HEALTHCARE PLATFORM : HEALTHY BUFFS

Team Members: Payoj Jain, Rahul Chowdhury, Ganesh Chandra Satish

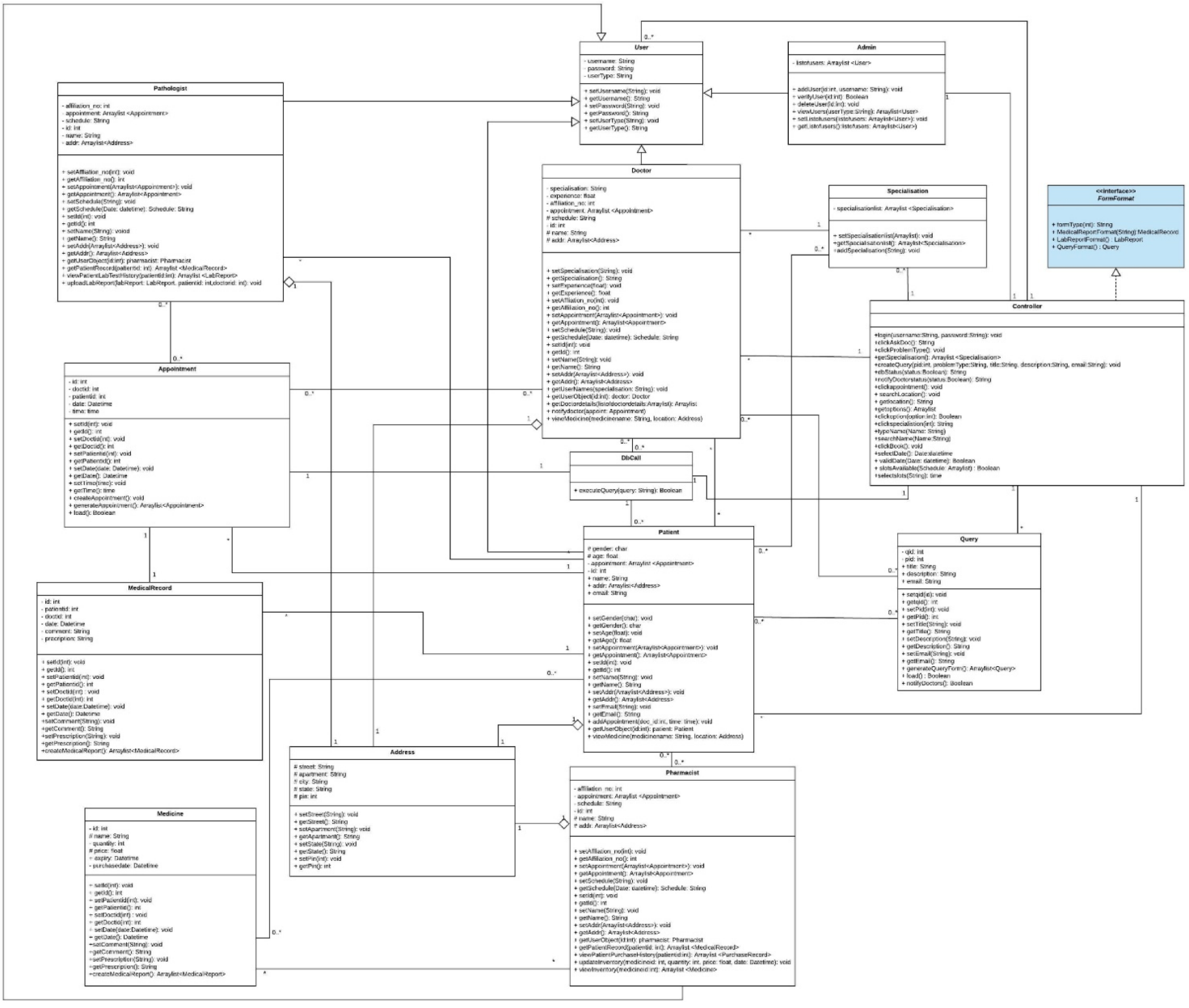
Team Number: 33

Title of the Project: HealthyBuffs

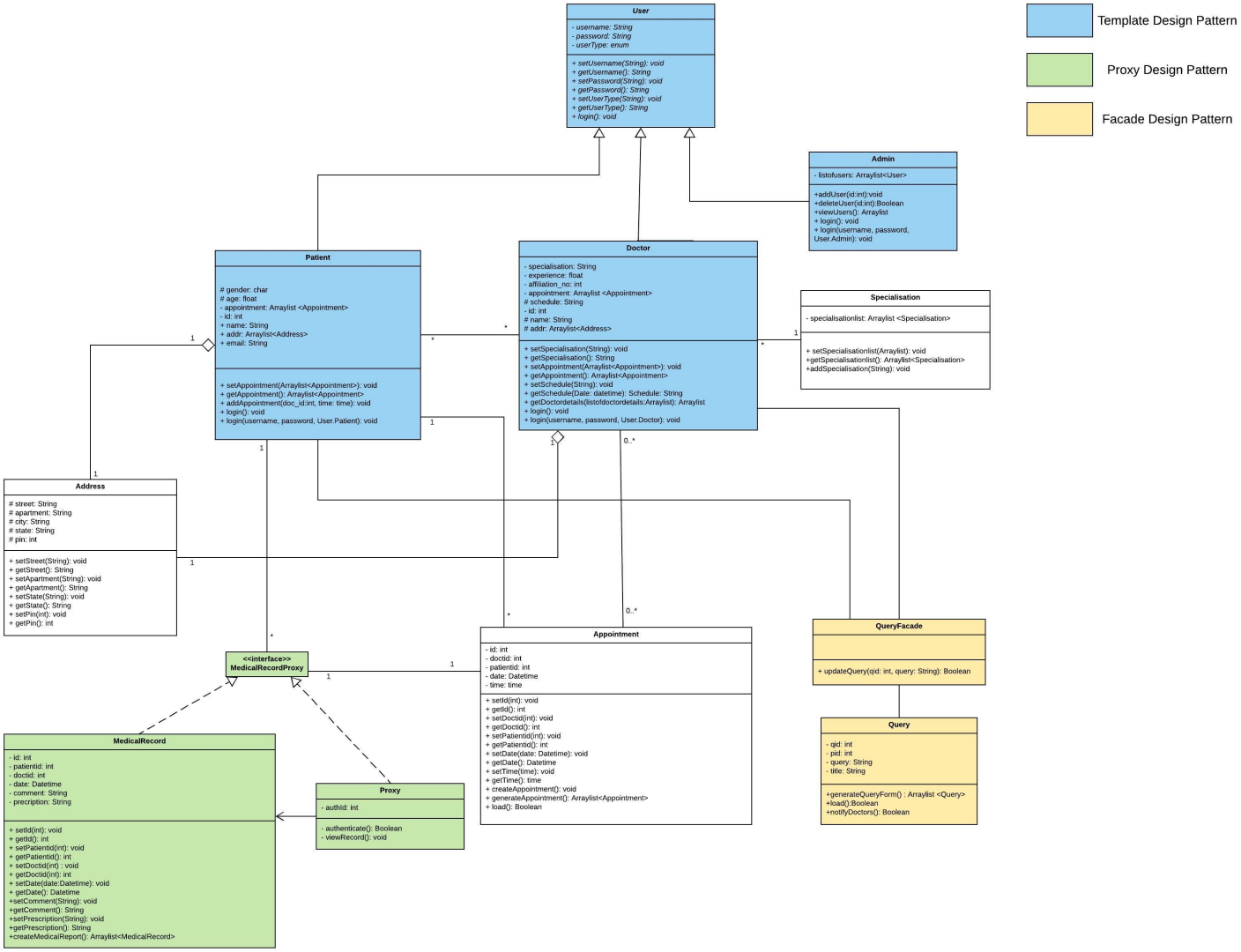
Vision: We are building a platform for CU members to search for doctors, pharmacies, pathologies etc. easily. This product helps people to book appointment instantly and look for medicine availability in nearby pharmacies. People can also consult doctors online anonymously by asking over the platform and get experts' answers.

Project Summary: HealthyBuffs is a healthcare management platform exclusively for students, faculties and other members of CU Boulder where people can locate doctors, pharmacists, pathologists, consultants and book appointments. This platform can also be used for looking up medicines availability in pharmacies. The portal also manages medical records/history of each patient so that it is easier for doctors to make their diagnosis and pharmacists can suggest medicines to patients.

Previous Class Diagram:



Completed Class Diagram:



Summary:

- Corrected the class diagram to include suggested changes.
- We have implemented following classes in Java
  - Query, Admin
- Implemented Design Patterns
- Created Completed class diagram for the implemented classes

Breakdown:

Rahul Chowdhury: Implemented class Query, Appointment

Ganesh Chandra Satish: Documentation, Implemented class Admin, Implemented Template Design Pattern

Payoj Jain: Implemented Proxy Design Pattern and Template Design Pattern

Estimate Remaining Effort:

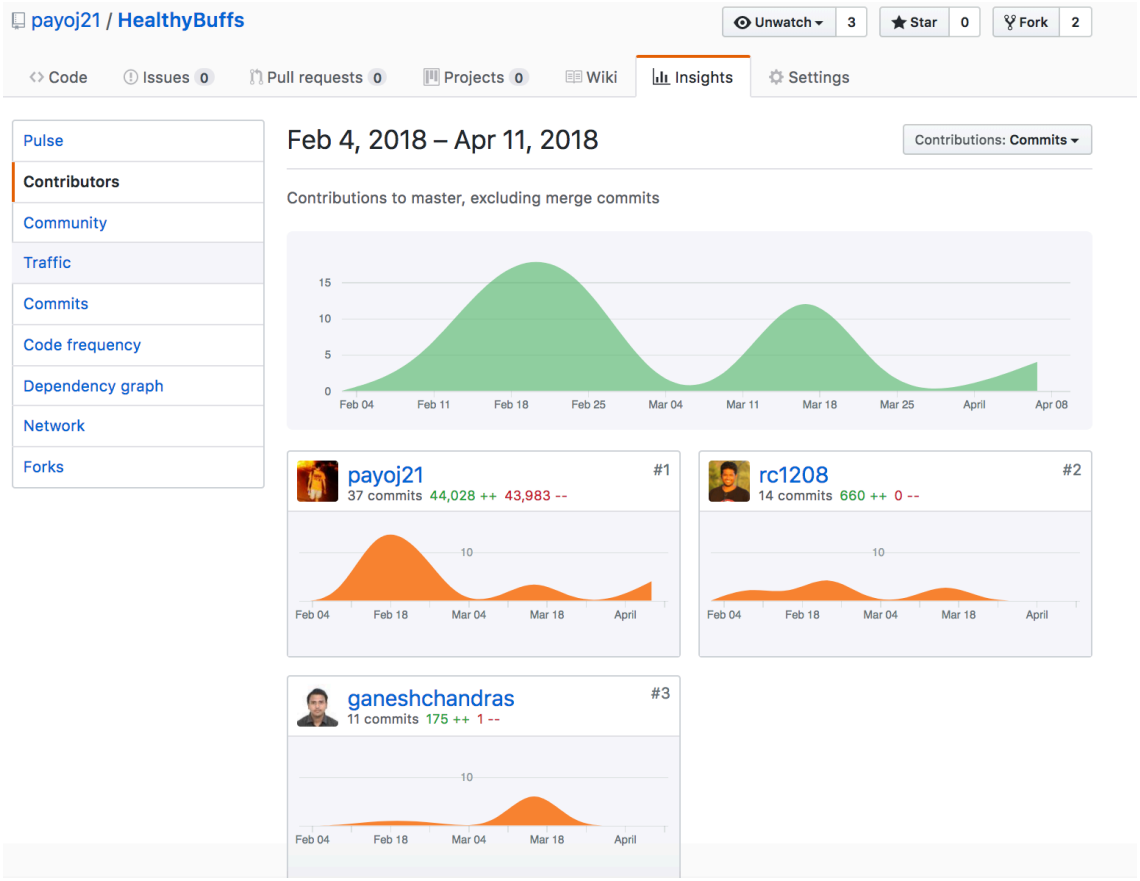
Estimated Hours Remaining:

Implementation of classes and design pattern: 10 hours

UI: 15 hours

Testing: 2 hours

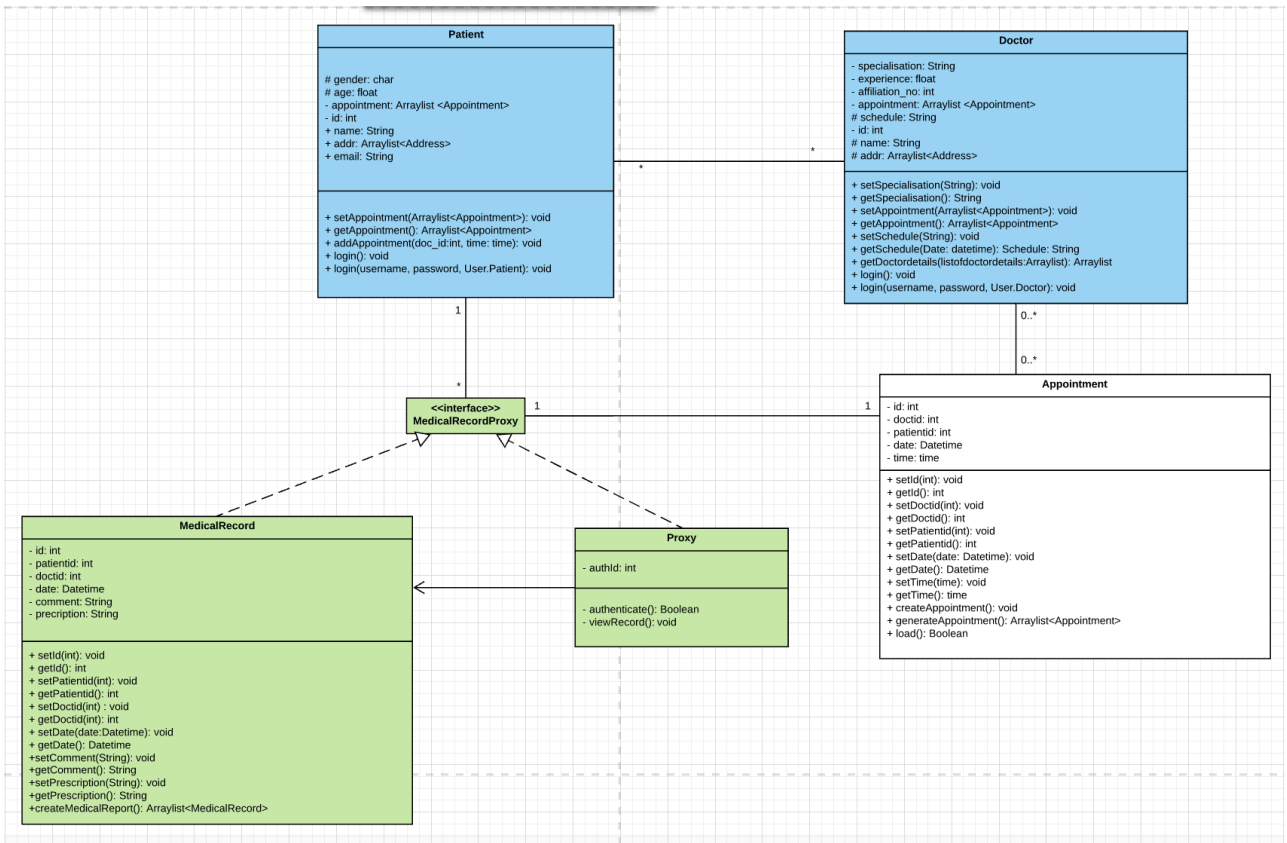
Github Graph:



Design Pattern

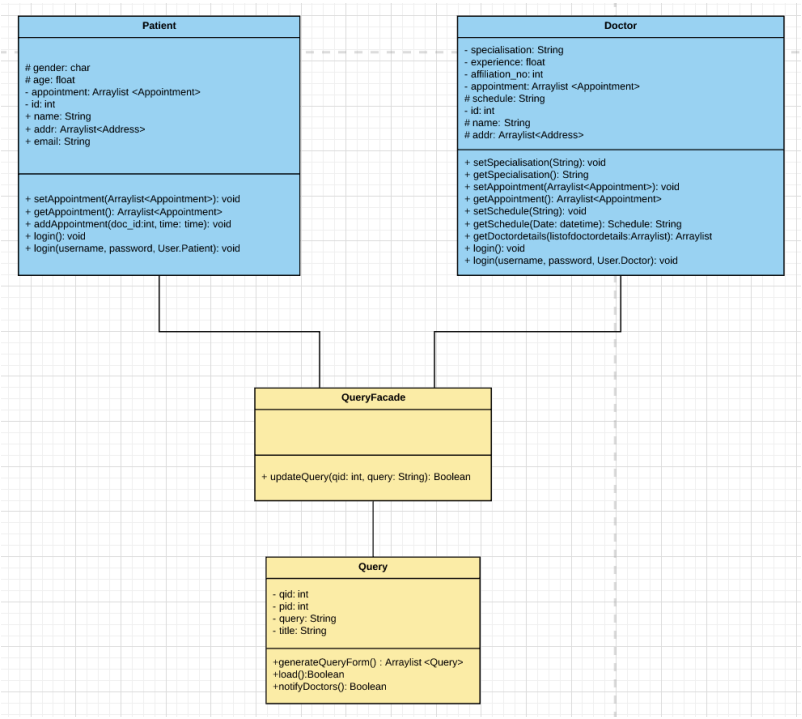
Proxy Design Pattern: To authenticate the doctors' and patients' access to the medical record. Only particular patients and doctors can access medical records of patients. To protect this access, we implemented "Protection Proxy". The presence of Proxy ensures that only authorized doctors can access the medical record.

Participants: Doctor and Patient



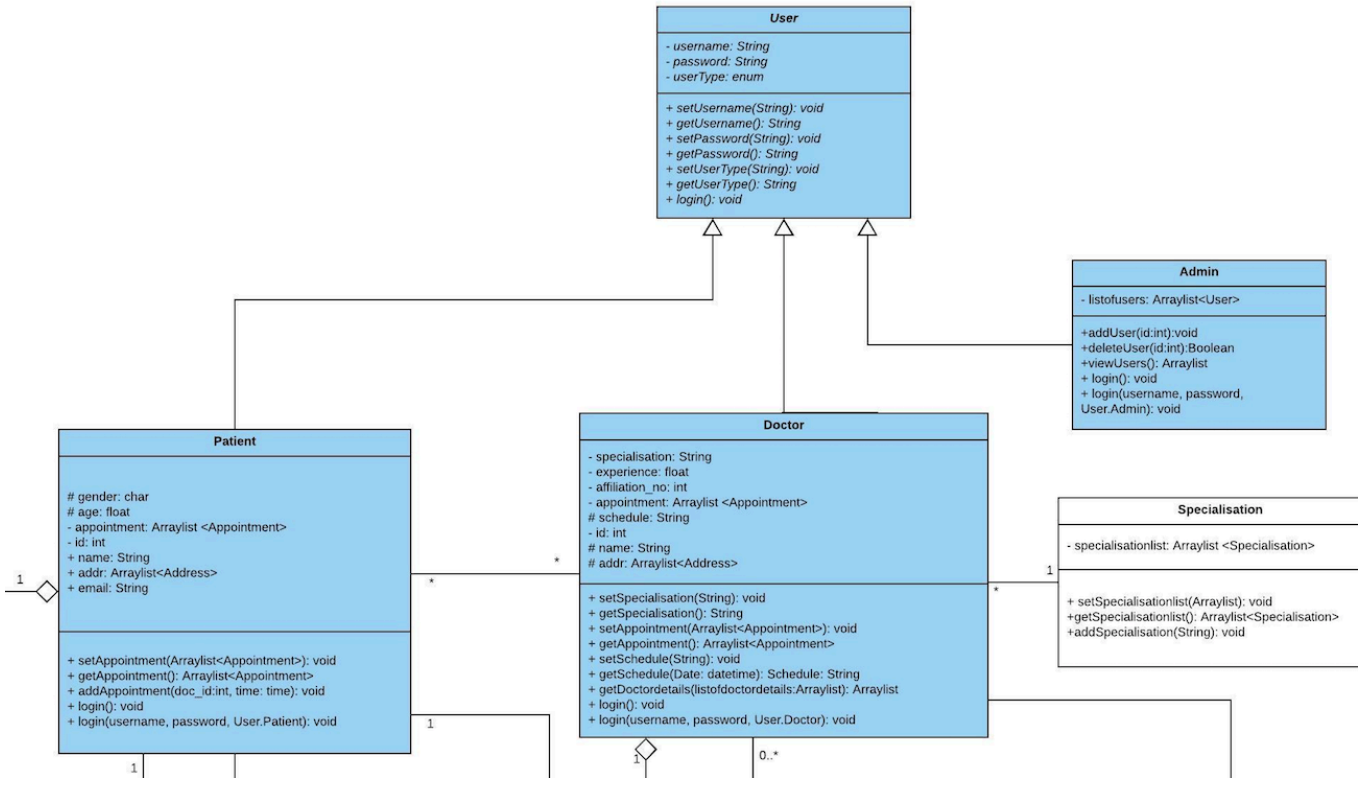
Facade Design Pattern: Query changes are common which should not affect the functionalities of the rest of the classes. For example, client code should have the same behaviour even if query system changes. Absence of Facade would cause changes in the query to be reflected on the Doctor and Patient where it should not. Hence, Facade is necessary to keep the functionalities of the Query independent from the other classes that use the Query class.

Participants: Doctor and Patient



Template Design Pattern: We currently have three types of users which are being handled by the User class. The steps involved in logging-in is similar except the login header which contains the information about the type of user logging into the platform.

Participants: Patient, Doctor, Admin



Final Iteration:

We are planning to implement the rest of the classes with as much functionality as possible. Also, we will be testing out current code on some sample users and try to fix bugs.

We have already setup the important and necessary steps required in the implementation.

A. Implementation of the classes:
B. User Interface
C. Unit testing