PROJECT-7 | CSCI 5448 | Object Oriented Analysis and Design

Project – Boulder Apartment Finder

Team Members

We are a team of three:

- Divya Pragadaraju
- Mukund Kalantri
- Shreyas Kapoor

A little about this Project:

The main objective of this project is to create a website that can find the availability of apartments in Boulder. Taking the idea from popular websites like Zillow, apartmentList, or our own Ralphie List, we plan to learn how to design and maintain any website end to end.

Final Statement on the project

This project “Boulder Apartment finder “turned out to be exactly as we wanted it to be. It runs on the local host.

Features implemented – Initially, we had projected that we would implement 8 use cases as follows:

- Sign-up user - User can sign up and enter their information in the system.
- Login user - User can log in into the system
- Edit-details - User can edit their information in the system.
- Apartment details- User clicks on an apartment widget and the screen shows details about the apartment
- Book Reservation- User can book a reservation to visit the apartment
- Cancel Reservation - User can cancel a reservation to visit the apartment
- Review apartment - User can write a review for the apartment
- Delete Review- User can delete a review that they have posted

Of All these use cases, we have implemented the first 7 of them. We have not implemented the “delete review” use case because we realized that it wouldn't be very crucial. Once the review has been submitted, it wouldn’t make much difference to delete it.

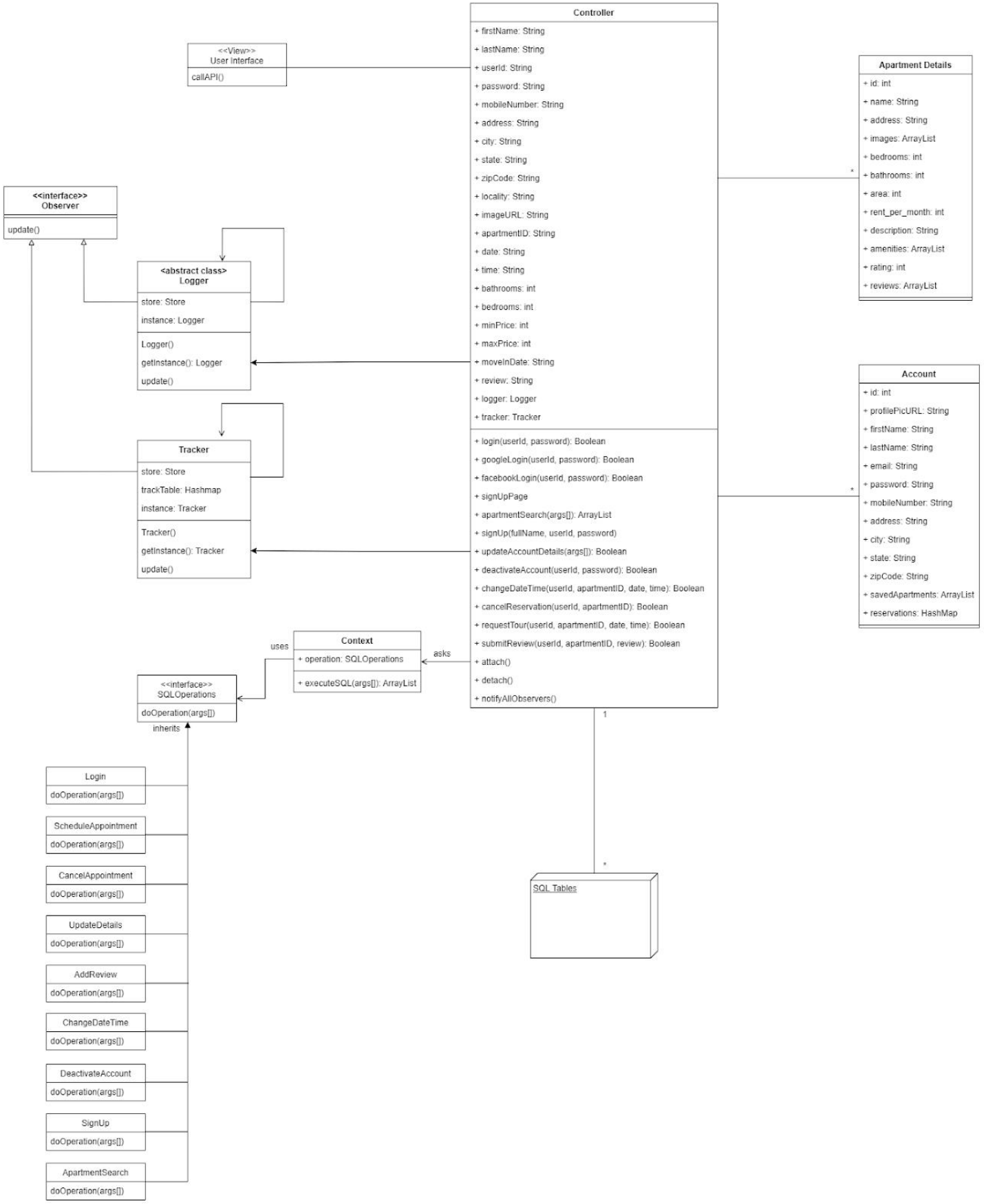
For storing data, we proposed that we would require four types of tables to store data and we have successfully implemented these tables:

- User_tbl- this table stores information of each user
- Apartment- this table contains all the details regarding
- Appointment - This table contains all your scheduled appointments
- Review – This table stores all the reviews a user gives.

Class Diagram

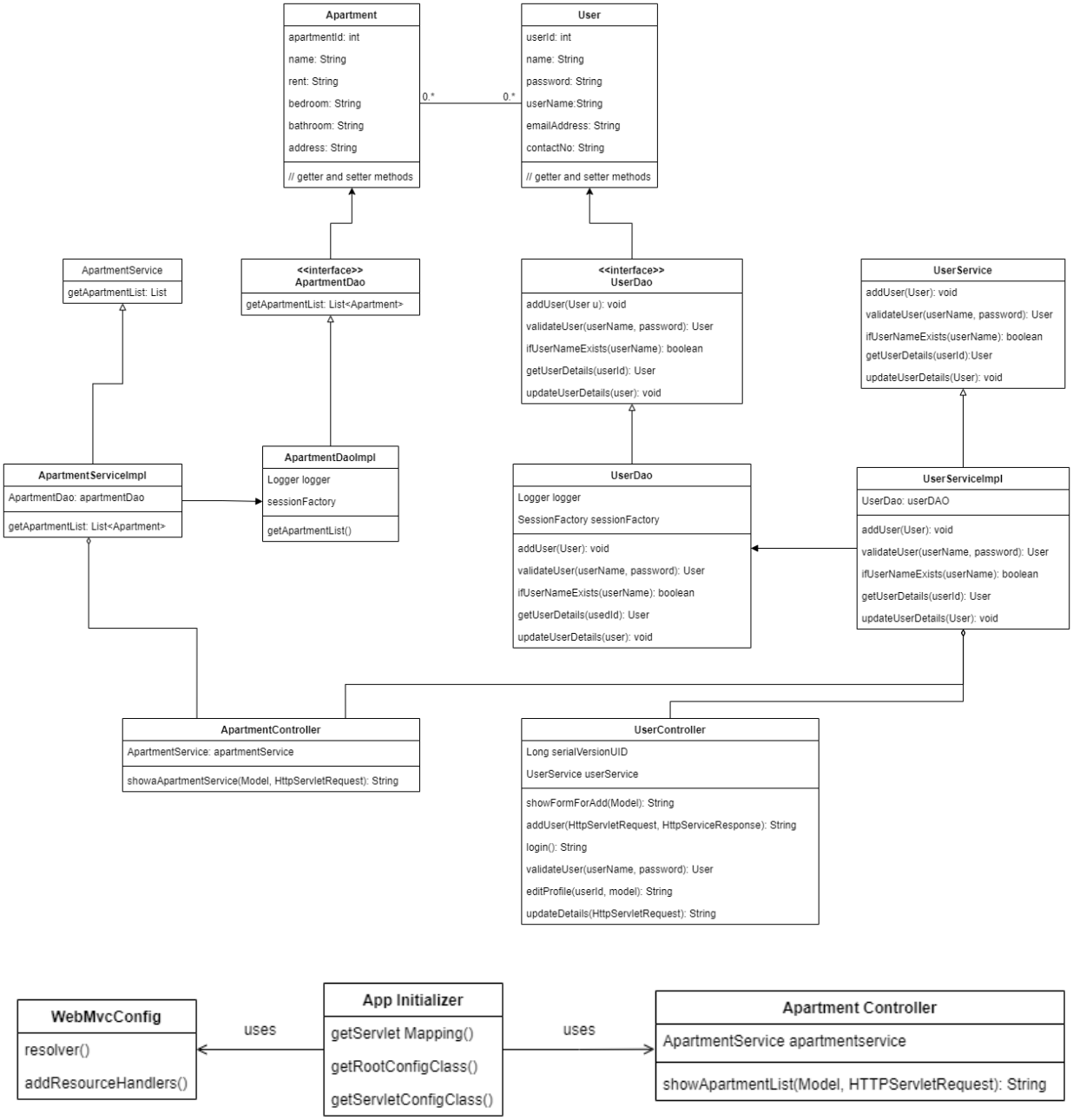
While implementing this project, we have made use of MVC pattern which stands for Model view controller pattern wherein the model part consists of application data, the controller for processing the user requests and building proper models while for View, we will have a UI screen for the web application from where users will interact. All this is done with the help of spring MVC Architecture. For implementing the functionality of the controller, we would want to use the factory pattern. Further, we are maintaining a logger file which will store all the actions taken on the application per day. We also have a tracker where we will maintain a list of schedules for apartment tours. All actions taken by any user will generate a notification that would be sent to both. It is important to have one logger file for each day for all the users and one tracker file throughout the life of this application.Thus, making use of observer pattern in this process. Following, we have different actions that would be performed on the SQL Databases. For this, we have a class for each type of action, and what command needs to be executed would be handled by Strategy. Also, we have different actions that would be performed on the SQL Databases thereby code refactoring with SQL would be handled with singleton.

From Project-5



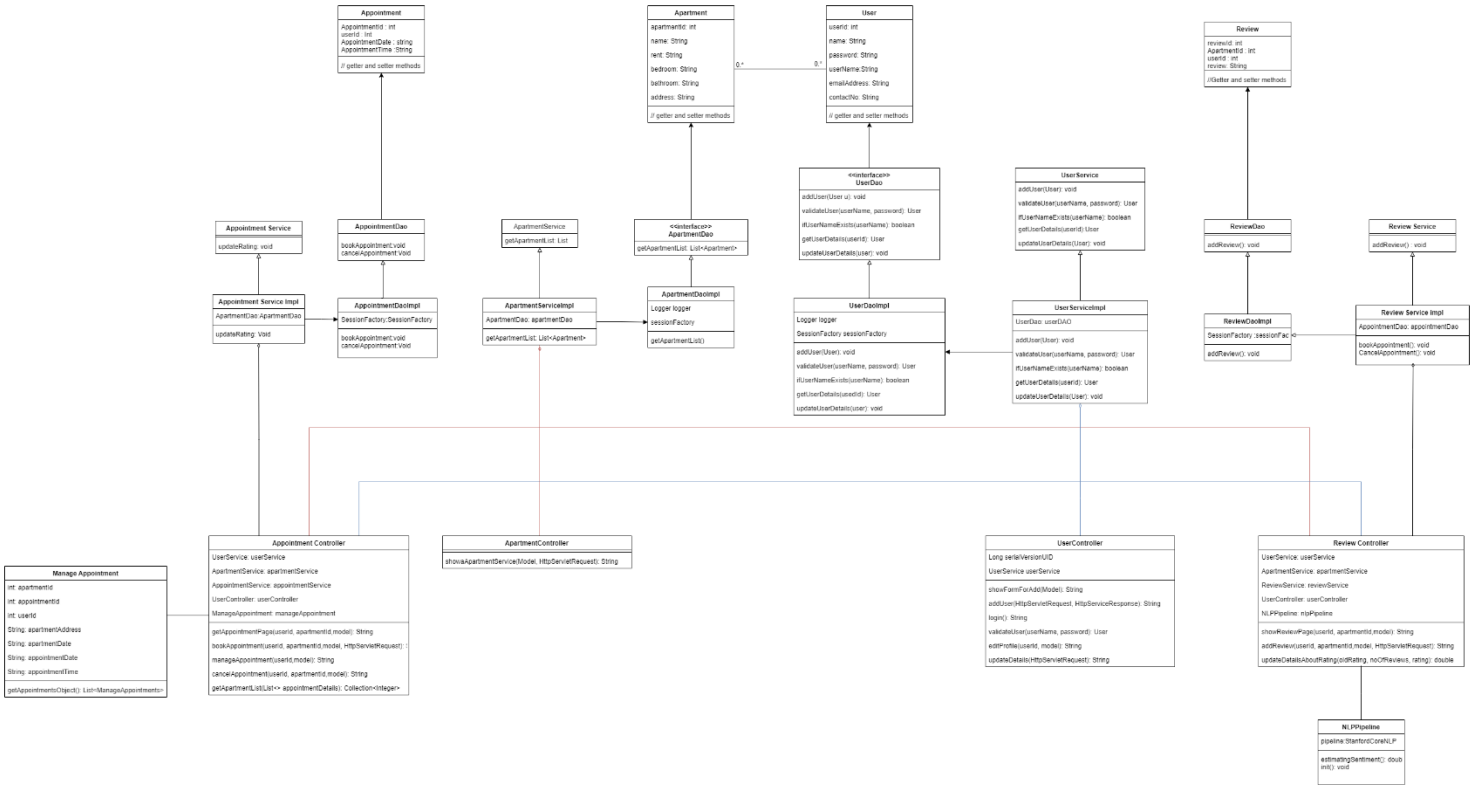
From Project 6

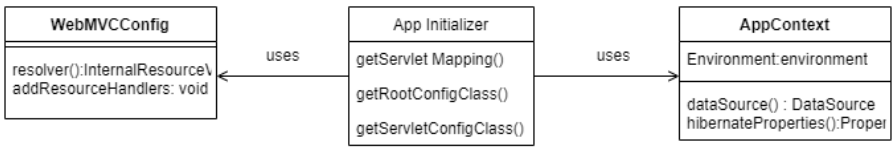
By this point, there are a few changes that we made as we are adhering to the ideas and approaches, we planned in the initial phase of this project except for wanting to use a factory pattern while implementing the functionality of the controller which is responsible for processing the requests of users while building appropriate models and finally pass them to the view for further processing instead of using strategy in this project.



From Project 7

There have been quite a few changes by the end of our project. We added Service, Dao, and Controller classes for Appointment and Review functionality. These classes have also been connected with the already existing classes. We are calculating rating using the semantic score of a review using NLP-based Libraries. We modified our WebMCVConfig, and AppContext classes a little bit.





Third Party Code Vs Original Code

For the initial setup of the project structure, we have taken reference from the following link

https://www.javaguides.net/2019/12/spring-mvc-crud-example-with-hibernate-jsp-mysql-maven-eclipse.html

To take reference about html, we have used - https://www.w3schools.com/html/

For CSS have referred to https://www.w3schools.com/css/

For JavaScript we referred to https://www.w3schools.com/w3js/

We also have used stack overflow to clear a few of our doubts.

Although we have taken few references from around the web for learning and understanding purposes, All the code that is written in this project is original.

Statement on the OOAD process for our overall Semester Project

1. The first issue that we faced was implementing the basic project structure. We had never used the combination of Spring Boot, Hibernate, SQL, Tomcat and Java. It took us almost 2 days to figure out how we should run the basic code on our machines.
2. During this process, we learned the importance of different versions of libraries and how some of the newer libraries were not compatible with older versions. Beyond that, we had to download some software like MySQL and tomcat and figure out how to run war files in the Tomcat server. The overall process was truly knowledgeable, and we now feel confident in running any java web project on our local machines.
3. Through this project, we have had a clearer understanding of how design patterns can be made use of in real time projects in a convenient and effective way. We could see how flexible and adaptative these patterns are toward changes that we encountered in each phase of this project.