Project Proposal

Intelligent Review System v2

Group Members and Skills

- Jason Chan
  - 3rd year CS major
  - Java, javascript, python, html, css, sql
- Harrison Li
  - 1st year master's CS student
  - Python, C, Java
- Jessica Bishop
  - 2nd year CS major
  - Java, Python, JavaScript
- Sukhmai Kapur
  - 1st year CS major
  - Java, javascript, html, css

Problem and Solution

Problem

Visualization tools help users quickly gain an understanding of a data set. As of right now, ITS has no way to visualize all of its question data. In addition, the current difficult scores for each question were calculated from an old matlab script. We can update this script to incorporate new factors. We plan on visualizing data in terms of its difficulty, category, and student along with several different factors. Currently, the Intelligent Review System also depends on a static JSON file to receive its data. We plan on connecting our database to the front end to ensure dynamic data transfer.

Solution

We would like to create a web application. We would use Reactjs for our view and Flask for back end. We plan on using Flask to build the API for the database. From there, the front end would be able to make requests to the API to pull the data in real-time. We would implement data visualization using chartjs or d3 depending on ease of use. We would also write an algorithm to calculate difficulty based on information stored in the database.

Potential Issues/ Problems we might encounter

- Clear communication between front end and back end
- Real-time communication with database, ensure database interactions do not add latency that impacts user experience
- Database setup or integrating Flask framework with existing database

- If we are going to update difficulty based on real time student responses, need an efficient way to recalculate difficulty scores in order to update the table

Deliverables

By the final presentations we would like to have the following things implemented:

- Real time communication between the database and the front end
- Several working, useful analysis for the data to be presented to the user

Working Plan

Time Schedule	**Task**	**Responsibility**
Week 5-6	**1. Finalize project proposal** **2. Planning phase**	**Front End: Make prototypes for screens** **Back End: Determine important SQL queries**
Week 7	**Setup** **Milestone 1: Have a concrete plan to start addressing front end and back end problems. Have front end mockups (drawn) as well as backend scaffolding up in github.**	**Front End: Ensure we can use React. Make basic page** **Back End: Set up local database, coordinate with front end on interface of API and import ITS.**
Week 9-10	**Milestone 2: Front End: Have a visualization of a basic graph complete** **Back End: Have basic API Endpoints set up. Make sure database connection works and can be easily extended to satisfy front end requests.**	**Front End: Make basic graph** **Back End: Create API Endpoints**
Week 11	**Planning**	**Front End: Identify graphs to display** **Back End: Determine difficulty quotient calculation**

Week 12 - 15	**Milestone 3: Front End: Have more than one visualization of graphs along with an implementation of a filter Back End: Have working endpoints and examples of data being transferred in real-time**	**Front End: Create Graphs and Filters Back End: Test existing Endpoints and add any as needed, assist Front End as needed**
Week 16	**Milestone 4: Finish up any extraneous problems. Complete Final Presentation.**	**Divide work as appropriate**
Week 17	**Milestone 5: Complete documentation, wrap up the work // merge with master**	**Push final changes to github and merge**

Implementation Tools and Resources

- GitHub: https://github.gatech.edu/VIP-ITS
- W3Schools: https://www.w3schools.com/
- Project Documentation Notebook
- Spring 2019 Github: https://github.gatech.edu/VIP-ITS/IRS-v2
- https://flask-restplus.readthedocs.io/en/stable/
- https://github.com/PyMySQL/PyMySQL#documentation or https://docs.sqlalchemy.org/en/latest/orm/tutorial.html (depending on how we choose to interface with the database)