CSCI 5448 –
Project Report
MyHealth_Services

Team: Aaresh Bachana
Anish Churi
Harshil Sheth

1. **What features were implemented?**

- BR – 002: All users must complete basic profile with all mandatory information
- UR – 001: User can sign-up to create a profile
- UR – 002: User can log in
- UR – 003: User can view profile
- UR – 006: User can search for doctor by specialty or name
- UR – 007: User view a doctor's schedule
- UR – 008: User can view and edit his schedule
- UR – 009: User can book an appointment with a doctor
- UR – 010: User can view his test results
- UR – 012: User can view Medical Bills
- UR – 014: User can request for a test and the results will be uploaded to his profile
- FR – 001: System should verify the validity of User-Id and Password
- FR – 002: System should update the doctor's schedule once an appointment is made to avoid appointment clashes
- FR – 004: System should pop up a notification message indicating the test results are uploaded
- FR – 006: System should pop up a notification message indicating the medical bills are uploaded
- NR – 001: User should be directed to homepage after login within a timeout period
- NR – 002: User should restart the system in event of a failure
- NR – 003: User should be notified about the confirmation of his appointment within a minute after scheduling an appointment

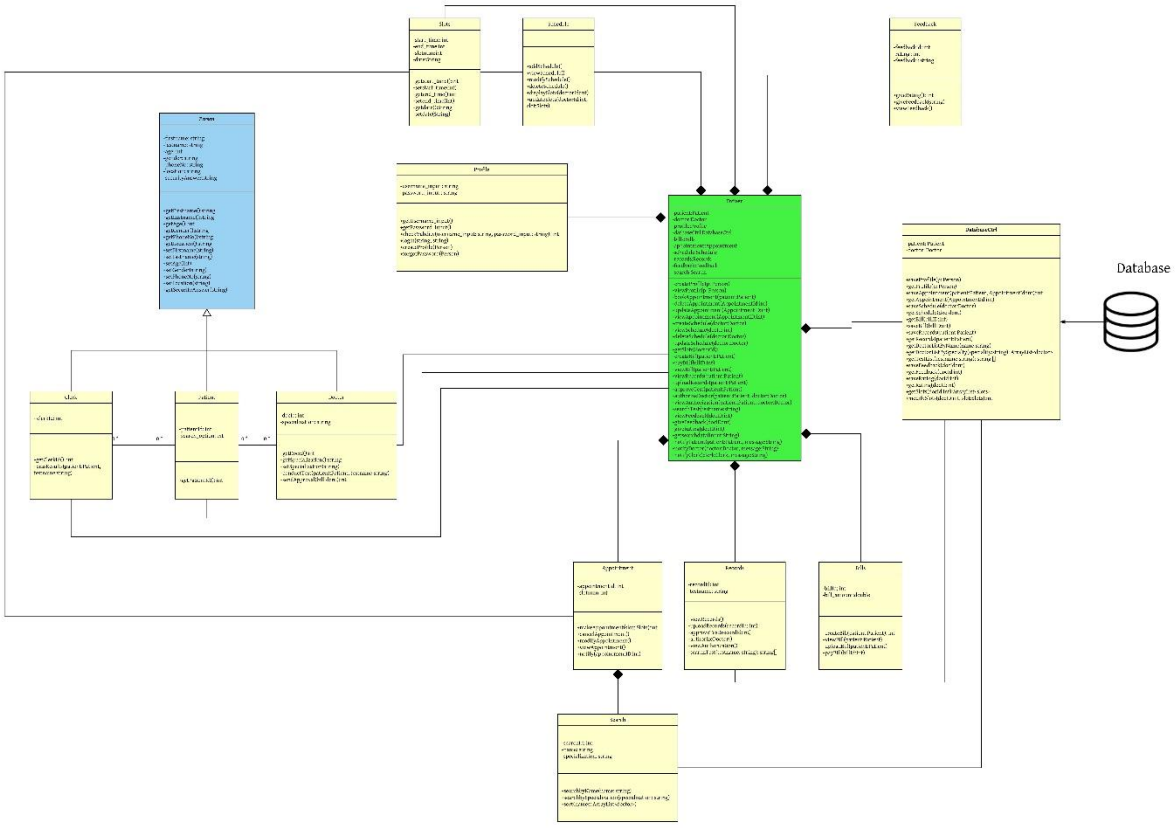
2. **What features were implemented?**

- BR – 001: All login User-Ids must be of format lastName.firstName#, where # is any number
- BR – 004: Send Reminders to Users a day prior to appointment
- BR – 005: Users can reschedule/cancel an appointment a day prior to scheduled appointment
- UR – 004: User can view his Medical Records
- UR – 005: User can authorize a doctor to view his medical records

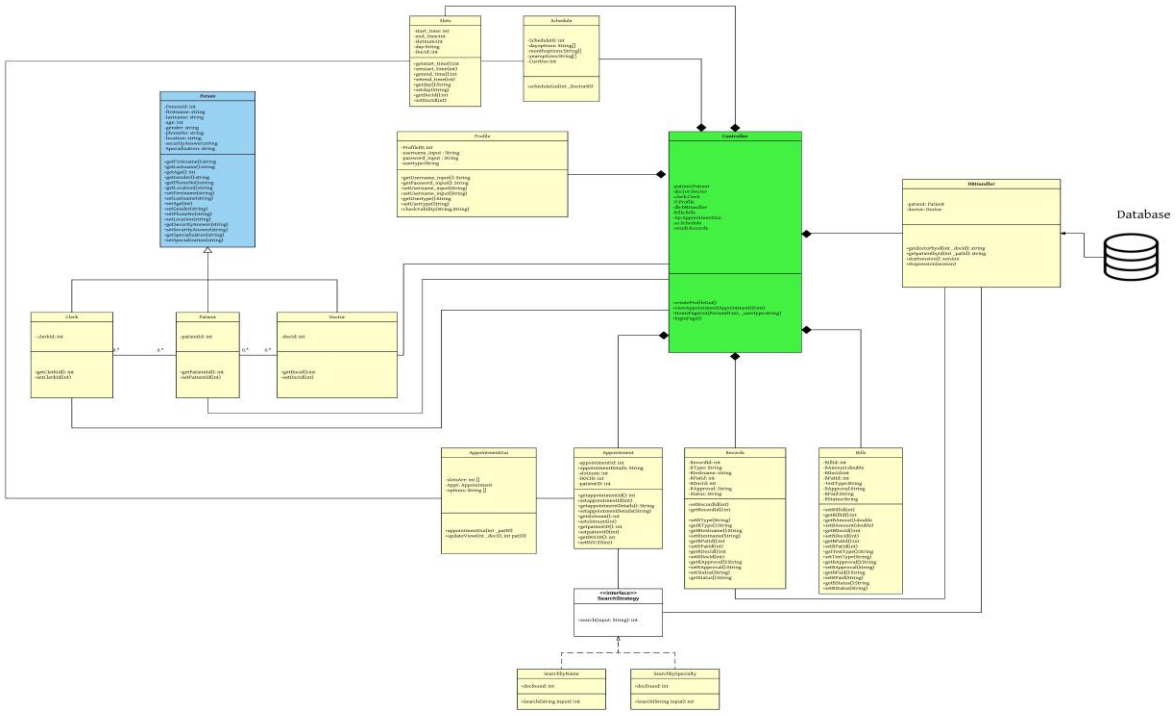
- UR – 011: Users can reschedule/cancel an appointment
- UR – 013: User can view Patient's medical history if authorized
- UR – 015: User can give additional feedback and rate the doctor
- FR – 003: System should ask for a security question in case of a user forgets his/her password
- FR – 005: System should update the user medical record with the latest test results
- NR – 004: Patients data should be kept private and secure
- NR – 005: System admin can maintain the system

3. **Show your Part 2 class diagram and your final class diagram. What changed? Why? If it did not change much, then discuss how doing the design up front helped in the development.**
   Original Class Diagram:

   For a closer look of this diagram, refer: **Class Diagram**



Updated Class Diagram:
For a closer look of this diagram, refer: **Updated Class Diagram**



Thus, if you notice the two diagrams, we tried our best to stick to most of the sections proposed in the initial class diagram. This helped in speeding up the implementation process because the class diagram provided a good framework of the entire system and their interaction with each other. Throughout the development process, we ensured that the classes are designed using a "Low Coupling, High Cohesion" principle and this was made easier but constructing a detailed class diagram. This also helped us implement most of our proposed functionalities.

The primary functions remained unchanged, but to accommodate all the functionalities, we had to make some changes to the initial design. As an attempt to refactor the code, we introduced the DBHandler class that we used to get doctor and patient information by ID. This was done because we had to scan for information using IDs several times in our code. Another class we added was the appointmentGUI which was closely related the appointment class. We tried our best to stick to the MVC design pattern for code design and this helped in faster code development.

4. Did you make use of any design patterns in the implementation of your final prototype? If so, how? If not, where could you make use of design patterns in your system?

The analysis of the project beforehand and creating class diagrams before even starting the implementation helped a lot with the implementation. We did a good job by identifying various models, views and controller. However, when we started with the implementation we realized that we could have used other design patterns and successfully incorporated few of those.
The following are the design patterns that have been used:

1. **Strategy Design Pattern:** The strategy design pattern was implemented by using two different algorithms for the searching mechanism. The two algorithms Search by name and Search by specialty were implemented depending on the checkbox criteria. The search strategy interface was implemented by both the classes. Even though the application of the strategy design pattern was not required it was a good practice for us to implement in the project**.**

The following are the design patterns that could have been used:

1. **Iterator Design Pattern:** This design pattern can be implemented to iterate through the large list of doctors and patients.
2. **Proxy Design Pattern:** This design pattern could be used to write a wrapper class so that the client code need not change even if there are implementation changes and also add extra functionalities.

5. What have you learned about the process of analysis and design now that you have stepped through the process to create, design and implement a system?

The overall design and implementation of this project helped us understand the software development cycle in detail. We have studied the concepts of SDLC theoretically in several other courses. But this course helped us understand the entire software development cycle better because of its practical approach for teaching. We realized the importance of software design before beginning implementation and also the importance of an object oriented approach that allows code reusability and modification, better understandability and reduced chances of erroneous implementation.

The quizzes helped us prepare all the concepts thought in class before we began serious implementation of our project idea. Activity – 2 for the project, where we decided our requirements, created activity diagrams to incorporate these requirements and created class and state diagrams helped us build a solid roadmap to plan our project flow.

We used Hibernate to be used as an ORM for the MySQL database and since it uses annotations to generate SQL statements. This saves a lot of effort and time for development of the project source code. Thus Hibernate provided a major advantage over using other normal databases and using specific SQL statements separately.

Lastly, we took additional efforts to identify code reuse and code smells in our implementation and we worked on refactoring our code using methods taught in class. We also realized the importance of refactoring the code throughout the development process and not in the end.