Team JSHD - Meeting 9/29/2021

- Q: How will we be getting guideline information?
    - We can download guideline definitions in the download page: http://hl7.org/fhir/uv/cpg/STU1/downloads.html
- Q: Are we fetching guidelines from another server, or will we host the guidelines in our own db?
    - We will only need to save ActivityDefinitions and PlanDefinitions into a database
    - id properties aren't unique across both activity and plan definitions, so we should create separate db schemas for each
- Q: How will we be hosting the project?
    - Possible options: heroku, firebase,
    - Firebase:
        - Free
        - Built in authentication
        - Harder to migrate, less flexible (e.g. less db options)
    - Heroku
        - Free
        - More flexible, easier to integrate with
    - Decided to go with heroku
- Q: What will we use with the API?
    - Express.js REST API
- Q: How will we test the code?
    - Options considered: Jest, Mocha and Jasmine
        - Jest is simple to use
        - Mocha does not have its own assertion libraries
            - Chai is used for assertion
            - Sinon is used for spies/stubs
        - Jasmine isn't able to run tests without a third party
    - Decided to go with Jest
- Q: Should we use CI/CD?
    - May be overkill; not sure if it's a necessary requirement of the project
    - Check with Mike?
    - In case we need it, github actions and jenkins are possible options
- Q: Do we need a frontend?
    - No. Maybe?
- Q: Authentication?
    - Unsure if needed

- **Todo:** Schedule a meeting with Alex in the JSHD channel in the course server to ask about frontend and authentication

Milestone / A1 Discussion:

- Milestone 1: Set up our environment, dependencies and tech stack
    - A github repo is created with our team members for the project
    - We have a node project that is runnable and has a simple test suite
        - Dependencies should be ready to install in one `npm i` command
        - Running the project takes only one command after dependencies are built
        - Running test suites takes only one command after dependencies are built
        - Project functionality and test coverage is not required for this milestone
    - The project is containerized in Docker
    - A readme is provided in top level directory of the repo with the following:
        - Instructions on installing and running the project
        - A summary of our tech stack and the relevant decision making
        - Milestones of the project with member ownership of each step
        - Directions to our collection of meeting notes
- Milestone 2: Create db/schemas and populate our downloaded guideline definitions
- Milestone 3: Implement express router and endpoints for fetching data and write unit tests
- Milestone 4: Write integration tests and create documentation for our app
    - We have written integration tests to verify the following functionality:
        - adding a new guideline to DB
        - reading a list of all available guidelines (?)
            - updating guideline's json
        - retrieve the guideline and verify contents
        - delete the guideline.
        - Should verify both activity and plan definition.
    - We have either a website or readme in our github repo (undecided), which documents the following:
        - What our project is about.
        - What the various API calls do, how to use them and an example.
- Possibly: ci/cd, alt goal: updating guidelines

Todo: Each member will pick a milestone to add details and take ownership of

Member Githubs

- Henry Gao: https://github.com/H-Gao
- James Huynh: https://github.com/CabbageCanFly
- Daniel Gao: https://github.com/dgao99
- Stew Esho: https://github.com/stewesho

Tech stack summary:

- Frontend: N/A
    - Will not be required as our project is just an API for other apps
- Backend: Node.js
    - Node.js with Express.js framework

        - Every team member has prior experience
        - Easy integration with mongodb
    - Flask
        - Not everyone used it
- Database: MongoDB
    - MongoDB
        - Mongoose integration with Node.js
        - Team has familiarity with mongo
    - We will only need to save ActivityDefinitions and PlanDefinitions into a database
    - id properties aren't unique across both activity and plan definitions, so we will create separate db schemas for each
    - Our database will only store JSON objects, so it's better not to use a relational database
- Hosting service: Heroku
    - Heroku
        - Free
        - Compared to Firebase, Heroku is more flexible and easier to integrate with other services
    - Firebase
        - Has its own DB and authentication
        - Limited flexibility. Ex. using our choice of DB.
- Container service: Docker
    - Vagrant
        - Advantage over docker: can run in non-linux environments because it functions as a VM
    - Docker
        - Team has prior experience with docker
        - More popular choice
    - App will be run in a linux environment, so we're choosing docker because we're more comfortable with it
- Testing: Jest
    - Jest is simple to use
    - Mocha does not have its own assertion libraries
        - Chai is used for assertion
        - Sinon is used for spies/stubs
    - Jasmine isn't able to run tests without a third party
    - Decided to go with Jest because the scope of the project isn't large enough to justify using a more heavyweight option like mocha

Todo: Henry will create github repo and make everyone an admin.

Decision to end meeting