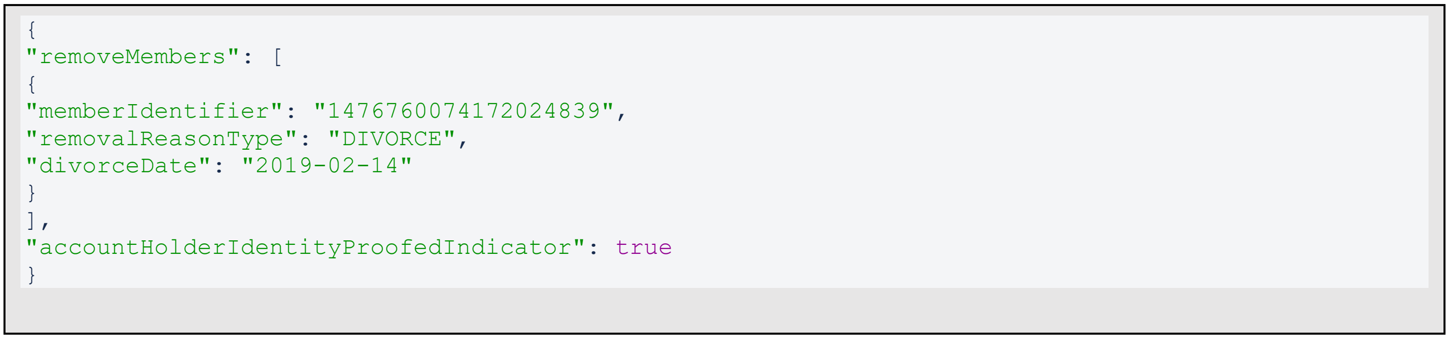
xxiii. Were any of these people ever in foster care?
xxiv. In what state was [Applicant name] in the foster care system?
xxv. Was [Applicant name] getting health care through [Name of state Medicaid program](Medicaid)?
xxvi. How old was [Applicant name] when [he/she] left the foster care system?

3. How can a requester remove an application member from the application and retain removal date and reason?
   a. There are couple of scenarios of where a requester would remove an application member
      i. Remove member who is a non-relevant non-applicant (as defined in 7.b.ii of the API Services section page X). In this case the requester should call the Remove Member API with the member ID of the member to be removed. This process will completely remove the member from the application without retaining any information.
      ii. Remove a deceased or divorced member. In this case the requester should call Remove Member with the member ID, date of death or date of divorce, and reason for removal of the member to be removed. This process will move the removed the member from the member segment to the removed member segment of the application application while retaining the name and DOB. On CiC, this member will be completely removed from the application.



```
{
"removeMembers": [{
"memberIdentifier": "1",
"removalReason": "DEATH",
"deathDate": "2019-01-31"
}]
}
```

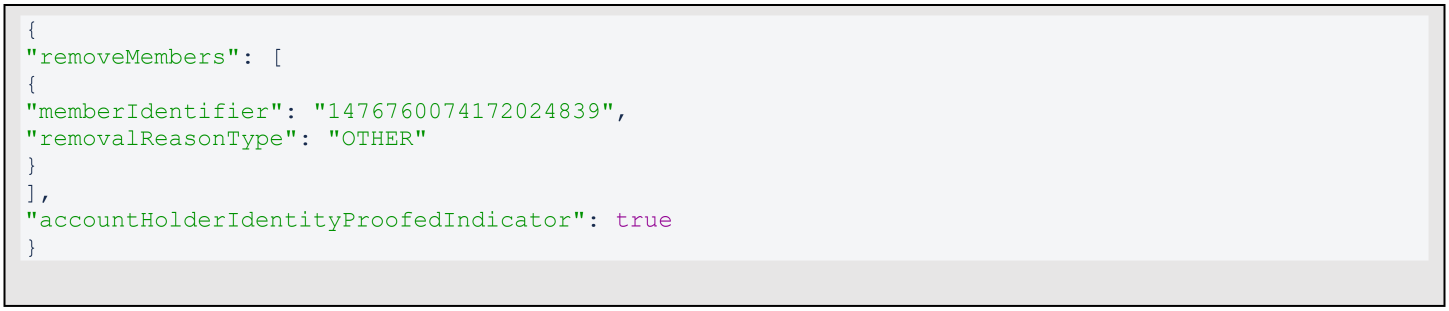
Figure 56: Example JSON Request for removing a deceased member



```
{
"removeMembers": [
{
"memberIdentifier": "1476760074172024839",
"removalReasonType": "DIVORCE",
"divorceDate": "2019-02-14"
}
],
"accountHolderIdentityProofedIndicator": true
}
```

Figure 57: Example JSON Request for removing a divorced member

iii. Remove a member due to reasons other than death or divorce. In this case the requester should call Remove Member with the member ID and reason for removal of the member to be removed. This process will move the removed the member from the member segment to the removed member segment of the application application while retaining the name and DOB. On CiC, this member will be completely removed from the application.



```
{
"removeMembers": [
{
"memberIdentifier": "1476760074172024839",
"removalReasonType": "OTHER"
}
],
"accountHolderIdentityProofedIndicator": true
}
```

Figure 58: Example JSON Response for removing a member due to other reasons

Responses:

1. How should the UI retrieve prior income attestations from the API response?
   a. When pre-populating a consumer’s prior attestations for current income, the requestor should check the esimatedForAptcIndicator to determine whether the currentIncome record is an estimated annual amount that should not be pre-populated with the other currentIncome records where the estimatedForAptcIndicator <> true.
   b. The UI should display **deductions** as a positive value and only display a negative amount if the income resulted in a **loss**, whereas requestors must provide a negative incomeAmount in the API for both **deductions** and income that resulted in **loss**. During both initial applications and CICs, the requestor should use the incomeSourceType to identify whether the currentIncome record is a deduction and convert incomeAmount to a *negative* value in the **SES API** or a *positive* value in the **UI**. Requestors should NOT convert the currentIncome.incomeAmount when the incomeSourceType is NOT a deduction.

Additional Documentation:

In addition to this FAQ, provided below is where requestors can find further documentation on CICs:

	APIary	Application Services Companion Guide	UI Question Companion Guide	App 3.0 Section Flows	UI Application Principles Document
Inputs	TBD	TBD	TBD	TBD	TBD
Responses	TBD		TBD	TBD	TBD
Validations		TBD			TBD
Clearing					TBD
Reference Data	TBD				TBD
API Interaction	N/A			TBD	TBD

General Questions

Access to Resources

How do I get access to the issuer or web-broker community on CMS zONE?

You first need to register and establish an account on the CMS Web Portal (if you don't already have one), then request access on zONE and join the issuer or web broker community.

To create an account on the CMS Web Portal:

- Go to https://portal.cms.gov/wps/portal/unauthportal/selfservice/newuserregistration/. Select "zONE: Opportunity to Network and Engage" in the "choose your application" field (zONE is typically at the bottom of the field). Agree to the CMS terms & conditions.
- On the next page, enter your personal information, create a user ID and password. You will have to select three security questions.

To get access to zONE:

- Log in to the CMS Enterprise Portal at https://portal.cms.gov/wps/portal/unauthportal/selfservic. Note: if you've just established your account, it may take a few minutes to be set up in the CMS Web Portal.
- On the first screen, click "request/add apps." CMS apps should display. Find zONE (typically near the bottom of the page) and click "request access"
- On the next page, select "zONE end user" and either issuer or web broker, whichever applies to you. Fill in the requested information and hit submit. Your request will be approved as soon as possible.

To get access to web broker/issuer community within zONE:

- Navigate to https://zone.cms.gov and log in using your CMS Web Portal account.
- Click "communities" in the top navigation.
- If you are an issuer, select "Issuer Community - Private" and click "join." If you are a web broker, select "Web Broker Community" and click "join." If you can't find them on the "communities" page, you can also search for the community by its full name in the search box at the top of the page. Your request will be approved as soon as possible.

How do I get access to CMS's daily issuer FFM update blast email message?

To be added to the distribution list, send an email to CMS_Issuer_Communications@cms.hhs.gov with the subject line: "Add POC to distribution list"

How do I get access to CMS's weekly Tuesday afternoon issuer technical call?

CMS holds a weekly issuer technical call every Tuesday from 3-4:30 p.m. Eastern Time. To access the webinar, visit: https://webinar.cms.hhs.gov/issuertechnicalworkgroup/. The dial-in information for the call is:
Phone: 1-877-267-1577
Meeting ID: 2284-2124

API Questions

For additional questions related to API onboarding/connectivity/RIDP/FARs, email dsh.support@qssinc.com with a subject line "EDE: API Question for [Insert Partner Name] Regarding [Subject of Question]." If applicable, copy your DEPOC. Technical questions related to APIs should be sent to the CMS_FEPS@cms.hhs.gov email help desk with a subject line "EDE: Technical Question for [Insert Partner Name] Regarding [Subject of Question]." If applicable, copy your DEPOC.

Where should we send a question related to API technical issues and not to API onboarding/connectivity/RIDP/FARs?

Questions related to APIs that are technical in nature should be sent to the CMS_FEPS@cms.hhs.gov email help desk with a subject line "EDE: Technical Question for [Insert Partner Name] Regarding [Subject of Question]" for the fastest response. If applicable, copy your DEPOC.

Where can I find information pertaining to the different APIs offered for Enhanced Direct Enrollment?

There is a basic overview of the APIs offered for Enhanced Direct Enrollment in Section 1 (Introduction) of the EDE Companion Guide. You can also find descriptions of each API along with request and response schemas in the Currently Available API Specs zip folder on zONE. The latest versions of these resources are available on zONE: https://zone.cms.gov/document/enhanced-direct-enrollment-ede-documents-and-materials

Testing Questions

For additional technical questions email CMS_FEPS@cms.hhs.gov with the subject line "EDE: Technical Question for [Insert Partner Name] Regarding [Subject of Question]." If applicable, copy your DEPOC.

When are the IMP1A or IMPL2 test environments available?

Generally speaking, the IMP1A and IMPL2 environments should be available Monday-Thursday until 8 p.m. Eastern Time, and Fridays until 5 p.m. Eastern Time. CMS will use the CMS daily issuer FFM update blast email message to notify partners of outages in either environment. For instructions on subscribing to the CMS daily issuer FFM update blast email message refer to the "General Questions" section of this FAQ document.

If IMP1A testing environment is down, does that mean the IMPL2 testing environment is also down?

The IMP1A environment and IMPL2 environment do not necessarily have outages at the same time. To be notified of outages in either environment, partners should subscribe to the CMS daily issuer FFM update blast email message. For instructions on how to subscribe to this email blast message refer to the "General Questions" section of this FAQ document.

If a partner completed testing for the RIDP/FARS API as part of our approval for DE Proxy for PY 2018, does it have to test again for Enhanced Direct Enrollment?

If a partner already has production access to these services, due to their participation in proxy direct enrollment, they should not need to complete any additional testing for these services.

Appendix

Appendix 1: Change Log

Author	Version	Date	Summary of Changes	Section	Page
AFS	1.0	05/16/18	Initial draft	Introduction Audience SES Documentation API Services Applications Sections, including: • Household Contact Information • Income General Questions	All
AFS	1.1	06/13/18	Added content	Application Sections, including: • Household Composition • Preliminary Eligibility • Sign and Submit • Eligibility Results • Change in Circumstance Appendix	All
AFS	1.2	07/03/18	Added Content	Added Application Sections, including: • More about this household • Citizenship and Immigration Updated Application Sections, including: • All sections with sample JSON • Income • Household Composition	All

Author	Version	Date	Summary of Changes	Section	Page
AFS	2.0	09/07/18	Added Content	Updated the following sections: • API Services ○ Added high-level scenarios for conflicting attestations ○ Added how to prepopulate applications from 2 years prior • Household Contact Information ○ Added Phase 1 and 2 clearing logic for Phase 3 applications • Household Composition ○ Added how to read relationships in the API response • More about this household ○ Added how to account for time zone differences ○ Added how to provide a Loss of MEC SEP on the day of submission • Sign and Submit ○ Added how to provide signature • Change in Circumstance ○ Removed how data from prior applications can cause UI errors ○ Added how to pre-populate in the UI and clear from the API un-attested answers	6, 9, 17, 38, 69-70, 77-78, 82, 88
AFS	2.0	10/16/18	Updated Content	Updated the following sections: • Citizenship and Immigration ○ Updated Enums for 189 to correctly reflect code	107
AFS	3.0	04/03/19	Updated Content	N/A	N/A

Author	Version	Date	Summary of Changes	Section	Page
AFS	4.0	07/01/2019	Added/Updated Content	1. Added clarity for remove member (page 4) 2. Added item #7 API services (page 10) 3. Corrected item #2 for Household Compostion (page 22) 4. Added item #7 for attestedHeadOfHousheoldIndicator guidance (page 24) 5. Clarified items #9 and #10 question for Household Compostion (page 26) 6. Corrected medicaidFamilyNotProvidedIndicator guidance hhcomp (page 31) 7. Added guidance for non-relevant non-applicants, item #1 ousehold Compostion (page 37) 8. Added guidance for new veteran fields item #9 Citizenship and Immigration Section (page 43) 9. Additional guidance for collecting address attestations Household Compostion item #8 (page 53) 10. Update sample JSONs for Income section (pages 59, 60 and 62) 11. Added guidance for tribal income item #11 Income section (page 63)	See Previous Column
AFS	5.0	09/10/2019	Updated document for Sept 2019	1. Remove guidance on when to prompt questions for Steps 1a/1b 2. Update CHIP Specific Questions section (p.87) 3. Improve the income section to clarify the SES calculated annual income amount must be displayed (p.59) 4. Add guidance on SEPs (eligible vs best) (p.91) 5. Add guidance for State and System Reference Data TTL (p.10) 6. FAQ updates for New Medicaid Block fields (p.53) 7. Add guidance to remove non-relevant non-applicants from the application (p.10) 8. Add guidance for employerOffersMinValuePlan (p.85) 9. Remove member instructions (p101-102) 10. Modify current veteran indicator guidance to include using the new veteran indicators (p.44)	See Previous Column

Appendix 2: Glossary

Term	Definition
Applicant	Application member who is applying for coverage
Coverage Year	Refers to the year which consumers are applying for coverage. If application is 2015 coverage year application, January 1 of the coverage year implies Jan 1 of 2015.
Qualified Health Plan (QHP)	An insurance plan that's certified by the Health Insurance Marketplace, provides essential health benefits, follows established limits on cost-sharing (like deductibles, copayments, and out-of-pocket maximum amounts), and meets other requirements under the Affordable Care Act. All qualified health plans meet the Affordable Care Act requirement for having health coverage, known as "minimum essential coverage."
Max APTC	The maximum amount of APTC a consumer can receive in a month.
Employer-Sponsored Coverage (ESI)	Health coverage offered to a consumer by an employer, regardless of whether the coverage is affordable or meets minimum value requirements.
Employer-Sponsored Coverage, Minimum Essential Coverage (ESI MEC)	Health coverage offered to a consumer by an employer that is both affordable and meets minimum value requirements
Non-Employer-Sponsored Insurance, Minimum Essential Coverage (non-ESI MEC)	Any health coverage offered to a consumer from an entity other than an employer that meets minimum essential coverage requirements.
Medicaid Child Age	A child under 19 or a full-time student aged 20-21 in select states per a "Y" for the "under21AndFTSOption"

Appendix 3: Acronyms and Abbreviations

Acronym / Abbreviation	Literal Translation
APTC	Advance Payments of the Premium Tax Credit
CCIIO	Center for Consumer Information and Insurance Oversight
CIC	Change In Circumstance
CHIP	Children's Health Insurance Program
CMCS	Center for Medicaid and CHIP Services
CMS	Centers for Medicare & Medicaid Services
CSR	Cost Sharing Reductions
CU	Change Utility
DE	Direct Enrollment
DMI	Data Matching Issue
EDS	External Data Source
EFT	Electronic File Transport
ES	Eligibility Support
ESC	Employer Sponsored Coverage
ESD	Eligibility Support Desktop
ESW	Eligibility Support Worker
FDD	Functional Design Document
FFE	Federally Facilitated Exchange
FTI	Federal Tax Information
FTR	Failure to Reconcile
HUB	Federal Services Data Hub
IPP	Insurance Plan Policy
IRS	Internal Revenue Service
MEC	Minimum Essential Coverage
OC	Office of Communications
OTS	Office of Technology Services
PDM	Periodic Data Match
PTN	Person Tracking Number
QHP	Qualified Health Plan
RPC	Reasonably Predictable Change
RRV	Redetermination and Renewal Verification
RTM	Requirement Traceability Matrix
SES	Standalone Eligibility Service
SNE	Submitted Not Enrolled

Acronym / Abbreviation	Literal Translation
SSA	Social Security Administration
SSN	Social Security Number
UI	User Interface

Appendix 4: Legacy System Differences

Below are differences between the guidance provided in this FAQ document and behaviors of the Legacy systems.

1. APTC Specific Questions: Employer Sponsored Coverage offers
    a. The guidance provided in this document recommends focusing only on current Employer Sponsored Coverage offers, rather than also inquiring about future changes to offers. The Legacy system asks about future changes to offers, such as whether the employer will make changes to the offer or whether the employee will enroll in an offer of coverage or disenroll from their current coverage.
    b. The Legacy system offers the option of "I don't know" for the following situations about current ESC offers. However, it is recommended that the UI only offers choices of "Yes" or "No" with consumer help text on how to find the information because an attestation of "I don't know" automatically makes the applicant ineligible for APTC:
        i. Whether the applicant is currently offered ESC
        ii. Whether the current offer of ESC meets the minimum value
        iii. The lowest cost self-only plan premium amount

Appendix 5: Immigration Status Reference Table

The Immigration Status Reference Table below is based on *Table 47 – Document Types* in the Verify Lawful Presence (VLP) v33 Business Service Definition (BSD) for the Federal Data Services Hub (Federal DSH). R = Required and O = Optional. Note: If Notice of Action (I-797) is selected in the UI, it's sent to the Hub as an 'Other' document type on the back-end.

Lawful Presence Documentation	alien Number	i94 Number	passport Number	passport Issuing Country	sevis Id	naturalization Certificate Number	citizenship Number	card Number	document Expiration Date	Other Type Text
RENTRY_PERMIT	R								O	
PERMANENT_RESIDENT_CARD_I_551	R							R	O	
CERTIFICATE_OF_CITIZENSHIP	O						R			
CERTIFICATE_OF_NATURALIZATION	O					R				
MACHINE_READABLE_IMMIGRANT_VISA_WITH_TEMPORARY_I_551_LANGUAGE	R		R	R					O	
TEMPORARY_I_551_STAMP_ON_PASSPORT_OR_I_94_I_94A	R		O	O					O	
REFUGEE_TRAVEL_DOCUMENT	R								O	
EMPLOYMENT_AUTHORIZATION_CARD_I_766	R							R	R	
ARRIVAL_DEPARTURE_RECORD_IN_FOREIGN_PASSPORT_I_94		R			O				O	
ARRIVAL_DEPARTURE_RECORD_IN_UNEXPIRED_FOREIGN_PASSPORT_I_94		R	R	R	O				R	
FOREIGN_PASSPORT		O	R	R	O				R	
CERTIFICATE_OF_ELIGIBILITY_FOR_NONIMMIGRANT_STUDENT_STATUS_I_20		O	O	O	R				O	
CERTIFICATE_OF_ELIGIBILITY_FOR_EXCHANGE_VISITOR_STATUS_DS_2019		O	O	O	R				O	
OTHER	R		O	O	O				O	R
OTHER		R	O	O	O				O	R

Appendix 6: SES Application Sample Requests/Responses

Create App

```
{
    "application": {
        "coverageYear": 2018,
        "coverageState": "DE",
        "linkedSystemUserIdentifier": "billyj4444@yopmail.com",
        "contactInformation": {
            "email": "t2123@yopmail.com",
            "primaryPhoneNumber": {
                "number": "2222222222",
                "type": "WORK"
            }
        }
    },
    "applicationMembers": [{
            "householdContactIndicator": true,
            "firstName": "First",
            "lastName": "Last",
            "birthDate": "1908-01-01",
            "mailingAddress": {
                "streetName1": "Street",
                "cityName": "City",
                "stateCode": "DE",
                "zipCode": "19805",
                "countyName": "County",
                "countyFipsCode": "10003"
            }
        }
```

```
            ]
}
```

Create App Response

```
{
    "resultType": "SUCCESS",
    "links": {
        "rel": "self",
        "href": "/applications/7364836"
    },
    "result": {
        "insuranceApplicationIdentifier": 7364836,
        "applicationVersionNumber": 1,
        "applicationCreationDateTime": "2018-06-27T18:32:39+00:00",
        "linkedSystemUserIdentifier": "billyj4444@yopmail.com",
        "applicationMembers": {
            "1254107644534669912": {
                "firstName": "First",
                "lastName": "Last",
                "birthDate": "1908-01-01"
            }
        },
        "useCase": "INITIAL_APP",
        "insuranceApplicationType": "LEGACY_ONLINE"
    }
}
```

Add Member Request (Adding Dependent Child)

```
{
  "application": {
   "accountHolderIdentityProofedIndicator": true
```

```
 },
 "applicationMembers": [{
   "requestingCoverageIndicator": true,
              "demographic": {
                "birthDate": "2017-12-25",
                "name": {
                  "firstName": "Added",
                  "middleName": "A",
                  "lastName": "Child"
                }
              }
  }
 ]
}
```

Add Member Response (Adding Dependent Child)

```
{
  "resultType": "SUCCESS_WITH_ERROR",
  "links": {
    "rel": "self",
    "href": "/applications/7364836/members"
  },
  "result": {
    "insuranceApplicationIdentifier": 7364836,
    "applicationVersionNumber": 1,
    "versionDateTime": "2018-06-27T18:35:48+00:00",
    "applicationCreationDateTime": "2018-06-27T18:32:39+00:00",
    "creatingSystemId": "LEGACY_ONLINE",
    "creatingSystemRole": "FFM_ONLINE",
    "creatingUserId": "6234186",
    "useCase": "INITIAL_APP",
    "deletedIndicator": false,
    "currentVersionIndicator": true,
    "comments": "Create New Version",
    "lastModifiedDateTime": "2018-06-27T18:35:48+00:00",
    "lastModifiedUserId": "SES",
    "insuranceApplicationType": "LEGACY_ONLINE",
    "attestations": {
      "application": {
        "coverageYear": 2018,
        "coverageState": "DE",
        "subjectToSampleIndicator": false,
        "contactMemberIdentifier": "1254107644534669912",
        "contactInformation": {
          "email": "t2123@yopmail.com",
          "primaryPhoneNumber": {
            "number": "2222222222",
            "type": "WORK"
          }
        }
      },
      "members": {
        "1254107644534669912": {
          "demographic": {
            "birthDate": "1908-01-01",
            "name": {
              "firstName": "First",
              "lastName": "Last"
            },
            "mailingAddress": {
              "streetName1": "Street",
              "cityName": "City",
              "stateCode": "DE",
              "zipCode": "19805",
              "countyFipsCode": "10003",
              "countyName": "County"
            }
          },
          "income": {
            "incomeExceedFPL": "I_DONT_KNOW",
            "seasonalWorkerIndicator": true
          },
          "family": {
            "nonCustodialClaimerBothMFSParentonAppIndicator": true
          },
          "lawfulPresence": {
            "naturalizedCitizenIndicator": false,
            "noAlienNumberIndicator": false
          }
        },
        "1254107644534669914": {
          "requestingCoverageIndicator": true,
          "demographic": {
            "birthDate": "2017-12-25",
            "name": {
              "firstName": "Added",
              "middleName": "A",
              "lastName": "Child"
            }
          },
          "income": {
            "incomeExceedFPL": "I_DONT_KNOW",
            "seasonalWorkerIndicator": true
          },
          "family": {
            "nonCustodialClaimerBothMFSParentonAppIndicator": true
          },
          "lawfulPresence": {
            "naturalizedCitizenIndicator": false,
            "noAlienNumberIndicator": false
          }
        }
      }
    },
    "computed": {
      "members": {
        "1254107644534669912": {
          "indianStatus": "NOT_APPLICABLE",
          "indianStatusReason": "575_NON_APPLICANT_NOT_PREVIOUSLY_VERIFIED",
          "nonEscMecStatus": "NOT_APPLICABLE",
          "nonEscMecStatusReason": "555_N_A_RULE_DOES_NOT_APPLY",
          "ssnStatus": "NOT_APPLICABLE",
          "ssnStatusReason": "352_NO_SSN_PROVIDED",
          "citizenshipStatus": "NOT_APPLICABLE",
          "citizenshipStatusReason": "531_ATSTD_NON_CITIZEN",
          "citizenshipDataEventStatus": "NOT_APPLICABLE",
          "preliminaryMedicaidStatus": "NOT_APPLICABLE",
          "preliminaryMedicaidStatusReason": "551_APPLICATION_NOT_REQUESTING_FA",
          "preliminaryChipStatus": "NOT_APPLICABLE",
          "preliminaryChipStatusReason": "551_APPLICATION_NOT_REQUESTING_FA",
          "preliminaryAptcStatus": "NOT_APPLICABLE",
          "preliminaryAptcStatusReason": "551_APPLICATION_NOT_REQUESTING_FA",
          "preliminaryEmergencyMedicaidStatus": "NOT_APPLICABLE",
          "preliminaryEmergencyMedicaidStatusReason": "TRANSLATION_UNAVAILABLE",
          "dependentChildCoveredStatus": "NOT_APPLICABLE",
          "dependentChildCoveredStatusReason": "551_APPLICATION_NOT_REQUESTING_FA",
          "chipStateHealthBenefitStatus": "NOT_APPLICABLE",
          "chipStateHealthBenefitStatusReason": "551_NOT_REQUESTING_FINANCIAL_ASSISTANCE",
          "chipWaitingPeriodStatus": "NOT_APPLICABLE",
          "chipWaitingPeriodStatusReason": "551_NOT_REQUESTING_FINANCIAL_ASSISTANCE",
          "qhpLawfulPresenceStatus": "NO",
          "qhpLawfulPresenceStatusReason": "395_DID_NOT_ATTEST_TO_NATURALIZED_CITIZENSHIP_OR_LP",
          "escMecStatus": "NOT_APPLICABLE",
          "escMecStatusReason": "560_NOT_REQUESTING_COVERAGE",
          "qhpResidency": {
            "qhpResidencyStatus": "NOT_APPLICABLE",
            "qhpResidencyStatusReason": "555_N_A_RULE_DOES_NOT_APPLY"
          }
        },
        "1254107644534669914": {
          "indianStatus": "NOT_APPLICABLE",
          "indianStatusReason": "575_NON_APPLICANT_NOT_PREVIOUSLY_VERIFIED",
          "nonEscMecStatus": "NOT_APPLICABLE",
          "nonEscMecStatusReason": "555_N_A_RULE_DOES_NOT_APPLY",
          "ssnStatus": "NOT_APPLICABLE",
          "ssnStatusReason": "352_NO_SSN_PROVIDED",
          "citizenshipStatus": "NOT_APPLICABLE",
          "citizenshipStatusReason": "531_ATSTD_NON_CITIZEN",
          "citizenshipDataEventStatus": "NOT_APPLICABLE",
          "preliminaryMedicaidStatus": "NOT_APPLICABLE",
        "preliminaryMedicaidStatusReason": "551_APPLICATION_NOT_REQUESTING_FA",
        "preliminaryChipStatus": "NOT_APPLICABLE",
        "preliminaryChipStatusReason": "551_APPLICATION_NOT_REQUESTING_FA",
        "preliminaryAptcStatus": "NOT_APPLICABLE",
        "preliminaryAptcStatusReason": "551_APPLICATION_NOT_REQUESTING_FA",
        "preliminaryEmergencyMedicaidStatus": "NOT_APPLICABLE",
        "preliminaryEmergencyMedicaidStatusReason": "TRANSLATION_UNAVAILABLE",
        "dependentChildCoveredStatus": "NOT_APPLICABLE",
        "dependentChildCoveredStatusReason": "551_APPLICATION_NOT_REQUESTING_FA",
        "chipStateHealthBenefitStatus": "NOT_APPLICABLE",
        "chipStateHealthBenefitStatusReason": "551_NOT_REQUESTING_FINANCIAL_ASSISTANCE",
        "chipWaitingPeriodStatus": "NOT_APPLICABLE",
        "chipWaitingPeriodStatusReason": "551_NOT_REQUESTING_FINANCIAL_ASSISTANCE",
        "qhpLawfulPresenceStatus": "NO",
        "qhpLawfulPresenceStatusReason": "395_DID_NOT_ATTEST_TO_NATURALIZED_CITIZENSHIP_OR_LP",
        "escMecStatus": "NOT_APPLICABLE",
        "escMecStatusReason": "551_NOT_REQUESTING_FINANCIAL_ASSISTANCE",
        "qhpResidency": {
          "qhpResidencyStatus": "NOT_APPLICABLE",
          "qhpResidencyStatusReason": "555_N_A_RULE_DOES_NOT_APPLY"
        }
      }
    }
  }
},
"error": {
  "errorType": "CLIENT_ERROR",
  "apiMessage": "Data Validation issue, please correct the data and try again",
  "errorCode": "validation_errors",
  "errors": [
      {
        "errorCode": "required",
        "path": "attestations.members[\"1254107644534669914\"].demographic.noHomeAddressIndicator"
      },
      {
        "errorCode": "field_cannot_be_empty",
        "path": "application.attestations.members[\"1254107644534669912\"].requestingCoverageIndicator"
      },
      {
        "errorCode": "required",
        "path": "attestations.members[\"1254107644534669914\"].lawfulPresence.lawfulPresenceStatusIndicator"
      },
      {
        "errorCode": "required",
        "path": "attestations.members[\"1254107644534669912\"].requestingCoverageIndicator"
      },
      {
        "errorCode": "required",
        "path": "attestations.application.requestingFinancialAssistanceIndicator"
      }
    ]
  }
}
```

Update Application Request

```
{
 "application" : {
  "accountHolderIdentityProofedIndicator" : true,
  "legalAttestations" : {
   "changeInformationAgreementIndicator" : true,
   "penaltyOfPerjuryAgreementIndicator" : true,
   "nonIncarcerationAgreementIndicator" : true,
   "renewEligibilityYearQuantity" : 0

  },
  "requestingFinancialAssistanceIndicator" : true

},
"members" : {
  "1254107644534669912" : {
   "requestingCoverageIndicator" : true,
   "demographic" : {
    "maritalStatus" : "UNMARRIED",
    "noHomeAddressIndicator" : false,
      "ssn": "339147551",
    "americanIndianAlaskanNativeIndicator" : true,
    "sex" : "MALE",
    "homeAddress" : {
          "streetName1": "Street",
                                        "cityName": "City",
                                        "stateCode": "DE",
                                        "zipCode": "19805",
                                        "countyName": "County",
                                        "countyFipsCode": "10003"
    }
   },
   "income" : {
    "annualTaxIncome" : {
      "incomeAmount" : 60000,
      "unknownIncomeIndicator" : false,
      "variableIncomeIndicator" : false
     },
     "currentIncome" : {
      "currentIncome1" : {
       "incomeAmount" : 60000,
       "incomeSourceType" : "JOB",
       "incomeFrequencyType" : "ANNUALLY",
       "jobIncome" : {
        "employerName" : "fhh"
       }
      }
     }
    },
    "family" : {
     "claimsDependentIndicator" : true,
     "parentCaretakerIndicator" : false,
     "absentParentIndicator" : false,
     "taxDependentIndicator" : false,
     "taxFilerIndicator" : true,
     "taxReturnFilingStatusType" : "HEAD_OF_HOUSEHOLD"
    },
    "lawfulPresence" : {
     "citizenshipIndicator" : true,
     "naturalizedCitizenIndicator" : false,
     "noAlienNumberIndicator" : false
    },
    "insuranceCoverage" : {
     "enrolledCoverages" : [{
```

```
     "insuranceMarketType" : "NONE"
    }
   ],
   "employerSponsoredCoverageOffers" : {
    "escOffer1" : {
     "employer" : {
      "name" : "Pizza Hut",
      "streetName1" : "123 Main St",
      "streetName2" : "Apt 100",
      "cityName" : "Wilmington",
      "stateCode" : "DE",
      "zipCode" : "19805",
      "plus4Code" : "1111",
      "countryCode" : "001",
      "countyName" : "Wilmington",
      "countyFipsCode" : "10003",
      "employerPhoneNumber" : "7038675309",
      "contact" : {
       "firstName" : "Pizza",
       "middleName" : "Hut",
       "lastName" : "Hut",
       "suffix" : "Jr.",
       "email" : "employer.contact@gmail.com",
       "phoneNumber" : "7038675309"
      }
     }
    }
   },
   "offeredEmployeeCoverage" : "NO"
  },
   "medicaid" : {
    "medicaidDeniedIndicator" : false

   },
   "chip" : {
    "coverageEndedIndicator" : false
   },
   "other" : {
    "americanIndianAlaskanNative" : {
     "personRecognizedTribeIndicator" : false
    },

    "reconcilePtcIndicatorType" : "NOT_APPLICABLE",
    "incarcerationType" : "NOT_INCARCERATED"
   }
  },
  "1254107644534669914":{
        "requestingCoverageIndicator": true,
          "nonMagi": {
            "blindOrDisabledIndicator": false,
            "longTermCareIndicator": false
          },
          "demographic": {
            "maritalStatus": "UNMARRIED",
            "noHomeAddressIndicator": false,
            "americanIndianAlaskanNativeIndicator": false,
            "sex": "FEMALE",
            "homeAddress": {
              "streetName1": "Street",
                             "cityName": "City",
                                   "stateCode": "DE",
                                   "zipCode": "19805",
                                   "countyName": "County",
                                   "countyFipsCode": "10003"
                },
                "mailingAddress": {
                    "streetName1": "Street",
                                   "cityName": "City",
                                   "stateCode": "DE",
                                   "zipCode": "19805",
                                   "countyName": "County",
                                   "countyFipsCode": "10003"
                }
            },
            "income": {
                "annualTaxIncome": {
                    "incomeAmount": 0,
                    "unknownIncomeIndicator": false,
                    "variableIncomeIndicator": false
                }
            },
            "family": {
                "claimsDependentIndicator": false,
                "parentCaretakerIndicator": false,
                "pregnancyIndicator": false,
                "absentParentIndicator": false,
                "taxDependentIndicator": true,
                "taxFilerIndicator": false
            },
            "lawfulPresence": {
```

```
           "citizenshipIndicator": true,
           "naturalizedCitizenIndicator": false,
           "noAlienNumberIndicator": false
         },
          "insuranceCoverage" : {
   "enrolledCoverages" : [{
     "insuranceMarketType" : "NONE"
    }
   ],
   "offeredEmployeeCoverage" : "NO"
  },
         "medicaid": {
           "medicaidDeniedIndicator": false
         },
          "chip" : {
   "coverageEndedIndicator" : false
  },
         "other": {
           "incarcerationType": "NOT_INCARCERATED"
         }
 }
},
"household" : {
 "familyRelationships" : [
  [
   "12541076445346699 12",
   "SELF",
   "12541076445346699 12"
  ],
   [
    "12541076445346699l2",
    "PARENT",
    "1254107644534669914",
    {
       "resideTogetherIndicator": true
    }
   ]
  ],
  "taxRelationships" : [
   [
    "1254107644534669912",
    "TAX_FILER",
    "1254107644534669912"
   ],
    [
    "1254107644534669912",
    "TAX_FILER",
    "1254107644534669914"
   ]
  ]
 }
}
```

Update App Response

```
{
  "resultType": "SUCCESS",
  "links": {
    "rel": "self",
    "href": "/applications/7364836"
  },
  "result": {
    "insuranceApplicationIdentifier": 7364836,
    "applicationVersionNumber": 1,
    "versionDateTime": "2018-06-27T18:42:37+00:00",
    "applicationCreationDateTime": "2018-06-27T18:32:39+00:00",
    "creatingSystemId": "LEGACY_ONLINE",
    "creatingSystemRole": "FFM_ONLINE",
    "creatingUserId": "6234186",
    "useCase": "INITIAL_APP",
    "deletedIndicator": false,
    "currentVersionIndicator": true,
    "comments": "Create New Version",
    "lastModifiedDateTime": "2018-06-27T18:42:37+00:00",
    "lastModifiedUserId": "testUserId",
    "insuranceApplicationType": "LEGACY_ONLINE",
    "computed": {
      "members": {
        "1254107644534669912": {
          "personTrackingNumber": "1202146336574826343",
          "indianStatus": "NO",
          "indianStatusReason": "178_ATSTD_NOT_INDIAN",
          "nonEscMecStatus": "NO",
          "nonEscMecStatusReason": "999_N_A",
          "ssnStatus": "YES",
          "ssnStatusReason": "999_N_A",
          "medicaidLawfulPresenceStatus": "NOT_APPLICABLE",
          "medicaidLawfulPresenceStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
          "citizenshipStatus": "YES",
          "citizenshipStatusReason": "999_N_A",
          "citizenshipDataEventStatus": "HUB_CALL_COMPLETED",
          "citizenshipDataFoundIndicator": true,
          "preliminaryMedicaidStatus": "NO",
          "preliminaryMedicaidStatusReason": "106_NOT_ALL_MDCAID_ELGBLT_CRITERIA_MET",
          "preliminaryChipStatus": "NO",
          "preliminaryChipStatusReason": "107_NOT_CHIP_ELGBL",
          "preliminaryAptcStatus": "YES",
          "preliminaryAptcStatusReason": "999_N_A",
          "preliminaryEmergencyMedicaidStatus": "NO",
          "preliminaryEmergencyMedicaidStatusReason": "109_NOT_MEET_REQUIREMENTS_FOR_EMERGENCY_MDCAID",
          "dependentChildCoveredStatus": "NOT_APPLICABLE",
          "dependentChildCoveredStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
          "chipStateHealthBenefitStatus": "NOT_APPLICABLE",
          "chipStateHealthBenefitStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
          "chipWaitingPeriodStatus": "NOT_APPLICABLE",
          "chipWaitingPeriodStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
          "emergencyMedicaidStatus": "NO",
          "emergencyMedicaidStatusReason": "109_NOT_MEET_REQUIREMENTS_FOR_EMERGENCY_MDCAID",
          "qhpLawfulPresenceStatus": "NOT_APPLICABLE",
          "qhpLawfulPresenceStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
          "under100FplWithLawfulPresenceDmiIndicator": false,
          "medicaidChipStandard": {
            "medicaidStandardPercent": 87,
            "medicaidStandardBasisType": "PARENT_CARETAKER_RELATIVE_CATEGORY",
            "chipStandardPercent": 0,
            "adultGroupCategoryStatus": "NO",
            "childCaretakerDeprivedStatus": "NOT_APPLICABLE",
            "parentCaretakerCategoryStatus": "YES",
            "parentCaretakerChildList": [
              "1254107644534669914"
            ]
          },
          "escMecStatus": "NO",
          "escMecStatusReason": "999_N_A",
          "incarcerationStatus": "NO",
          "incarcerationStatusReason": "100_ONLY_SSA_DATA_AVAILABLE_BUT_NOT_COMPATIBLE",
          "income": {
            "employer": [
              {
                "employer": "fhh"
              }
            ],
            "medicaidIncomeStatus": "NO",
            "chipIncomeStatus": "NO",
            "medicaidChipIncomeStatusReason": "533_ATSTD_ABOVE_MDCAID_STANDARD_CHIP_STANDARD_IS_0_INELGBLE_FOR_MDCAID_CHIP",
            "attestedMonthlyIndividualIncomeAmt": 5000,
            "attestedAnnualizedAptcIndividualIncomeAmt": 60000,
            "irsDataStatus": "HUB_CALL_COMPLETED",
            "irsDataStatusReason": "999"
          },
          "medicaidHouseholdComposition": {
            "medicaidTaxRoleType": "INDIVIDUAL_TAX_FILER",
            "medicaidHouseHoldStatus": "YES",
            "medicaidHouseHoldStatusReason": "999",
            "medicaidHouseholdSize": 2,
            "medicaidHouseholdMemberIdentifiers": [
              1254107644534669912,
              1254107644534669914
            ]
          },
          "medicaidNonMagiReferralStatus": "YES",
      "medicaidNonMagiReferralStatusReason": "598_NON_MAGI_REFERRAL_CRITERIA_MET",
      "qhpResidency": {
        "qhpResidencyStatus": "YES",
        "qhpResidencyStatusReason": "999_N_A",
        "taxHouseholdResidencyAppliesIndicator": false
      },
      "medicaidResidencyStatus": "YES",
      "medicaidResidencyStatusReason": "999_N_A",
      "fiveYearBarStatus": "NOT_APPLICABLE",
      "fiveYearBarStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
      "residencyAddress": {
        "residencyAddressSourceType": "HOME",
        "streetName1": "Street",
        "cityName": "City",
        "stateCode": "DE",
        "zipCode": "19805",
        "countyName": "County",
        "countyFipsCode": "10003"
      }
    },
    "1254107644534669914": {
      "personTrackingNumber": "1245826307461722307",
      "indianStatus": "NO",
      "indianStatusReason": "178_ATSTD_NOT_INDIAN",
      "nonEscMecStatus": "NO",
      "nonEscMecStatusReason": "999_N_A",
      "ssnStatus": "NOT_APPLICABLE",
      "ssnStatusReason": "352_NO_SSN_PROVIDED",
      "medicaidLawfulPresenceStatus": "NOT_APPLICABLE",
      "medicaidLawfulPresenceStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
"citizenshipStatus": "YES",
"citizenshipStatusReason": "397_DID_NOT_PROVIDE_SSN_WITH_CITIZENSHIP_ATTESTATION",
"citizenshipDataEventStatus": "NOT_APPLICABLE",
"preliminaryMedicaidStatus": "NO",
"preliminaryMedicaidStatusReason": "106_NOT_ALL_MDCAID_ELGBLT_CRITERIA_MET",
"preliminaryChipStatus": "NO",
"preliminaryChipStatusReason": "107_NOT_CHIP_ELGBL",
"preliminaryAptcStatus": "YES",
"preliminaryAptcStatusReason": "999_N_A",
"preliminaryEmergencyMedicaidStatus": "NO",
"preliminaryEmergencyMedicaidStatusReason": "109_NOT_MEET_REQUIREMENTS_FOR_EMERGENCY_MDCAID",
"dependentChildCoveredStatus": "NOT_APPLICABLE",
"dependentChildCoveredStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
"chipStateHealthBenefitStatus": "NOT_APPLICABLE",
"chipStateHealthBenefitStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
"chipWaitingPeriodStatus": "NOT_APPLICABLE",
"chipWaitingPeriodStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
"emergencyMedicaidStatus": "NO",
"emergencyMedicaidStatusReason": "109_NOT_MEET_REQUIREMENTS_FOR_EMERGENCY_MDCAID",
"qhpLawfulPresenceStatus": "NOT_APPLICABLE",
"qhpLawfulPresenceStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
"under100FplWithLawfulPresenceDmiIndicator": false,
"medicaidChipStandard": {
  "medicaidStandardPercent": 212,
  "medicaidStandardBasisType": "CHILD_CATEGORY",
  "chipStandardPercent": 0,
  "adultGroupCategoryStatus": "NO",
  "childCaretakerDeprivedStatus": "NOT_APPLICABLE",
  "parentCaretakerCategoryStatus": "NO"
},
"escMecStatus": "NO",
"escMecStatusReason": "999_N_A",
"incarcerationStatus": "NO",
"incarcerationStatusReason": "999_N_A",
"income": {
  "medicaidIncomeStatus": "NO",
  "chipIncomeStatus": "NO",
  "medicaidChipIncomeStatusReason": "533_ATSTD_ABOVE_MDCAID_STANDARD_CHIP_STANDARD_IS_0_INELGBLE_FOR_MDCAID_CHIP",
  "attestedMonthlyIndividualIncomeAmt": 0,
  "attestedAnnualizedAptcIndividualIncomeAmt": 0,
  "irsDataStatus": "HOLD",
  "irsDataStatusReason": "546"
},
"medicaidHouseholdComposition": {
  "medicaidTaxRoleType": "DEPENDENT_ON_PARENT",
  "medicaidHouseHoldStatus": "YES",
  "medicaidHouseHoldStatusReason": "999",
  "medicaidHouseholdSize": 2,
  "medicaidHouseholdMemberIdentifiers": [
    1254107644534669912,
    1254107644534669914
  ]
},
"medicaidNonMagiReferralStatus": "NO",
"medicaidNonMagiReferralStatusReason": "108_NOT_MEET_REQUIREMENTS_FOR_NON_MAGI_REFERRAL",
"qhpResidency": {
  "qhpResidencyStatus": "YES",
  "qhpResidencyStatusReason": "999_N_A",
  "taxHouseholdResidencyAppliesIndicator": false
},
        "medicaidResidencyStatus": "YES",
        "medicaidResidencyStatusReason": "999_N_A",
        "fiveYearBarStatus": "NOT_APPLICABLE",
        "fiveYearBarStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
        "residencyAddress": {
          "residencyAddressSourceType": "HOME",
          "streetName1": "Street",
          "cityName": "City",
          "stateCode": "DE",
          "zipCode": "19805",
          "countyName": "County",
          "countyFipsCode": "10003"
        }
      }
    },
    "taxHouseholds": {
      "1254107644534669912": {
        "taxHouseHoldComposition": {
          "taxHouseholdStatus": "YES",
          "taxHouseholdStatusReason": "999_N_A",
          "taxHouseholdMemberIdentifiers": [
            "1254107644534669912",
            "1254107644534669914"
          ]
        },
        "annualIncome": {
          "attestedAnnualTaxHouseholdIncomeAmount": 60000,
          "attestedAnnualTaxHouseholdIncomePercent": 369.46,
          "annualIncomeStatus": "YES",
          "annualIncomeStatusReason": "564_VERIFIED_BASED_ON_EXTERNAL_INCOME_DATA",
            "usedDocumentAnnualTaxIncomeAmountExplanationRequiredAmount": false
          }
        }
      }
    }
  }
}
```

Submit App Request

```
{
 "idProofingIndicator": true,
 "signatureName": "Susan Griffith",
 "signatureDate": "2017-12-14",
 "signatureType": "APPLICANT"
}
```

Submit App Response

```
{
  "resultType": "SUCCESS",
  "links": {
    "rel": "self",
    "href": "/application/7364836"
  },
  "result": {
    "coverageYear": "2018",
    "insuranceApplicationIdentifier": 7364836,
    "applicationVersionNumber": 1,
    "lastModifiedDateTime": "2018-06-27T18:43:16.395Z",
    "lastModifiedUserId": "testUserId",
    "deletedIndicator": false,
  "currentVersionIndicator": true,
  "versionDateTime": "2018-06-27T18:43:16.395Z",
  "applicationStatus": "SUBMITTED",
  "comments": "Create New Version",
  "creatingSystemId": "LEGACY_ONLINE",
  "creatingSystemRole": "FFM_ONLINE",
  "creatingUserId": "6234186",
  "applicationSubmissionDateTime": "2018-06-27T18:43:10.436Z",
  "creationOriginType": "FFM_ONLINE",
  "applicationCreationDateTime": "2018-06-27T18:32:39.927Z",
  "useCase": "INITIAL_APP",
  "submissionOriginType": "FFM_ONLINE",
  "lastConsumerSubmissionDateTime": "2018-06-27T18:43:10.436Z",
  "insuranceApplicationType": "LEGACY_ONLINE",
  "computed": {
    "members": {
      "1254107644534669912": {
        "personTrackingNumber": "1202146336574826343",
        "indianStatus": "NO",
        "indianStatusReason": "178_ATSTD_NOT_INDIAN",
        "nonEscMecStatus": "NO",
        "nonEscMecStatusReason": "999_N_A",
        "qhpStatus": "YES",
        "qhpStatusReason": "999_N_A",
        "ssnStatus": "YES",
        "ssnStatusReason": "999_N_A",
        "medicaidLawfulPresenceStatus": "YES",
        "medicaidLawfulPresenceStatusReason": "999_N_A",
        "under100FplWithLawfulPresenceDmiIndicator": false,
        "aptcStatus": "YES",
        "aptcStatusReason": "999_N_A",
        "secretaryHardshipExemptionIndicator": false,
        "citizenshipStatus": "YES",
        "citizenshipStatusReason": "999_N_A",
        "citizenshipDataEventStatus": "HUB_CALL_COMPLETED",
        "citizenshipDataFoundIndicator": true,
        "preliminaryMedicaidStatus": "NO",
        "preliminaryMedicaidStatusReason": "106_NOT_ALL_MDCAID_ELGBLT_CRITERIA_MET",
        "preliminaryChipStatus": "NO",
        "preliminaryChipStatusReason": "107_NOT_CHIP_ELGBL",
        "preliminaryAptcStatus": "YES",
        "preliminaryAptcStatusReason": "999_N_A",
        "preliminaryEmergencyMedicaidStatus": "NO",
        "preliminaryEmergencyMedicaidStatusReason": "109_NOT_MEET_REQUIREMENTS_FOR_EMERGENCY_MDCAID",
        "dependentChildCoveredStatus": "NOT_APPLICABLE",
        "dependentChildCoveredStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
        "chipStateHealthBenefitStatus": "NOT_APPLICABLE",
        "chipStateHealthBenefitStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
        "chipWaitingPeriodStatus": "NOT_APPLICABLE",
        "chipWaitingPeriodStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
        "emergencyMedicaidStatus": "NO",
        "emergencyMedicaidStatusReason": "109_NOT_MEET_REQUIREMENTS_FOR_EMERGENCY_MDCAID",
        "unbornChildChipStatus": "NOT_APPLICABLE",
        "unbornChildChipStatusReason": "555_N_A_RULE_DOES_NOT_APPLY",
        "transferApplicantToStateStatus": "YES",
        "transferApplicantToStateStatusReason": "555_N_A_RULE_DOES_APPLY",
        "qhpLawfulPresenceStatus": "NOT_APPLICABLE",
        "qhpLawfulPresenceStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
        "csrStatus": "NO",
        "csrStatusReason": "419_GREATER_THAN_250FPL_AND_NOT_INDIAN",
"csrVariant": "NOT_APPLICABLE",
"medicaidStatus": "NO",
"medicaidStatusReason": "106_NOT_ALL_MDCAID_ELGBLT_CRITERIA_MET",
"chipStatus": "NO",
"chipStatusReason": "107_NOT_CHIP_ELGBL",
"medicaidChipStandard": {
  "medicaidStandardPercent": 87,
  "medicaidStandardBasisType": "PARENT_CARETAKER_RELATIVE_CATEGORY",
  "chipStandardPercent": 0,
  "adultGroupCategoryStatus": "NO",
  "childCaretakerDeprivedStatus": "NOT_APPLICABLE",
  "parentCaretakerCategoryStatus": "YES",
  "parentCaretakerChildList": [
    "1254107644534669914"
  ]
},
"escMecStatus": "NO",
"escMecStatusReason": "999_N_A",
"incarcerationStatus": "NO",
"incarcerationStatusReason": "100_ONLY_SSA_DATA_AVAILABLE_BUT_NOT_COMPATIBLE",
"income": {
  "medicaidIncomeStatus": "NO",
  "chipIncomeStatus": "NO",
  "medicaidChipIncomeStatusReason": "408_ATSTD_ABOVE_MDCAID_CHIP_STANDARD",
  "attestedMonthlyIndividualIncomeAmt": 5000,
  "attestedAnnualizedAptcIndividualIncomeAmt": 60000,
  "irsDataStatus": "HUB_CALL_COMPLETED",
  "irsDataStatusReason": "999"
},
"medicaidHouseholdComposition": {
        "medicaidTaxRoleType": "INDIVIDUAL_TAX_FILER",
        "medicaidHouseHoldStatus": "YES",
        "medicaidHouseHoldStatusReason": "999",
        "medicaidHouseholdSize": 2,
        "medicaidHouseholdMemberIdentifiers": [
          1254107644534669912,
          1254107644534669914
        ]
      },
      "medicaidNonMagiReferralStatus": "YES",
      "medicaidNonMagiReferralStatusReason": "598_NON_MAGI_REFERRAL_CRITERIA_MET",
      "qhpResidency": {
        "qhpResidencyStatus": "YES",
        "qhpResidencyStatusReason": "999_N_A",
        "taxHouseholdResidencyAppliesIndicator": false
      },
      "medicaidResidencyStatus": "YES",
      "medicaidResidencyStatusReason": "999_N_A",
      "fiveYearBarStatus": "NOT_APPLICABLE",
      "fiveYearBarStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
      "residencyAddress": {
        "residencyAddressSourceType": "HOME",
        "streetName1": "Street",
        "cityName": "City",
        "stateCode": "DE",
        "zipCode": "19805",
        "countyName": "County",
        "countyFipsCode": "10003"
      }
    },
        "1254107644534669914": {
          "personTrackingNumber": "1245826307461722307",
          "indianStatus": "NO",
          "indianStatusReason": "178_ATSTD_NOT_INDIAN",
          "nonEscMecStatus": "NO",
          "nonEscMecStatusReason": "999_N_A",
          "qhpStatus": "YES",
          "qhpStatusReason": "999_N_A",
          "ssnStatus": "NOT_APPLICABLE",
          "ssnStatusReason": "352_NO_SSN_PROVIDED",
          "medicaidLawfulPresenceStatus": "NOT_APPLICABLE",
          "medicaidLawfulPresenceStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
          "under100FplWithLawfulPresenceDmiIndicator": false,
          "aptcStatus": "YES",
          "aptcStatusReason": "999_N_A",
          "secretaryHardshipExemptionIndicator": false,
          "citizenshipStatus": "YES",
          "citizenshipStatusReason": "397_DID_NOT_PROVIDE_SSN_WITH_CITIZENSHIP_ATTESTATION",
          "citizenshipDataEventStatus": "NOT_APPLICABLE",
          "preliminaryMedicaidStatus": "NO",
          "preliminaryMedicaidStatusReason": "106_NOT_ALL_MDCAID_ELGBLT_CRITERIA_MET",
          "preliminaryChipStatus": "NO",
          "preliminaryChipStatusReason": "107_NOT_CHIP_ELGBL",
          "preliminaryAptcStatus": "YES",
          "preliminaryAptcStatusReason": "999_N_A",
          "preliminaryEmergencyMedicaidStatus": "NO",
          "preliminaryEmergencyMedicaidStatusReason": "109_NOT_MEET_REQUIREMENTS_FOR_EMERGENCY_MDCAID",
          "chipStateHealthBenefitStatus": "NOT_APPLICABLE",
          "chipStateHealthBenefitStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
          "chipWaitingPeriodStatus": "NOT_APPLICABLE",
          "chipWaitingPeriodStatusReason": "652_NOT_PRELIM_MEDICAID_OR_PRELIM_CHIP_ELIGIBLE",
          "emergencyMedicaidStatus": "NO",
          "emergencyMedicaidStatusReason": "109_NOT_MEET_REQUIREMENTS_FOR_EMERGENCY_MDCAID",
          "unbornChildChipStatus": "NOT_APPLICABLE",
          "unbornChildChipStatusReason": "555_N_A_RULE_DOES_NOT_APPLY",
          "qhpLawfulPresenceStatus": "NOT_APPLICABLE",
          "qhpLawfulPresenceStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
          "csrStatus": "NO",
          "csrStatusReason": "419_GREATER_THAN_250FPL_AND_NOT_INDIAN",
          "csrVariant": "NOT_APPLICABLE",
          "medicaidStatus": "NO",
          "medicaidStatusReason": "106_NOT_ALL_MDCAID_ELGBLT_CRITERIA_MET",
          "chipStatus": "NO",
          "chipStatusReason": "107_NOT_CHIP_ELGBL",
          "medicaidChipStandard": {
            "medicaidStandardPercent": 212,
            "medicaidStandardBasisType": "CHILD_CATEGORY",
            "chipStandardPercent": 0,
            "adultGroupCategoryStatus": "NO",
            "childCaretakerDeprivedStatus": "NOT_APPLICABLE",
            "parentCaretakerCategoryStatus": "NO"
          },
          "escMecStatus": "NO",
          "escMecStatusReason": "999_N_A",
          "incarcerationStatus": "NO",
          "incarcerationStatusReason": "999_N_A",
          "income": {
            "medicaidIncomeStatus": "NO",
            "chipIncomeStatus": "NO",
            "medicaidChipIncomeStatusReason": "408_ATSTD_ABOVE_MDCAID_CHIP_STANDARD",
      "attestedMonthlyIndividualIncomeAmt": 0,
      "attestedAnnualizedAptcIndividualIncomeAmt": 0,
      "irsDataStatus": "NOT_APPLICABLE",
      "irsDataStatusReason": "551"
    },
    "medicaidHouseholdComposition": {
      "medicaidTaxRoleType": "DEPENDENT_ON_PARENT",
      "medicaidHouseHoldStatus": "YES",
      "medicaidHouseHoldStatusReason": "999",
      "medicaidHouseholdSize": 2,
      "medicaidHouseholdMemberIdentifiers": [
        1254107644534669912,
        1254107644534669914
      ]
    },
    "medicaidNonMagiReferralStatus": "NO",
    "medicaidNonMagiReferralStatusReason": "108_NOT_MEET_REQUIREMENTS_FOR_NON_MAGI_REFERRAL",
    "qhpResidency": {
      "qhpResidencyStatus": "YES",
      "qhpResidencyStatusReason": "999_N_A",
      "taxHouseholdResidencyAppliesIndicator": false
    },
    "medicaidResidencyStatus": "YES",
    "medicaidResidencyStatusReason": "999_N_A",
    "fiveYearBarStatus": "NOT_APPLICABLE",
    "fiveYearBarStatusReason": "569_APPLICANT_ATSTD_CITIZEN",
    "residencyAddress": {
      "residencyAddressSourceType": "HOME",
      "streetName1": "Street",
      "cityName": "City",
          "stateCode": "DE",
          "zipCode": "19805",
          "countyName": "County",
          "countyFipsCode": "10003"
        }
      }
    },
    "application": {
      "bestSEP": {
        "status": "NO",
        "statusReason": "252_NOT_GRANTED_NOT_SEP_ELGBL",
        "subjectToSviIndicator": false
      },
      "finalQhpEffectiveStartDate": "2018-08-01",
      "finalQhpEffectiveEndDate": "2018-12-31",
      "enrollmentGroups": [
        {
          "enrollmentGroupIdentifier": "984ddfed-5c2d-47dc-9a99-31c491126298",
          "enrollmentGroupName": "Group 0",
          "insuranceProductDivisionType": "HEALTHCARE",
          "enrollees": {
            "1254107644534669914": {
              "personTrackingNumber": "1245826307461722307",
              "subscriberIndicator": false
            },
            "1254107644534669912": {
              "personTrackingNumber": "1202146336574826343",
              "subscriberIndicator": true
            }
          }
        }
      ]
    },
    "taxHouseholds": {
      "1254107644534669912": {
        "annualIncome": {
          "attestedAnnualTaxHouseholdIncomeAmount": 60000,
          "attestedAnnualTaxHouseholdIncomePercent": 369.46,
          "annualIncomeStatus": "YES",
          "annualIncomeStatusReason": "999_N_A"
        },
        "maxAPTC": {
          "ptcFplPercentage": 369.46,
          "taxHouseholdAnnualIncomeAmount": 60000,
          "maxAPTCAmount": 1257,
          "slcspPremiumAmount": 1734.53,
          "slcspIdentifier": "76168DE0420004"
        },
        "taxHouseHoldComposition": {
          "taxHouseHoldStatus": "YES",
          "taxHouseHoldStatusReason": "999_N_A",
          "taxHouseholdMemberIdentifiers": [
            1254107644534669912,
            1254107644534669914
          ]
        }
      }
    },
    "variableDeterminations": [
      {
          "creationDateTime": "2018-06-27T18:43:10.436Z",
          "memberVariableDeterminations": [
            {
              "1254107644534669912": {
                "catastrophicEligibilityIndicator": false,
                "lpaIndicator": false
              }
            },
            {
              "1254107644534669914": {
                "catastrophicEligibilityIndicator": true,
                "catastrophicEligibilityReasonType": "AGE",
                "lpaIndicator": false
              }
            }
          ],
          "coverageState": "DE"
        }
      ]
    }
  }
}
```