

Healthcare Appointment Management System

Gabriel Sher, Kosty Arnouk, Mahmoud Fawaz, Karson Or, Aroha Upreti

Department of Engineering, University of Ottawa

SEG 2105: Introduction to Software Engineering

Prof. Hussein Al Osman

December 4, 2023

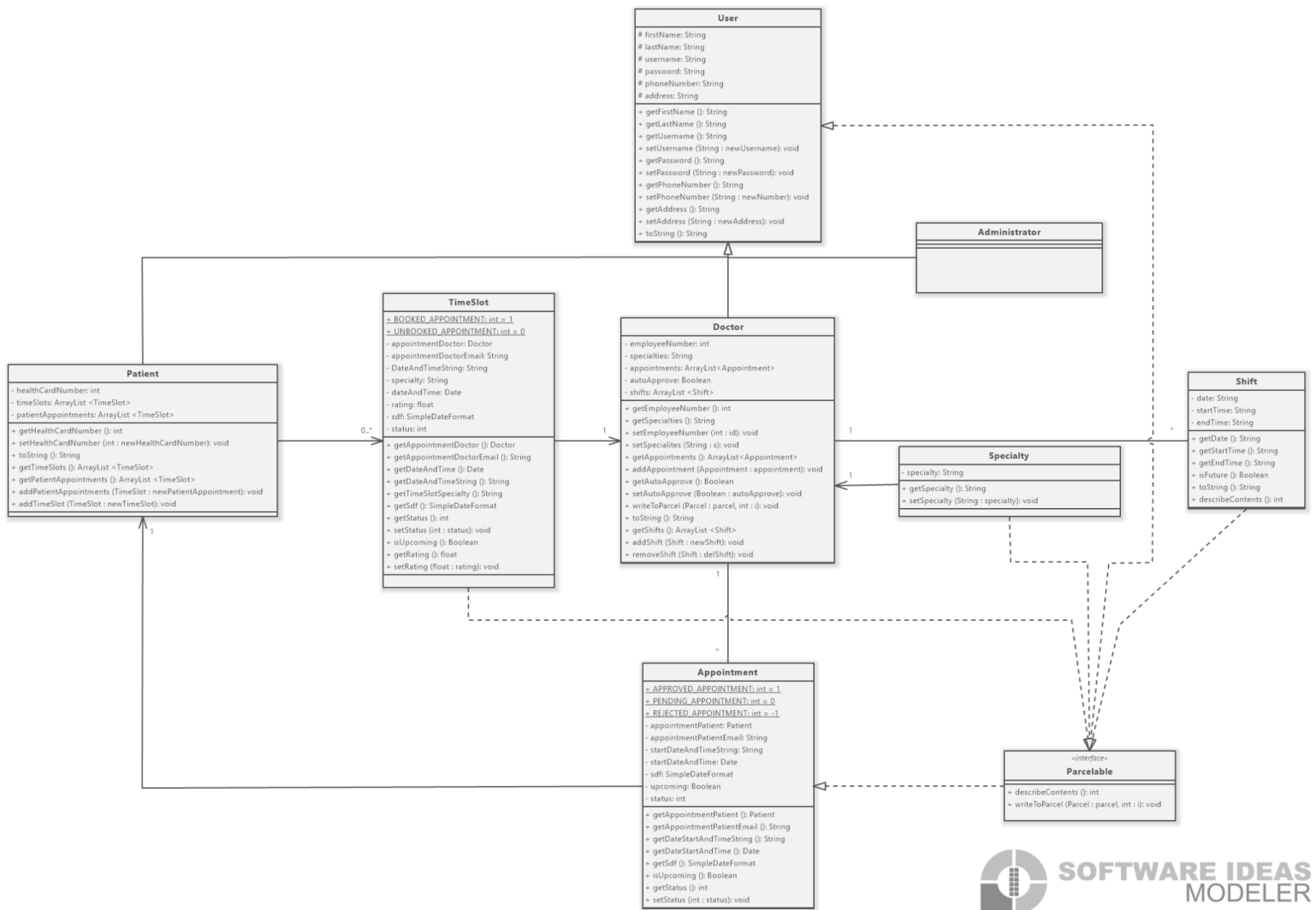
Introduction

For this project we had to create a Healthcare Appointment Manage System app. The app would support three different kinds of users that all had their own unique functionalities. Users would be able to create accounts as either doctors or patients and would be able to login to access features that were unique to that type of user. There was also an admin account that was responsible for accepting and rejecting new accounts that had been created, either allowing or preventing that user from logging in. The functionality for the doctors and the patients were relatively similar with some important differences distinguishing the two.

Doctors would be able to create shifts that could be cancelled so long as there were no appointments during it. They could also see past appointments and their upcoming appointments which they could reject or accept. With the press of a button they could also allow the app to auto-approve any newly created appointments so that they would not have to do so manually. Cancelling an appointment would allow other (or the same) patients to book another appointment during the previously occupied timeslot.

Patients are able to book appointments during available time slots, designated by the upcoming shifts of doctors. They are able to search for doctors by specifying the specialty and they will be presented with all of the available appointment time slots with that doctor. After they have booked appointments, they will be available to view in the upcoming appointments page where a patient can view all of the relevant information such as the date, time and name of the doctor associated with that appointment. Users can also cancel an appointment as a patient, but they will not be allowed to do so if they are within 60 minutes of the appointment start time. After an appointment has already passed a patient can view their past appointments and rate their experience with their doctor. Those are all of the main functionalities that needed to be implemented in this project.

UML Diagram



Contributions of Team Members

Deliverable 1:

Feature of Task	Team Member
A user can create a Patient or Doctor account.	Gabriel
The Administrator, Doctor, or Patient user can see the “welcome screen” after successful authentication. The welcome screen specifies the user role.	Kosty + Karson + Mahmoud
The user can log off.	Gabriel + Mahmoud
All fields are validated. There are appropriate error messages for incorrect inputs.	Karson
(BONUS) The group uses a DB	Aroha
UML Diagram	Aroha

Deliverable 2:

Feature of Task	Team Member
When a Patient or Doctor registers, their account information is stored in the DB (along with an indicator of whether their account registration has been approved, rejected, or not processed yet).	Aroha
The Administrator can view the list of registration requests.	Gabriel + Aroha
The Administrator can view the information associated with each request (the information the user entered during registration).	Gabriel
The Administrator can approve or reject a registration request.	Gabriel + Aroha
If approved, the registration request disappears from the list of registration requests.	Gabriel

If rejected, the registration request is added to the list of rejected registration requests.	Gabriel
The Administrator can view the list of previously rejected registration requests.	Gabriel
The Administrator can approve a previously rejected request.	Gabriel
The registration requests are stored in the DB.	Aroha
When a Patient or Doctor attempts to login, they are either directed to the welcome page, notified that their registration request was rejected, or informed that their registration request has not been processed yet.	Aroha
UML Diagram	Aroha

Deliverable 3:

Feature of Task	Team Member
The Doctor can view a list of upcoming appointments.	Gabriel
The Doctor can view the information of a Patient that requested an appointment (i.e., first name, last name, email, address, phone number, health card number).	Gabriel
The Doctor can approve or reject an appointment request.	Gabriel
The Doctor can cancel a previously approved appointment.	Gabriel + Mahmoud
The Doctor can view a list of past appointments.	Gabriel
The Doctor may enable a setting so that all appointment requests are automatically approved by the system without further action on their part.	Gabriel + Aroha
The Doctor can view the list of upcoming shifts they are working.	Karson

The Doctor can add a new shift by specifying the date, start-time, and end-time of the shift. All fields must be validated. Hence, the Doctor cannot enter a date that has already passed or a shift that conflicts with another one they had previously added.	Karson
The Doctor can delete an existing shift.	Karson
UML Diagram	Aroha


Deliverable 4:

Feature of Task	Team Member
The 4 Unit test cases (simple local tests) are implemented. There is no need to include instrumentation or Espresso Tests (UI).	Gabriel
The Patient can view a list of upcoming appointments.	Kosty + Gabriel
The Patient can cancel an upcoming appointment. Cancellations should only be possible if the appointment is not scheduled to start in the next 60 minutes.	Mahmoud
The Patient can view their past appointments.	Gabriel
The Patient can rate a Doctor with whom they previously had an appointment.	Gabriel + Mahmoud
The Patient can search for appointments by specifying a medical specialty and selecting a time slot from the available ones. An appointment is a 30-minute time slot.	Mahmoud + Karson
Once booked, the appointment appears in the Patient's list of upcoming appointments	Gabriel + Kosty + Karson
A booked time slot is not listed when Patients look for appointments.	Mahmoud
The Doctor cannot delete a shift if it is associated	Gabriel + Kosty

with one or more Patient appointments.	
Report	Gabriel
UML Diagram	Aroha

Screenshots

Login Menu:



example@email.com

Password

Sign In

New User? Register

Which type of account would you like to create?

Patient

Doctor

Patient Signup

First Name

Last Name

example@gmail.com

Password

Phone Number

Street Name, Number, City, Province

Health Card Number

Join Now

Doctor Signup

First Name

Last Name

example@gmail.com

Password

Phone Number

Street Name, Number, City, Province

Enter 1+ Specialties (Seperate with ',')

Employee ID

Join Now

Admin Screens:

Logout

View Pending Accounts

View Accepted Accounts

View Rejected Accounts

Return

Full Name: Test Test Phone Number: 1234567890
Email: test@gmail.com Employee Number: 123
Address: 51 x.s Specialties: apple

Full Name: dtest dtester Phone Number: 6138936565
Email: dtest@gmail.com Employee Number: 8838
Address: hills,123,ottawa,ontario Specialties: surgeon

Accept Reject

Return

Full Name: changes hi Phone Number: 6134940099
Email: change@gmail.com Health Card Number: 1234567890
Address: hi,5,ont,ot

Full Name: Patient Patient Phone Number: 7055555555
Email: dulted@juottawa.ca Health Card Number: 1234567890
Address: To St, 1, Ottawa, ON

Full Name: Aroha Upreti Phone Number: 8199196099
Email: uaroha@gmail.com Health Card Number: 1234567890
Address: Unversity Street, 120, Ottawa, Ontario

Full Name: Test Patient Phone Number: 0000000000
Email: test@patient.com Health Card Number: 9
Address: up, 1, djb, shaps

Full Name: Gabe Test Phone Number: 4664676766
Email: patient@testing.com Health Card Number: 987654321
Address: also, 1828, silek, deowo

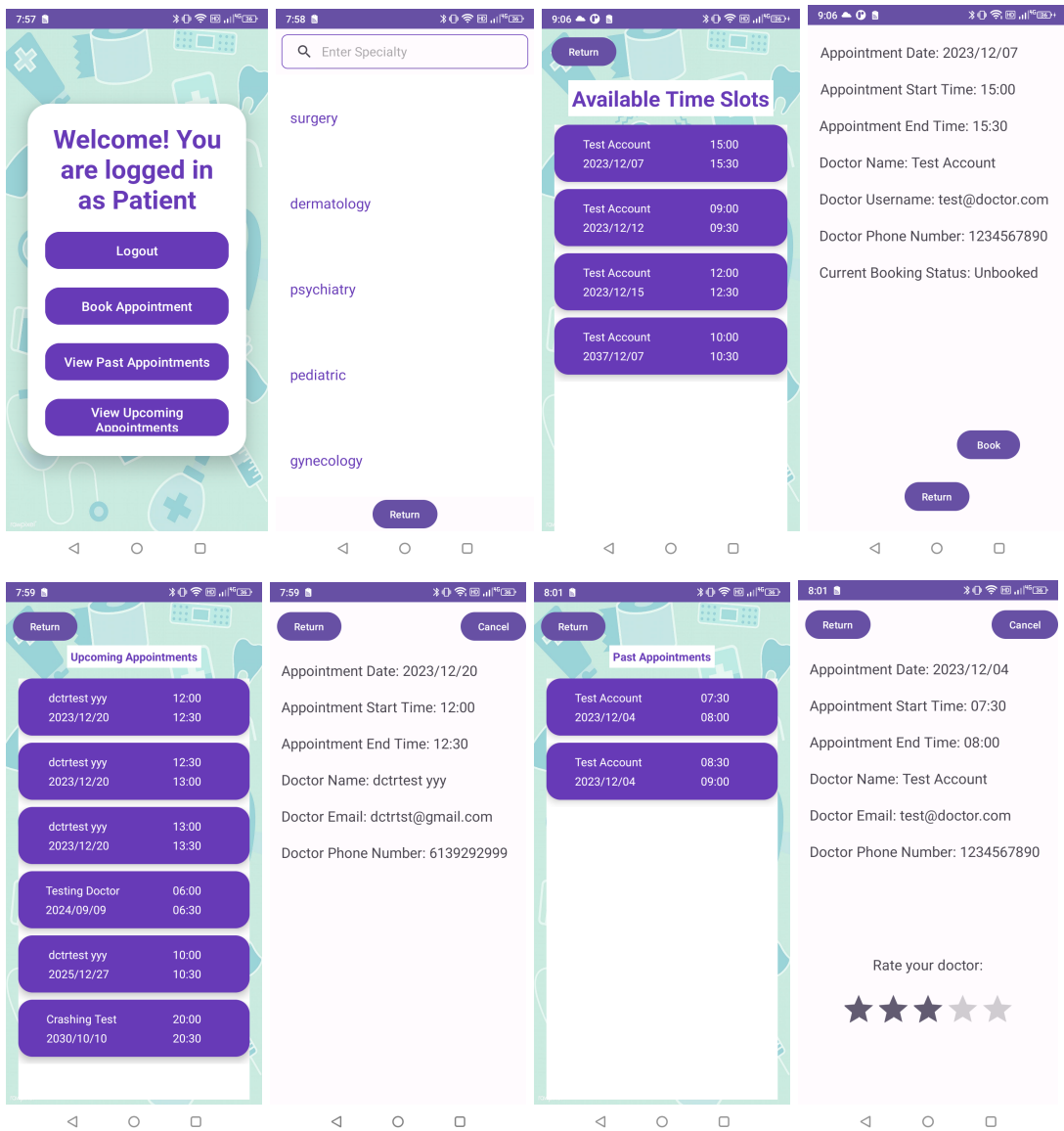
Full Name: Patient One Phone Number: 1234567890
Email: example1@example.com Health Card Number: 1

Accept

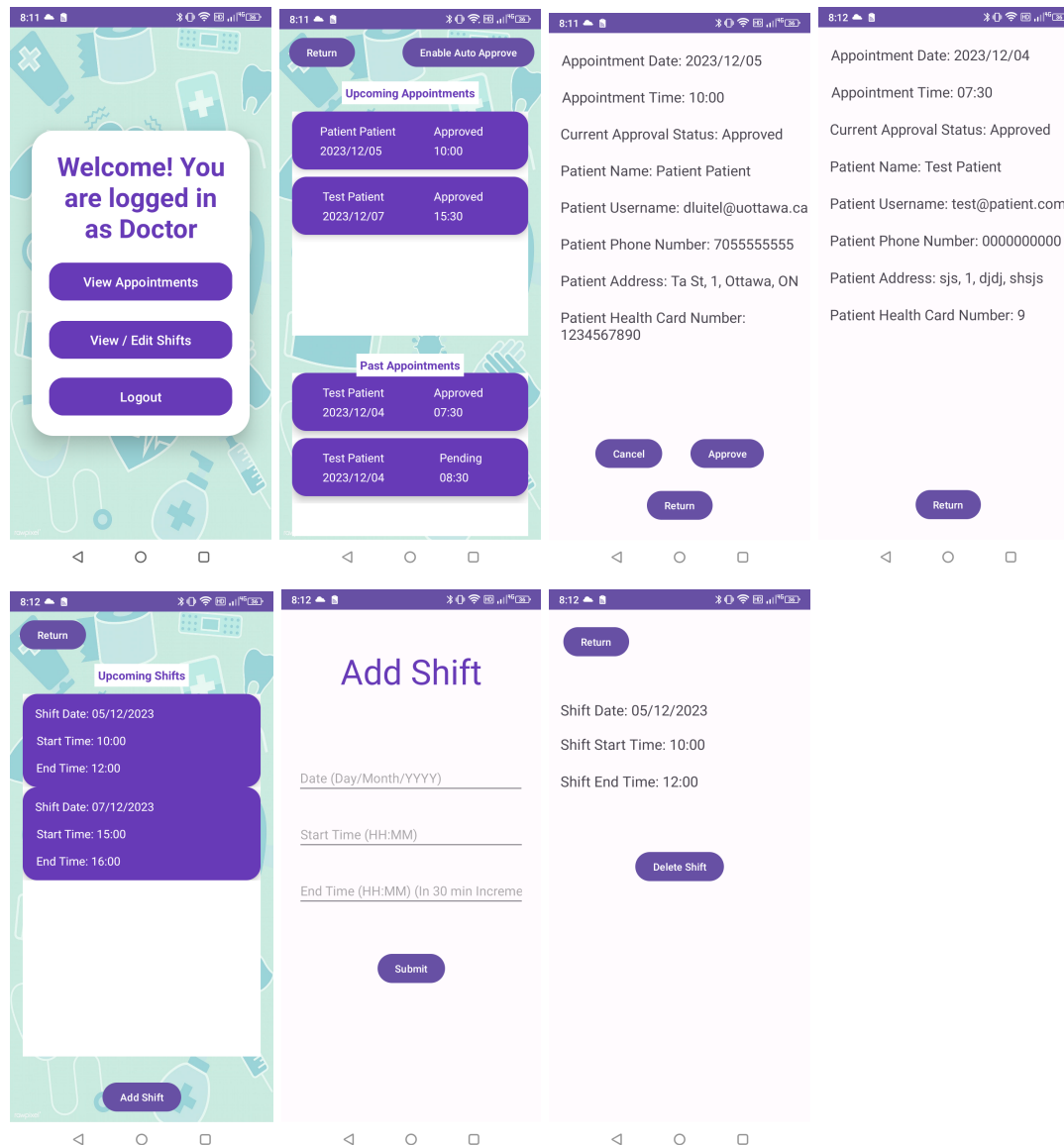
Return

Accept

Patient Screens:



Doctor Screens:



Lessons Learned

One of the biggest struggles during this project was integrating Firebase. This was a new tool and also caused the largest amount of problems for the app. There was a large learning curve when it came to understanding how to properly use Firebase and its tool, and there is probably much more to learn still. Despite this we were able to make use of the database to store a large variety of important data, required for running the app as intended.

Another important lesson was how to work better as a team. Specifically relating to understanding the work done by others and doing work that would be easy to understand. Throughout the entirety of the project there were many cases where parts of the app had been developed individually and were not compatible with the rest of the app and had to be adjusted. As the semester progressed, this did occur less and we gained a much greater understanding of the importance of strong and constant communication throughout development.