WIP CFSA Design Spec

System diagram

Implementation Option 1: No backend

Hard-code all the eligibility logic and questions into the front-end. User inputs all there
PROs:

- No points of service integration

CONs:

- Other systems can't use the same eligibility criteria.
- Progam owners can’t easily add, update or delete eligibility criteria
- If we want to build a new front-end we have to repeat the eligibility logic.
- Couples front-end to eligibility results.

Implementation Option 2: Backend API + front end

Store and share eligibility criteria via an eligibility API (e.g. backend talks to front-end)
PROs:

- Easier to build new front-ends.
- Possible for services to submit or change new eligibility criteria.
- Potential to programmatically define these changes and have them be auto-populated in front-end systems

CONs:

- More complex to implement.
- Complexities of integration: If the backend changes the front end could break.

As a thought experiment, we can start by grouping related questions to see if the front end can programmatically add or update questions in a way to make sense, but likely it will need some customization for the best user experience.

Front-end can either hard-code question ids or programmatically fetch and group related questions. [ADD ME:] PROs and CONs to this approach

API Resources

Eligibility Questions

An eligibility question is a text string representing information to be gathered from a citizen.

```
{
  "uuid": "63febb90-2a56-42c0-986a-1e09611ce620",
  "question": "What is your age?",
  "category": "demographic",
  "subcategory": "age",
  "data_type": number
```

}

At this time, anyone can create or fetch eligibility questions, but eventually only superadmins should create eligibility questions. This is because we don't want a lot questions that essentially ask the same thing. IMPROVEMENT: How can we fuzzy match submitted questions to ensure minimal overlap? Goal is that we don't ask essentially the same information more than once.
Creating eligibility questions should happen a lot when the system is created an infrequently after.

Program

There can only be one program per eligibility criteria (but multiple eligibility criteria per program). Each program must register itself. Will be important when handling authorizations.

Eligibility Criteria

An eligibility criteria associates a program with a set of questions and corresponding answers required for eligibility


```
{
 "uuid": "4495d8c3-71d0-492a-ba9f-de21c05e298a",
 "program_uuid": "8aeb6956-5028-46f8-b89c-470466631fc0",
 "criteria": [{
   "question_uuid": "63febb90-2a56-42c0-986a-1e09611ce620",
   "alt_text": "How old are you?",
   "eligible_responses": [65, null],
   "eligible_response_type": "range"
 }]
}
```


Eligibility Submission


```
{
 "submission": [{
   "question_uuid": "63febb90-2a56-42c0-986a-1e09611ce620",
   "response": 67
 }]
}
```


Workflows

Program wants to integrate with the eligibility portal

[ADD ME]

User submits data and gets referrals

[ADD ME]

Open questions

- How to handle data conversions - say a program requires date of birth - this is essentially the same as age but we don't want to ask both age and date of birth.
- Is there an example where one program has multiple sets of eligibility criteria? Is this adequately handled with the feature that a single program can have multiple eligibility criteria?
- How to create a decision tree to minimize the number of questions any user has to answer

Future Improvements

- Integration with a citizen API / data hub which makes it possible to store information previously submitted by a citizen and their results (along with results expiration).
- eligibility status: associates a citizen with eligibility (which changes according to an expiration)
- enrollment status: same as eligibility only relevant after they have enrolled. Potentially requires integration with program systems unless it becomes the source of truth for enrollment.
- Design and integration with geolocation which makes it possible to identify a geographic region of eligibility for eligibility criteria and place addresses in or out of that geographic location.
- Integration with program systems which make it possible to understand enrollment status and submit an application on behalf of a citizen.
- Note the case where enrollment in a program is a criteria for eligibility in another program.
- Integration with data validation services
- Address validation
- Identity verification
- Authorizations: Only authorized persons can submit eligibility criteria on behalf of a program.
- Analytics dashboard of referrals
    - Brittney - sounds like something we already have