CSCI 2263
PI 02 - Design

Assigned: February 7, 2020
Due: February 21, 2020 @ 17:00h

Project Iteration 2: Design

In this iteration you are refining your initial requirements and starting to code. Start with your requirements document (copy it over). Update it based on feedback received from your requirements document, and include the explicit new refinements listed below as well. Your document needs to include all of the following components, in sections/subsections named as the boldface below.

- Give your names, group number, and project title at the top.
- Copy over and update the Vision Statement, Features list, and UI sketches from iteration 1 to reflect your improved understanding of where the project is headed. Additionally modify your feature list to identify key features of your project, the features or use-cases you plan on targeting first for implementation, and the ”if time permits” features as extended features.
- The Use-cases again - if they were good and did not change just copy them over from iteration 1. Otherwise, revise as needed. As in the previous iteration, you only need to give use-cases for non-trivial sequences of events that are not already clear from the UI sketches or other information.
- Carefully document any Interfaces internal to your application. For example if you are using a RESTful server, give full documentation of all endpoints. See the Spotify API endpoint documentation for an example of well-documented endpoints; the Todo example of lecture and the first homework are other examples of endpoint documentation. The happy path method is to use Postman to specify your API as we did in the TODO app and the first assignment; later you can turn your specification into tests.
- UML Class diagrams of your design.
  - The goal here is to make an initial crack at what the data structures and actions/methods will be for your application.
  - You don't need the fanciest class diagrams but we do expect appropriate use of associations, including the multiplicity (0..n, 2, , etc) and whole-part (diamond) annotations; use the proper arrow for inheritance; and, include fields not otherwise shown in the diagram as well as methods (no getters/setters, just meaty stuff) in the class box. The design lecture notes give examples and list the features of UML you need to use.

- The diagram(s) should be reasonably complete in that associations and actions (aka methods) alluded to in the use-cases/storyboards should be shown in the diagram.
- Keep the diagrams from becoming so big and the edges so tangled that it is not readable. An unreadable diagram is a useless diagram. For instance, don't include obvious stuff like getter/setter methods.
- Consider diagramming each package/component separately so there are many small diagrams instead of one large diagram.
- UI designs are very much dependent on the framework used, and you likely will not need to diagram the front-end.

⊙ Describe the Architecture: give your full technology stack as well as any other APIs/frameworks you plan on incorporating.

- Include an architecture overview paragraph so we can see you have a basic understanding of the components and how they will interact.
- Include a picture of how your stack will appear in deployment if it is not standard.
- List all proposed frameworks and libraries you plan to use including web server, database, helper libraries such as JSON helper libraries, Google maps, Facebook API, etc.

⊙ Initial code

- For this iteration the goals are to get everyone with a good development environment, getting familiar with the development stack you are using, and writing initial code.
- If you want to write some prototyping code to test some library or framework you are using, put it in a special spot in your Github repository, for example in a prototyping/ directory.
- You are required to have at least something there, including a hundred (or more) lines of original code.
- Please add a sentence or two to the wiki under this heading describing your initial implementation.
- All members of the group need to be able to have the dev stack stood up on their own development platform.
- Everyone will also need to make at least one commit to the GitHub repo for the group. If its just a line in the readme so be it, but you need to have checked out the repo on your own computer and made the change there, don't just use the GitHub interface to edit.

Evaluation of Your Group Members

Each project group member is also required to confidentially evaluate the performance of other members of his or her group. The Markdown form to use is here, there are multiple copies of the form, fill out one for each team member. All peer evaluations will be held in 100% confidence.

Posting Instructions

As with requirements, make a GitHub wiki page called Design which includes the above information. The peer evaluation should be submitted to the "Peer Review 01 - Iterations 01 & 02" dropbox on moodle.