Project Proposal – DSCI 551

Group Members:

Wenjia Wang:

Background: Majoring in finance in undergraduate school. Currently majoring in applied data science.

Skills: I am knowledgeable about python and used to utilize python to complete financial data analysis projects including portfolio contribution and text analysis during my undergraduate courses.

Jianfei Xiao：

Background: Earned a Bachelor's degree in Computer Science with a focus on Artificial Intelligence from the University of Nottingham.

Skills: Skilled in programming languages such as Java and Python. Project experience includes leading the development of YOLOv5 models for object detection, with a focus on optimizing neural network architectures. Developed a Convolutional Neural Network-based system for Facial Expression Recognition, aimed at enhancing accuracy through feature extraction and model fine-tuning.

Zhongtian Ye:

Background: I was majoring in Econometrics and minor in Statistics in University of Illinois at Urbana-Champaign. Now I am majoring in Applied Data Science.

Skills: I mainly used R in my undergraduate classes to address cause effects within different economic scenarios by applying statistical techniques such as instrumental variables (IV), two-stage least squares (2SLS), regression discontinuity design (RDD) and difference-in-differences (DID). I also had some experience of using Python to address time-series questions in economic forecasting.

Project Requirements:

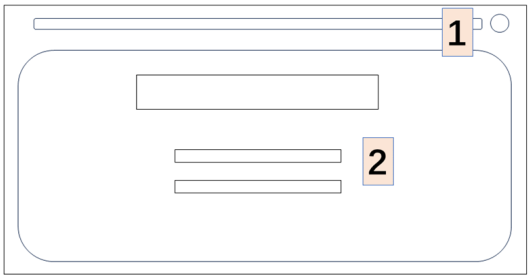
Our team needs to develop a web-based application that allows the users to filter, sort and search by keywords through the data stored in databases. For instance, if the user tries to book a train ticket from one place to another. By inputting the origin, destination and travel date to the application, it would display all the tickets on the specific date. By applying the filter function, it

would display one specific type of trains such as regular, express or high-speed railway trains. Simultaneously, sort function can be used to search for the trains departing from the earliest to the latest in one day or search for ticket prices from the lowest to the highest across a day. In that sense, it will be more convenient for the users to determine which train they could choose based on their needs.

Also, our team needs to develop a user interface which allows database managers to manage the data in the databases. For example, since there will be lots of trains departing in a day, a single database may be not enough to store all the data, thus it will be confusing for a database manager to look for one single instance in a huge database. Therefore, it is better to filter the dataset based on some conditions such as the type of trains or departing location across different regions in a country, and store them separately in two or more databases. Moreover, since the schedule of trains may change frequently due to rail construction work, weather conditions and number of customers, the user interface should have the function to make the database manager easily insert, modify and delete the data in the databases.

Planned Implementation:

Our team sketched this draft as a preliminary low-level prototype to illustrate our conceptual framework of our intended web application. Each element within the draft has been tagged for detailed explanation, providing a comprehensive overview of our design strategy.



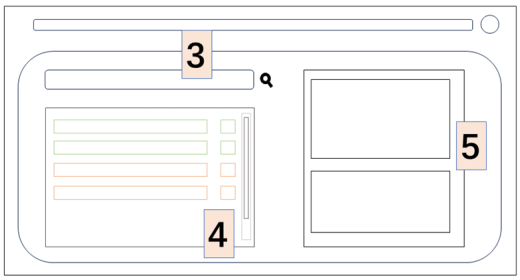
- **Tag 1**

  We plan to develop a web-application with Chrome on Windows OS. Given our team members' proficiency in Python, we selected Flask as our backend framework to ensure the seamless development process.

- **Tag 2**

  As mentioned in the project requirements, the quantity of the dataset has a substantial size. Thus, implementing a Log in System is essential to manage our user permissions effectively. Users will be categorized into "normal users'' with read-only access and "administrators" with rights to modify and delete. After the user type in the username and

the password, we will check the user permission as we will maintain a User table in a SQL database. Different User interface, which means the “Modified” and “Delete” options



buttons display or not, will be provided according to the permissions.

- **Tag 3**

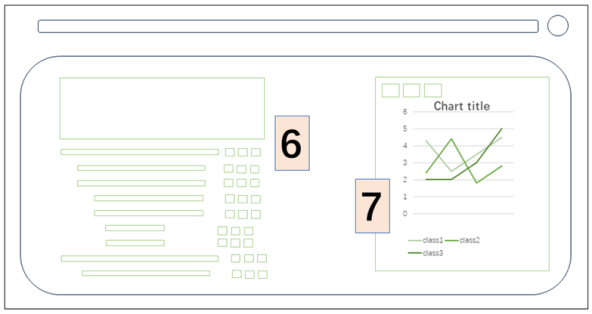
  After logging in to this system, the web will redirect to this interface. The search box supports the querying functions across different databases (Or tables). If we have enough time, we may try to build a likelihood search feature, otherwise just specific search. Search results will display databases matching the query criteria.

- **Tag 4**

  The List of the databases. We may use different colors to distinguish between the SQL(RED) databases and the NoSQL (GREEN) databases. The selection button on the right hand side of the name will lead the users to the next page (Here the figure 3 and 4), facilitating deeper interaction with the chosen database.

- **Tag 5**

  We want to provide some extra information to offer insights into the data's structure and scale. For example, the size of the datasets. There's kind of additional knowledge presented textually or graphically as appropriate.

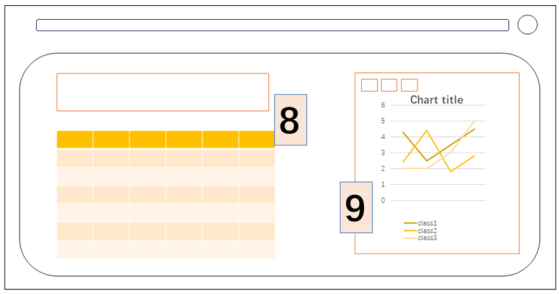


- **Tag 6**

Here we use Green to identify the selected NoSQL database. The JSON data will show line by line, administrators will have access to additional functionality, including options to modify or delete records, highlighted by three buttons alongside each record.

- **Tag 7**

  We also want to provide some visualization charts and graphs to give the users some intuitive feelings of the data. Thus, we may integrate some data visualization frameworks into the website to generate the figures.



- **Tag 8**

  Similarly, here we use Red to identify the selected SQL database. The tables displayed sequentially, with administrative functions available for record management.

- **Tag 9**

  Same as Tag 7 but possibly develop different types of visual representations to improve user interaction with SQL database information.

Timeline: *From 2.4 to 4.17, 10 WEEKS*

Project Timeline

Front End:
- Technology:
  - HTML + CSS + JavaScript
- Frameworks:
  - Vue.js, d3.js
- Expected Duration:
  - 1-2 weeks

Database:
- Technology:
  - SQL: MySQL
  - NoSQL: MongoDB
- Expected Duration:

        - 1-2 weeks

    Back End:

    - Technology:
        - Flask (Python)
    - Expected Duration:
        - 4-6 weeks

Overall Project Timeline:

- Front End + Database:
    - Estimated Duration: 2-4 weeks
- Full Stack Development (Front End + Database + Back End):
    - Estimated Duration: 6-10 weeks

Team Responsibilities:

Wenjia Wang:

I would pay my effort to cooperate with teammates in developing a web-based application with Flask including tasks such as setting up the framework, defining routes, creating templates, managing requests and responses, and connecting to a database. I will facilitate teamwork by organizing projects into directories, defining routes using decorators, and connecting Flask to a database of choice, such as MySQL or MongoDB.

Zhongtian Ye:

My task mainly focuses on designing the user interface in order to help database managers and end-users easily access the data in the database. I will use Flask and Shiny for Python to design an easily implemented and high efficiency user interface, create highly interactive visualizations, real time dashboards and sophisticated workflow apps. I would also choose the dataset that our team will be working on, which will be fit with the database that our team will construct and web application functions that we will develop.

Jianfei Xiao:

As the team leader, I will use my skills and experience in developing Convolutional Neural Network-based systems, as well as skills in programming languages such as Java and Python to coordinate the work of the two team members throughout the project and ensure that the project is progressing steadily. In terms of breakdown, I will be leading the back-end development efforts and focusing on the integration of Flask and database management, overseeing the design and implementation of Flask routing, aligning with the distributed database architecture, and designing and implementing user interfaces for database administrators and end-users.