System Design Documentation

Patient Tracker System

Siara Saylor

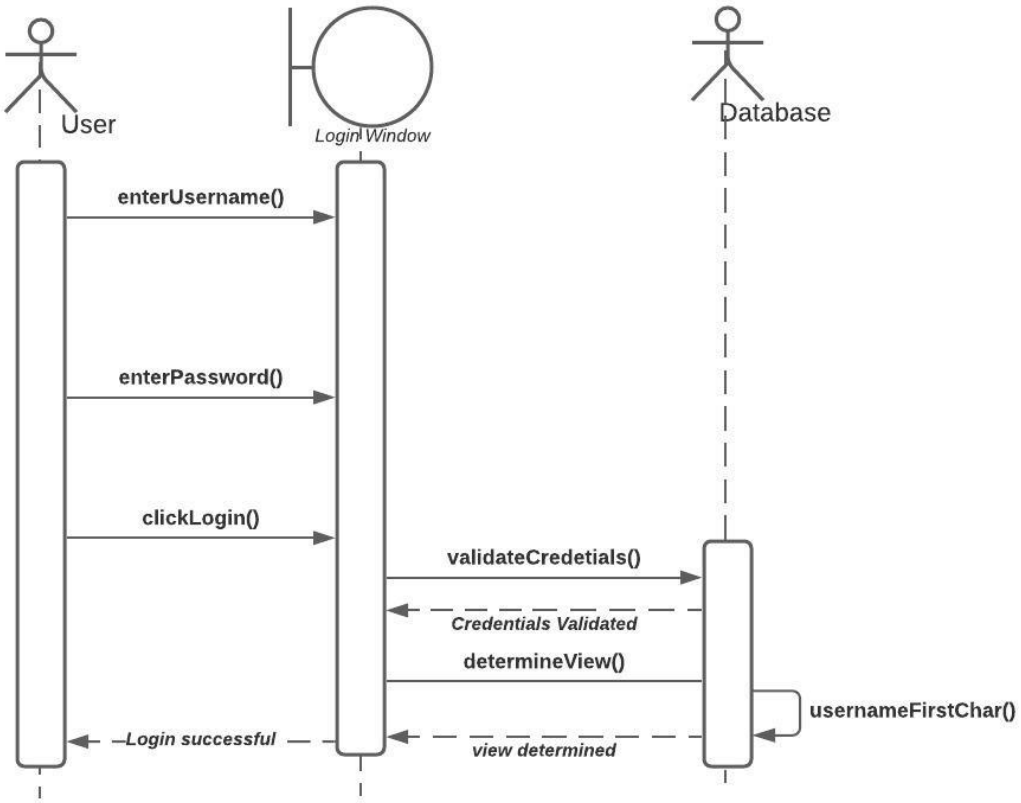
INFO-C451

Table of Contents

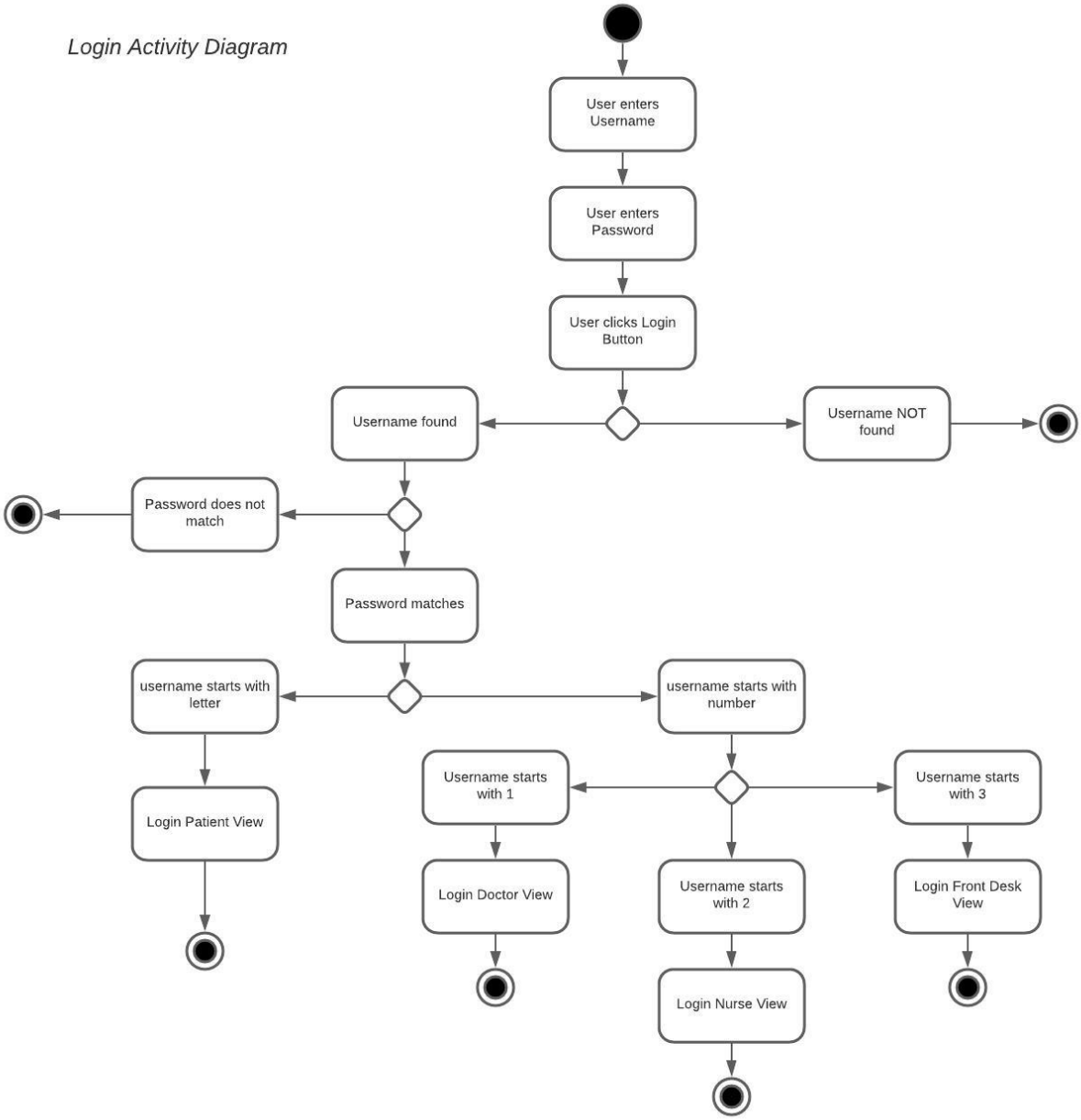
Section I: Interaction Diagrams

Req-ID 1 Login: Using the login function, there are four different views; patient, front desk staff, nurse, and doctor. All staff will have usernames that begin with a number and patients will have a username that begins with a letter. In order to not duplicate code by coding each staff page for each specific view, the program will utilize the user's username to constantly determine which view they need to be in. Every page will extend the staff menu class, to help determine the view to display. This will utilize the open-closed principle by having one class for the staff menu extended to all the staff pages. This will allow each staff page to be viewed by the specific staff member based on their username. The login window will be extended to this staff menu class, so it will be able to reference the user's username throughout the duration the user is logged into the program.

Login Sequence Diagram

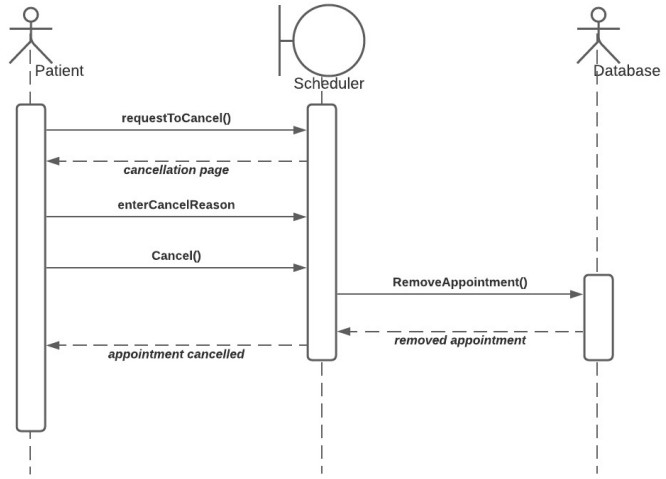


Login Activity Diagram

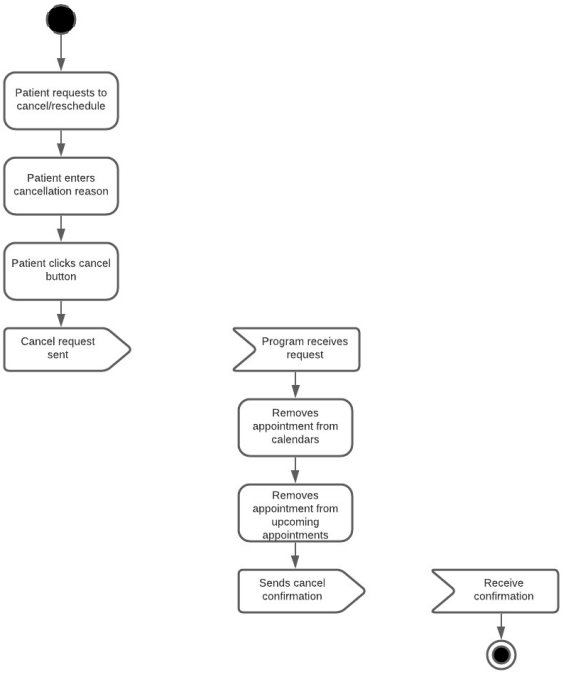


Req-ID 3 Cancel Appointment: For a patient to cancel an appointment, they will click the cancel/reschedule button. This will redirect the patient to the cancel/reschedule page. To cancel an appointment, the patient will just enter a reason to cancel and then click the cancel button. The system will then remove the appointment from the staff calendar and the upcoming appointments page for the patient. The design principle that will be utilized will be the single-responsibility principle. This principle will be perfect in this case due to cancelling an appointment is done by clicking a single button.

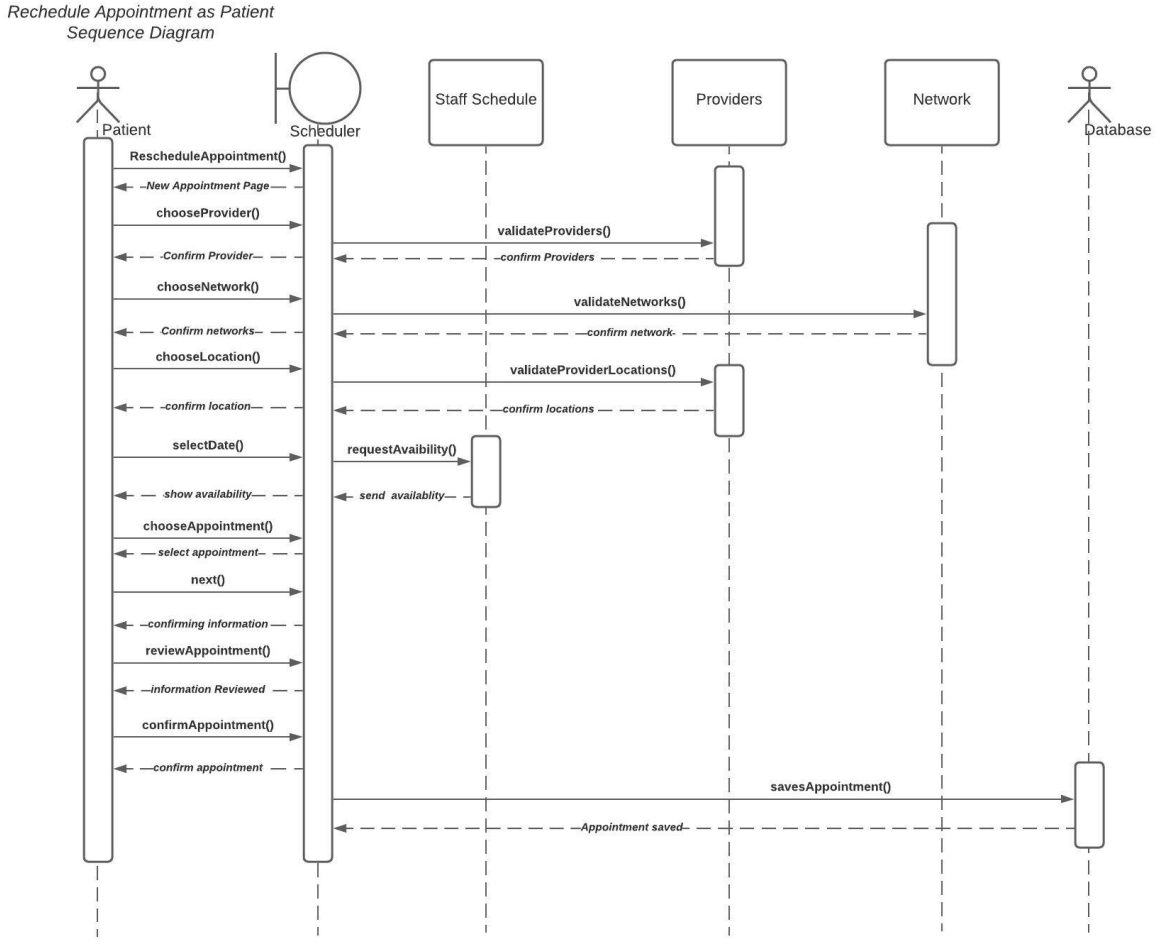
Cancel Appointment as a Patient Sequence Diagram



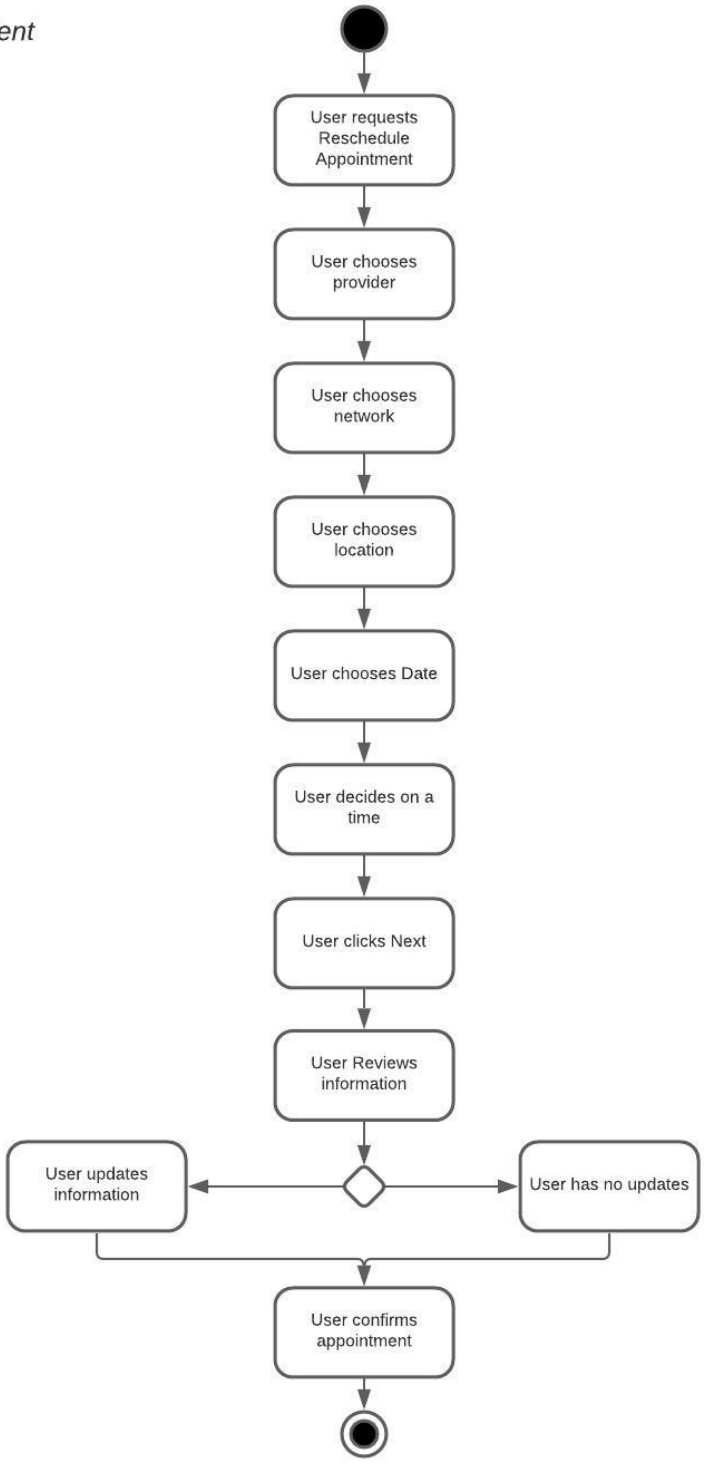
Cancel Appointment as a Patient Activity Diagram



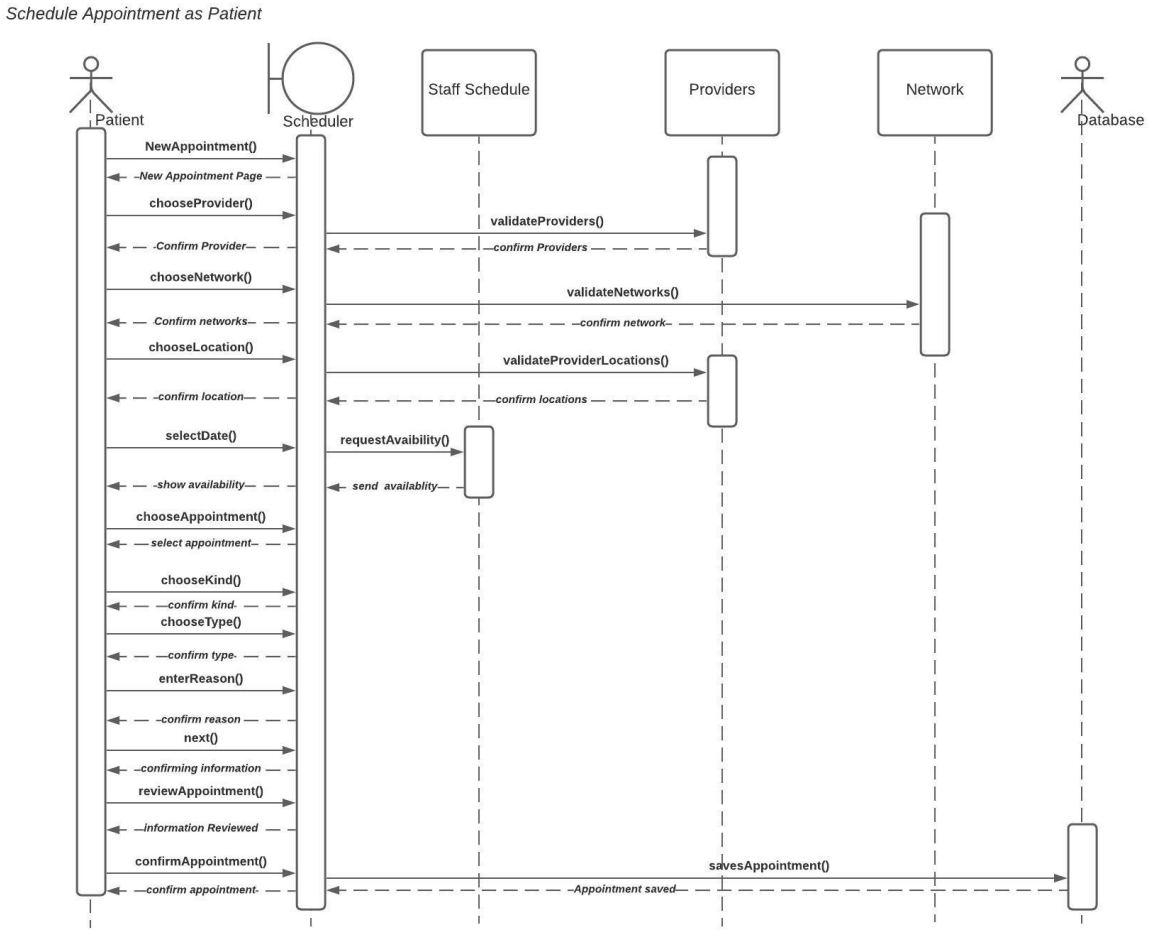
Req-ID 4 Reschedule Appointment: To reschedule an appointment, the user will click the cancel/reschedule appointment. The patient will then be redirected to the cancel/reschedule page. The second half of the page is set up similar to the schedule appointment page. The user will then choose a provider, network, location, date and time available. From there they will then be redirected to the review page where they can keep the kind of appointment, type of appointment, and reason for the appointment. Once confirmed the date will update the appointment to the new date. This will utilize the open-closed principle due to updating an appointment that has already been created.



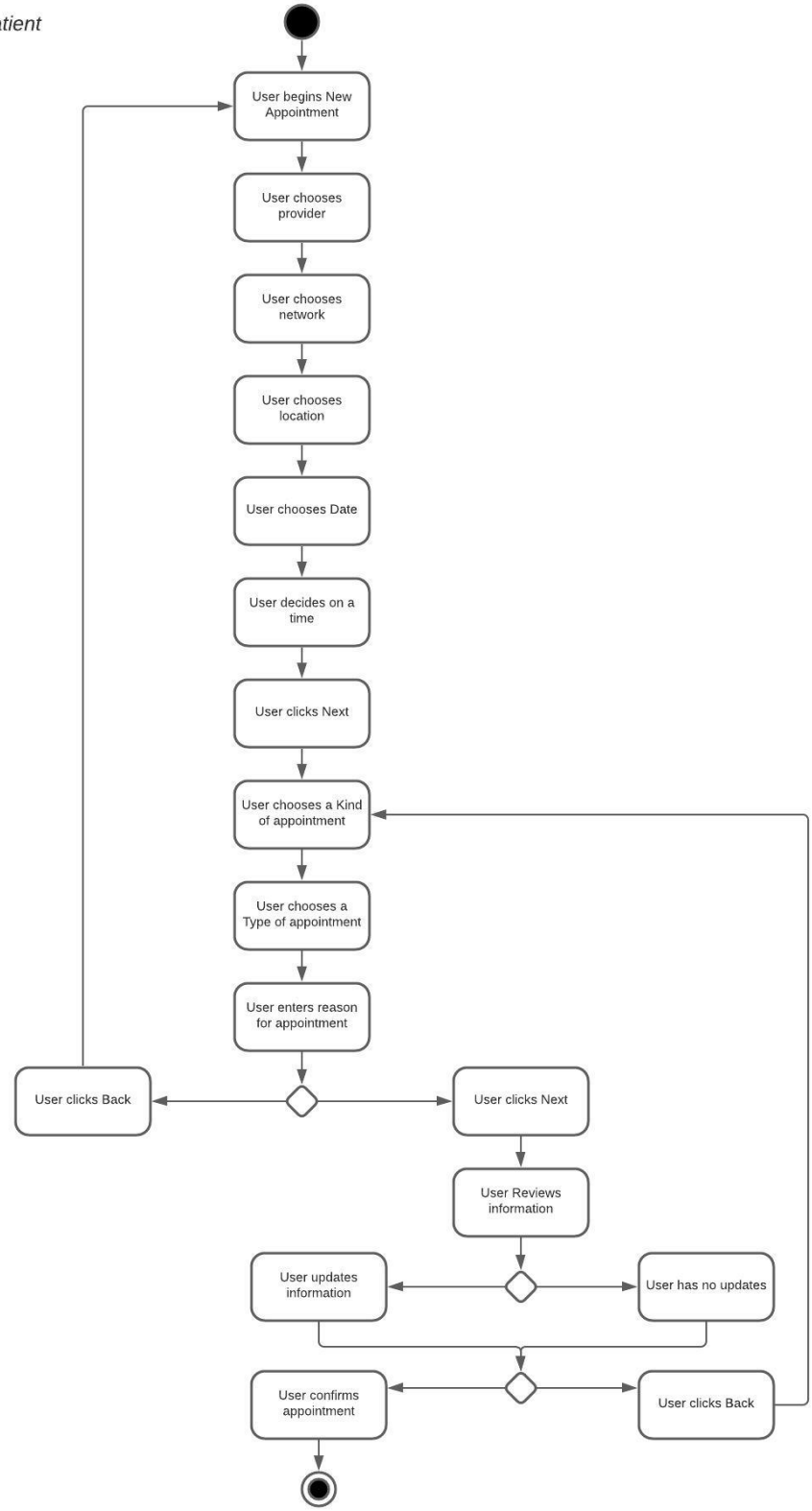
Reschedule Appointment as Patient Activity Diagram



Req-ID 5 Schedule Appointment: To schedule an appointment as a patient, the user must first choose a provider, network, location, and date. They can then choose an available time based off of the staff schedule. The patient will then click next, which will take them to another screen where they can choose the kind of appointment, type of appointment and the reason for their appointment. If they click the back button they will be redirected to the previous page, the next button will direct the patient to the review page, where they can update their information or review it then confirm the appointment. For the purpose of scheduling an appointment in the patient view, we will be utilizing the interface-segregation principle (ISP). ISP will help break down the steps of scheduling an appointment into three sections broken down into screens. This will help the patients be able to focus on what is needed on that one screen versus seeing everything on one screen.

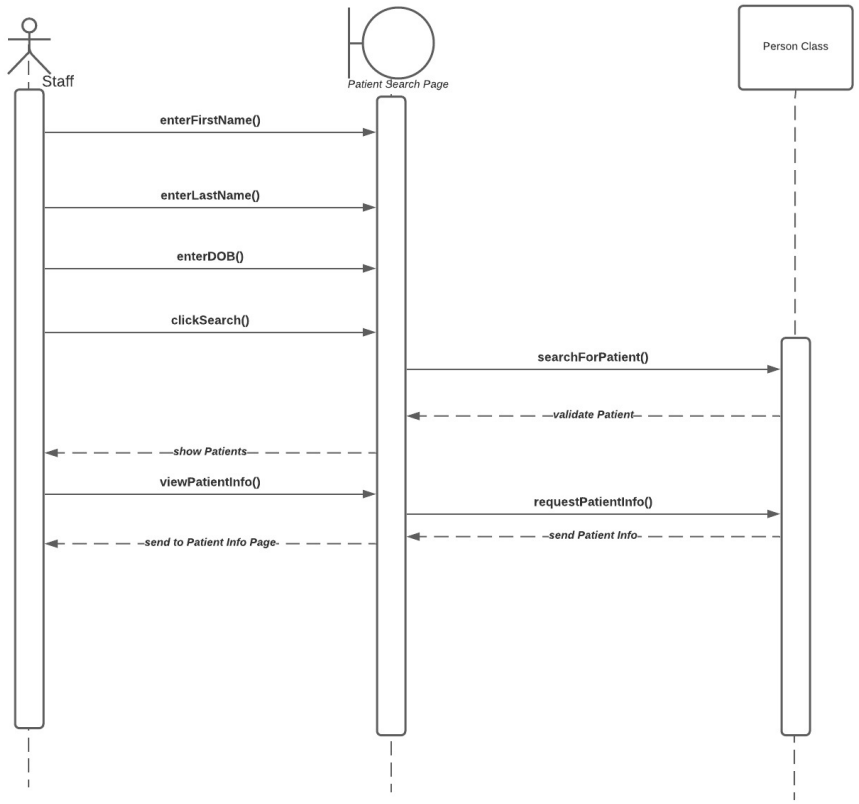


Schedule Appointment as Patient Activity Diagram

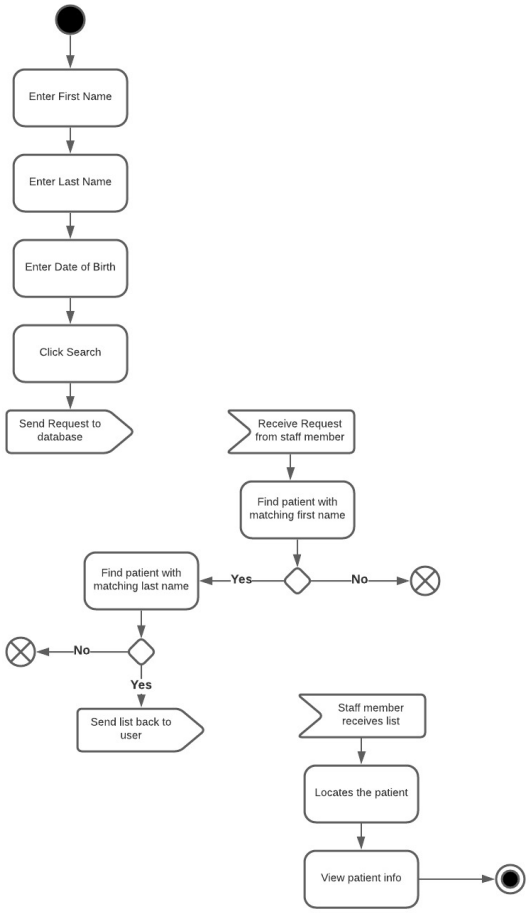


Req-ID 7 Patient Search: The patient search page is available to all the staff members and allows them to be able to search for a patient using their first name, last name, and date of birth. This page will utilize the single responsibility principle, which will have the program search a text document for the person.

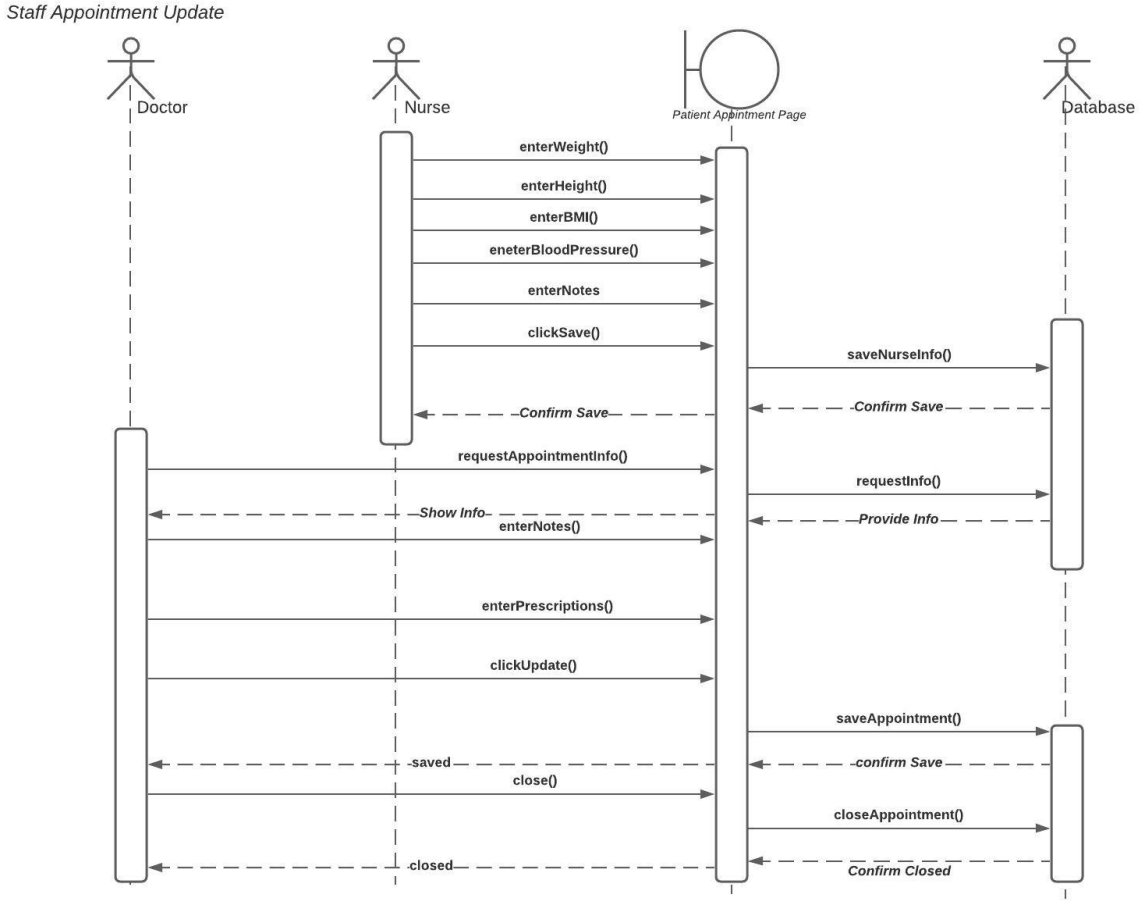
Patient Search as any Staff Member
Sequence Diagram



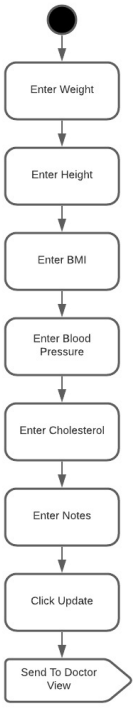
Patient Search as any Staff Member
Activity Diagram

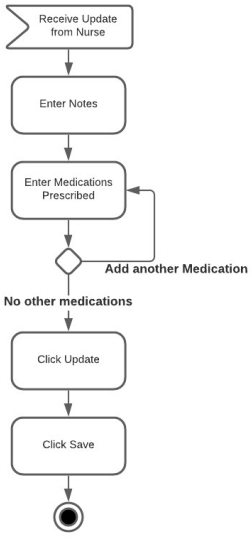


Req-ID 8 Staff Appointment Update: The staff appointment page allows the nurse and doctor to put in all the information gathered from the appointment. To be able to utilize the nurse view and the doctor view of the staff appointment update page, we will be utilizing the Liskov substitution principle. This principle will allow the nurse to only view what they are able to update. Once the nurse clicks the update button, the program will send the update to the doctor view, where the doctor will be able to view and add any additional notes and any prescribed medications.



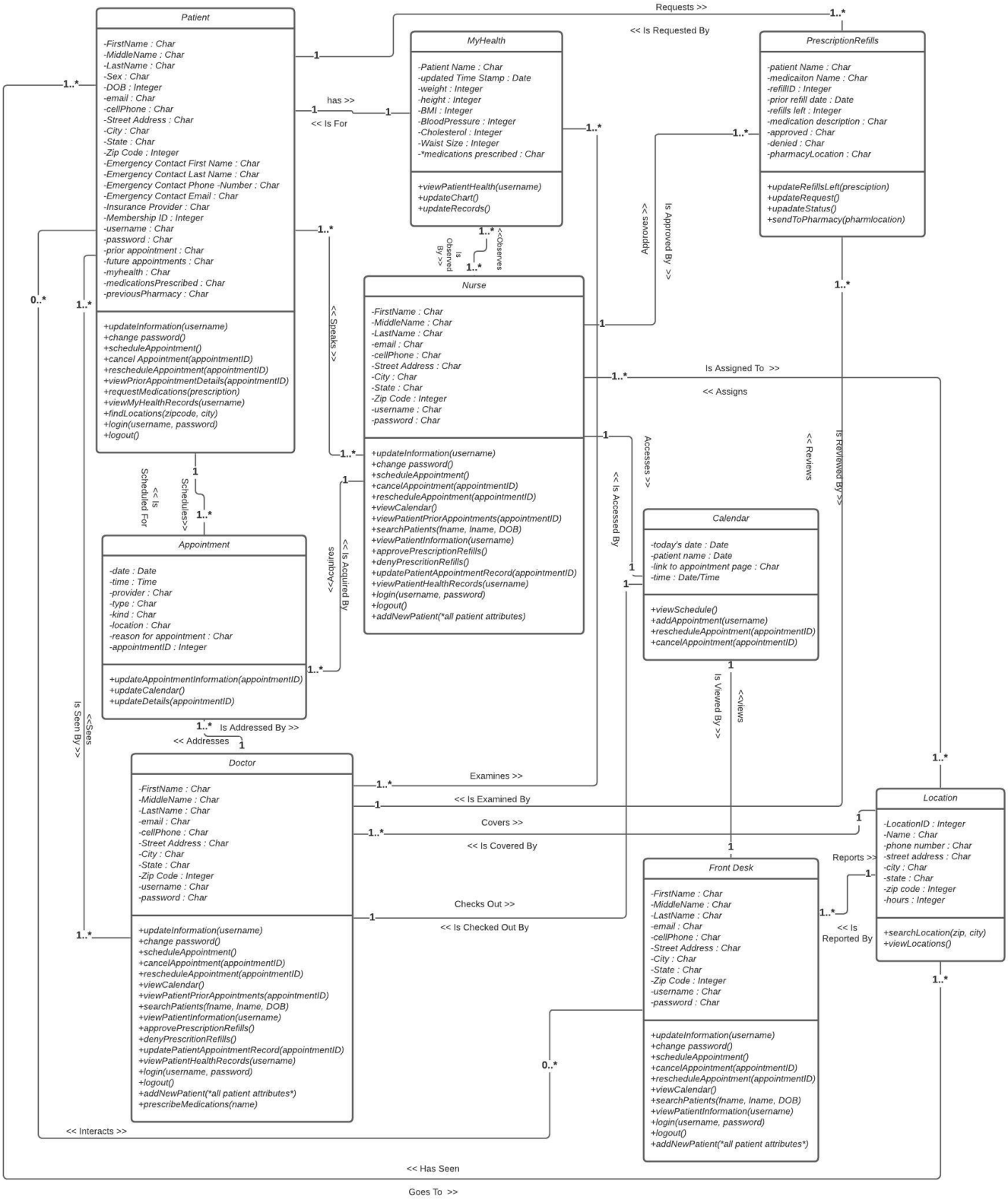
Staff Appointment Update Activity Diagram





Section II – Class Diagram and Interface Specification

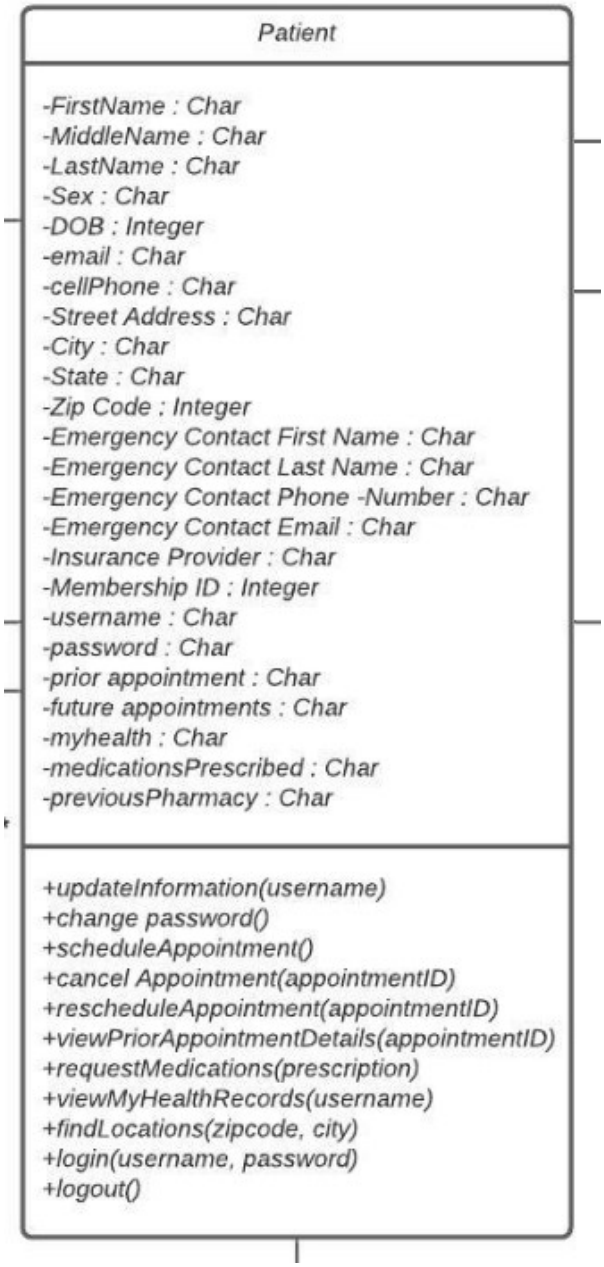
Class Diagram



Data Types and Operation Signatures

UML Notation	Nature of the Relationship	Class1	Class2	Description
1..*	One-to-many	Patient	Doctor	A patient can have one doctor or many doctors.
1..*	One-to-many	Doctor	Patient	A doctor can have one patient or many patients.
1..*	One-to-many	Patient	Nurse	A patient can see one nurse or many nurses.
1..*	One-to-many	Nurse	Patient	A nurse can have one patient or many patients.
1..*	One-to-many	Patient	Appointment	A patient can have one appointment or many appointments.
1	One-and-Only-One	Appointment	Patient	An appointment can only have one patient.
0..*	Zero-to-many	Patient	Front Desk	A patient may interact with zero front desk staff to many front desk staff.
0..*	Zero-to-many	Front Desk	Patient	A front desk staff may interact with zero patients to many patients.
1	One-to-many	Patient	Location	A patient may go to one location or many locations.
1..*	One-to-many	Location	Patient	A location can have one patient to many patients.
1	One-and-Only-One	Patient	My Health	A patient can only have one My Health account.
1	One-and-Only-One	My Health	Patient	A My Health account can only be for one patient.
1..*	One-to-many	Patient	Prescription Refills	A patient can have one to many prescription refills.
1	One-and-Only-One	Prescription Refills	Patient	A prescription refill is only for one patient.
1	One-and-Only-One	Front Desk	Calendar	A front desk staff will only have access to one calendar.
1	One-and-Only-One	Calendar	Front Desk	A calendar is only available to one front desk staff.
1	One-and-Only-One	Front Desk	Location	A front desk staff only works at one location.
1..*	One-to-many	Location	Front Desk	A location can have one to many front desk staff employees.
1..*	One-to-many	Nurse	Appointment	A nurse can have one appointment or many appointments.

1	One-and-Only-One	Appointment	Nurse	An appointment can only have one nurse.
1	One-and-Only-One	Nurse	Calendar	A nurse can view only one calendar.
1	One-and-Only-One	Calendar	Nurse	A calendar is only available for one nurse.
1..*	One-to-many	Nurse	Location	A nurse can work at one to many locations.
1..*	One-to-many	Location	Nurse	A location can have one to many nurses employed.
1..*	One-to-many	Nurse	My Health	A nurse can have access to one to many My Health accounts.
1..*	One-to-many	My Health	Nurse	A My Health account can be accessed by one to many nurses.
1..*	One-to-Many	Nurse	Prescription Refills	A nurse can approve/deny one to many prescription refills.
1	One-and-Only-One	Prescription Refills	Nurse	A prescription refill can only be approved/denied by one nurse.
1..*	One-to-many	Doctor	Appointment	A doctor can have one to many appointments.
1	One-and-Only-One	Appointment	Doctor	An appointment can only have one doctor.
1	One-and-Only-One	Doctor	Calendar	A doctor can view only one calendar.
1	One-and-Only-One	Calendar	Doctor	A calendar is only available for one doctor.
1..*	One-to-many	Doctor	Location	A doctor can work at one to many locations.
1..*	One-to-many	Location	Doctor	A location can have one to many doctors employed.
1..*	One-to-many	Doctor	My Health	A doctor can have access to one to many My Health accounts.
1..*	One-to-many	My Health	Doctor	A My Health account can be accessed by one to many doctors.
1..*	One-to-many	Doctor	Prescription Refills	A doctor can approve/deny one to many prescription refills.
1	One-and-Only-One	Prescription Refills	Doctor	A prescription refill can only be approved/denied by one doctor.



1. **Patient Class:** This class is for every patient in the system.

Attributes: The attributes for this class are mainly characters (Strings in Java). This includes First Name, Middle Name, Last Name, Sex, Email, Cell Phone Number, Street Address, City, State, Emergency Contact First Name, Emergency Contact Last Name, Emergency Contact Phone Number, Emergency Contact Email, Insurance Provider, username, password, Prior Appointments, Future Appointments, MyHealth, Medications Prescribed, and Previous Pharmacy Location. The date of birth (DOB), zip code, and membership ID will all be integers.

Methods:

i. The updateInformation(username) method allows the user to update their information in the system.

ii. The changePassword() method allows the user to change the password of the user. This will update their information as well.

iii. scheduleAppointment() method allows the patient to schedule an appointment.

iv. cancelAppointment(appointmentID) method allows the patient to cancel a future appointment. What is needed for this is the appointmentID to make sure the correct appointment is being cancelled.

v. rescheduleAppointment(appointmentID) will allow the patient to reschedule a future appointment. What is needed for this is the appointmentID to make sure the correct appointment is being rescheduled.

vi. findLocations(zipcode, city) allows the patient to search for a location by entering the zipcode and city of the location the patient is searching for.

vii. requestMedications(prescription) allows the patient to request refills for a prescription by passing the prescription name.

viii. viewMyHealthRecords(username) allows the patient to view their health records by passing their username to pull up their information.

ix. Login(username, password) allows the patient to log into the system by entering their username and password.

x. Logout() this allows the patient to log out of the system.

xi. viewPriorAppointmentDetails(appointmentID) allows the patient to view a prior appointment that they have chosen. What is needed for this is the appointmentID to verify which appointment the patient wishes to view.

2. **Front Desk Class:** This is for every front desk staff member that has access to the application.

Attributes: Most of the attributes for the front desk staff are characters. These attributes include first name, middle name, last name, email, cell phone, street address, city, state, username and password. The only attribute that is an Integer is the zip code.

Methods:

i. updateInformation(username) allows the front desk staff to update their information and patient information.
ii. changePassword() allows the front desk staff to change their password.
iii. scheduleAppointment() allows the front desk staff to schedule an appointment for a patient.
iv. cancelAppointment(appointmentID) allows the front desk staff to cancel an appointment for a patient. What is needed for this is the appointmentID to make sure the correct appointment is being cancelled.
v. rescheduleAppointment(appointmentID) allows the front desk staff to reschedule an appointment for a patient. What is needed for this is the appointmentID to make sure the correct appointment is being rescheduled.
vi. viewCalendar() allows the front desk staff to view the calendar.
vii. searchPatients(fname, lname, DOB) this allows the front desk staff to search for a patient using their first name, last name, and date of birth (DOB).
viii. viewPatientInformation(username) this allows the front desk staff to view the patient's contact information by using the patient's username.
ix. Login(username, password) this allows the front desk staff to log into the system using a username and password.
x. Logout() this allows the front desk staff to log out of the system.
xi. addNewPatient(*all patient attributes*) this allows the front desk staff to create a new patient and add them to the system. The staff member will need the patient's first name, middle name, last name, DOB, phone number, email, SSN, insurance provider, membership ID, street address, city, state, zip code, emergency contact first and last name, phone, email, username, and password.

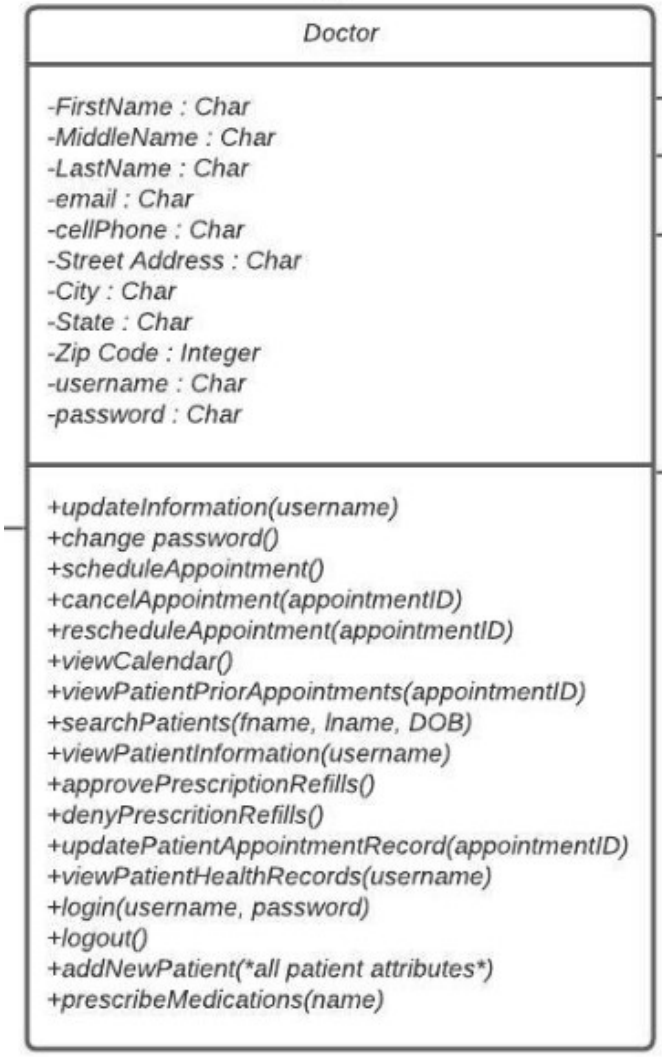
3.Nurse Class: This is for every nurse that has access to the application.

Attributes: The attributes are mostly characters. The attributes that are characters include the nurse's first name, middle name, last name, email, cell phone, street address, city, state, username, and password. The only attributes that is an Integer is the zip code.

Methods:

i. updateInformation(username) allows the nurse to update the information of their account or a patient's account.
ii. changePassword() allows the nurse to update their password.
iii. scheduleAppointment() allows the nurse to schedule an appointment for a patient.
iv. cancelAppointment(appointmentID) allows the nurse to cancel an appointment for a patient. The appointmentID will have to be passed in order to cancel the right appointment.
v. rescheduleAppointment(appointmentID) allows the nurse to reschedule an appointment for a patient. The appointmentID will have to be passed in order to reschedule the correct appointment for the patient.
vi. viewCalendar() allows the nurse to view the calendar.
vii. viewPatientPriorAppointments(appointmentID) allows the nurse to view a patient's prior appointment by passing the appointmentID for the appointment the nurse wants to view.
viii. searchPatients(fname, lname, DOB) this allows the nurse to search for a patient using their first name, last name, and date of birth (DOB).
ix. viewPatientInformation(username) this allows the nurse to view the patient's contact information by using the patient's username.
x. approvePrescriptionRefills() allows a nurse to approve a prescription request.
xi. denyPrescriptionRefills() allows a nurse to deny a prescription request.
xii. updatePatientAppointmentRecord(appointmentID) allows a nurse to update the appointment page for the patient's appointment. The appointment ID will be passed in order for the nurse to update the correct appointment.
xiii. viewPatienHealthRecords(username) this allows a nurse to view the patient's health record by passing the patient's username.

xiv. Login(username, password) this allows the nurse to log into the system using a username and password.
xv. Logout() this allows the nurse to log out of the system.
xvi. addNewPatient(*all patient attributes*) this allows the nurse to create a new patient and add them to the system. The staff member will need the patient's first name, middle name, last name, DOB, phone number, email, SSN, insurance provider, membership ID, street address, city, state, zip code, emergency contact first and last name, phone, email, username, and password.



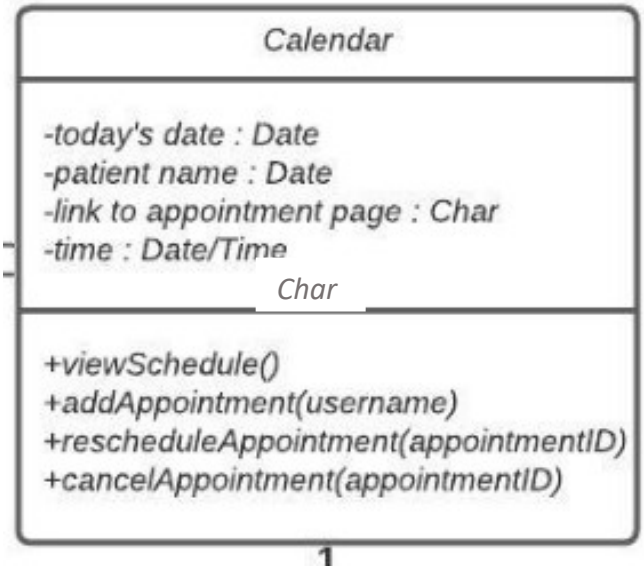
4. **Doctor Class:** This is for every doctor that has access to the application.

Attributes: The attributes are mostly characters. The attributes that are characters include the doctor's first name, middle name, last name, email, cell phone, street address, city, state, username, and password. The only attributes that is an Integer is the zip code.

Methods:

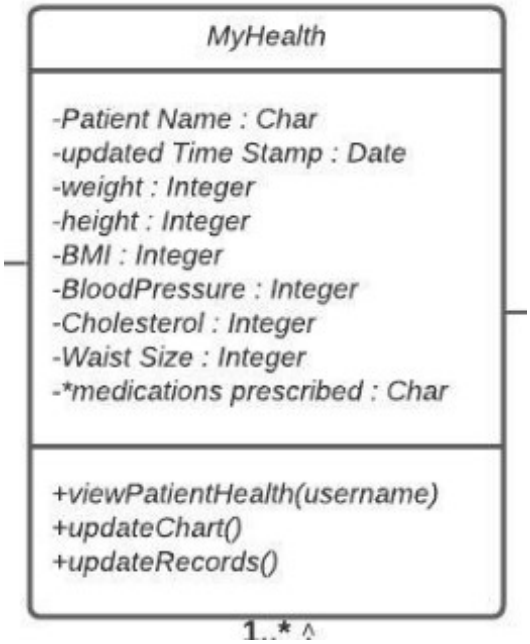
i. updateInformation(username) allows the doctor to update the information of their account or a patient's account.
ii. changePassword() allows the doctor to update their password.
ii. scheduleAppointment() allows the doctor to schedule an appointment for a patient.
v. cancelAppointment(appointmentID) allows the doctor to cancel an appointment for a patient. The appointmentID will have to be passed in order to cancel the right appointment.
v. rescheduleAppointment(appointmentID) allows the doctor to reschedule an appointment for a patient. The appointmentID will have to be passed in order to reschedule the correct appointment for the patient.
i. viewCalendar() allows the doctor to view the calendar.
ii. viewPatientPriorAppointments(appointmentID) allows the doctor to view a patient's prior appointment by passing the appointmentID for the appointment the doctor wants to view.
viii. searchPatients(fname, lname, DOB) this allows the doctor to search for a patient using their first name, last name, and date of birth (DOB).
ix. viewPatientInformation(username) this allows the doctor to view the patient's contact information by using the patient's username.
x. approvePrescriptionRefills() allows a doctor to approve a prescription request.

xi. denyPrescriptionRefills() allows a doctor to deny a prescription request.
xii. updatePatientAppointmentRecord(appointmentID) allows a doctor to update the appointment page for the patient's appointment. The appointment ID will be passed in order for the doctor to update the correct appointment.
xiii. viewPatienHealthRecords(username) this allows a doctor to view the patient's health record by passing the patient's username.
xiv. Login(username, password) this allows the doctor to log into the system using a username and password.
xv. Logout() this allows the doctor to log out of the system.
xvi. addNewPatient(*all patient attributes*) this allows the doctor to create a new patient and add them to the system. The staff member will need the patient's first name, middle name, last name, DOB, phone number, email, SSN, insurance provider, membership ID, street address, city, state, zip code, emergency contact first and last name, phone, email, username, and password.
xvii. prescribeMedications(name) allows a doctor to prescribe medications to patients on the appointment page. Only the name will of the prescription will have to be passed.



5. **Calendar Class:** This is a class that is available for all three staff views and updates as the appointments are added, removed, or rescheduled.
Attributes: The attributes for the calendar class include today's date as a Date. Patient's name and link to appointment page are both characters. Finally, time is going to be Date/Time.

Methods:
i. viewSchedule() allows the staff member to view the calendar and see all the appointments for the month.
ii. addAppointment(username) allows the staff member to create an appointment for a patient using the patient's username. This will add the appointment to the calendar.



iii. rescheduleAppointment(appointmentID) allows the staff member to reschedule an appointment using the appointment ID. This will update the appointment on the calendar.

iv. cancelAppointment(appointmentID) allows the staff member to cancel an appointment using an appointment ID. This will remove the appointment from the calendar.

6. **MyHealth Class:** This class is to show each patient's health record and progress through their appointments.

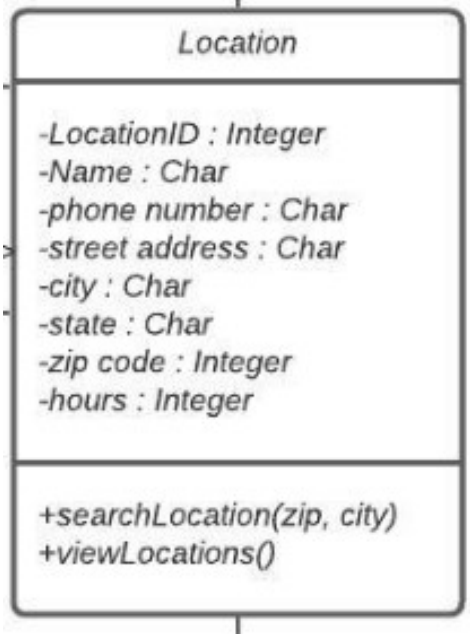
Attributes: The attributes for MyHealth include the patient's name, and any medications prescribed, which are character data types. The updated time stamp is a date data type. All the remaining attributes including weight, height, BMI, Blood Pressure, Cholesterol, and Waist Size are integers.

Methods:

i. viewPatientHealth(username) allows a staff member or a patient to view the health page of the username that is passed. This pulls up all the username's information.

ii. updateChart() is when the user clicks on the button for one of the attributes and the chart of progress updates to the right side of the page.

iii. updateRecords() when a staff member closes an appointment, it updates the patient's record by adding the new information to this page.



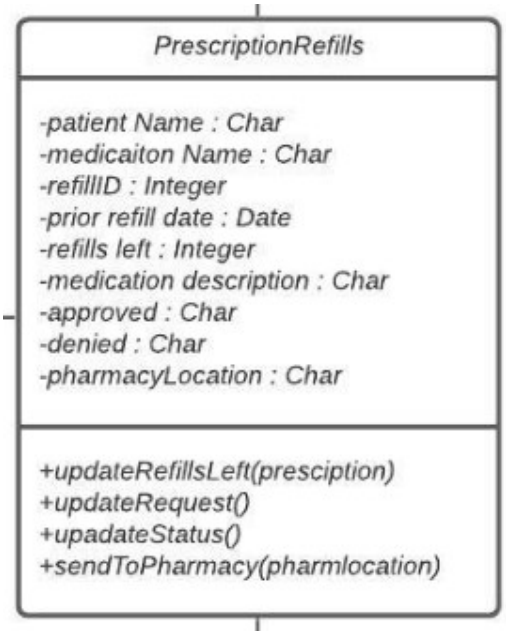
7. **Locations Class:** This class is for each location available for patient's to make appointments and for staff to access the application.

Attributes: The location ID will have a data type of an integer. This will allow each location to have a unique title. The location's name, phone number, street address, city, and state will be character data types. The zip code and hours will be integers.

Methods:

i. searchLocation(zip, city) is a method that searches for a location using the zip code and city that is entered by the patient.

ii. viewLocations() shows the locations information for any in that area.

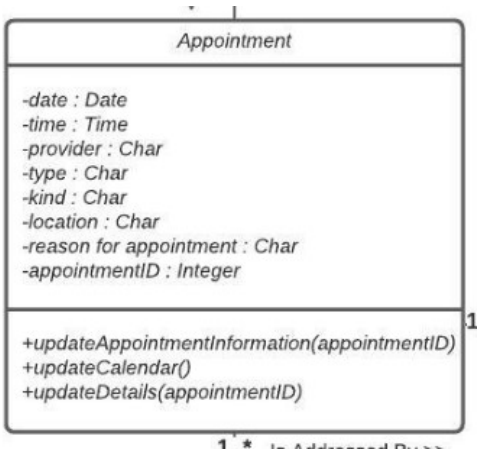


8. **Prescription Refills Class:** This class is for all the prescription refills that have been submitted and reviewed.

Attributes: The patient's name, medication name, medication description, approved, denied, and pharmacy location will all be character data types for this class. The prior refill date will be a date data type. The refill ID and refills left will be integers.

Methods:

i. updateRefillsLeft(prescription) is a method that will reduce the count of how many refills are left once a request has been approved by a nurse or doctor.
ii. updateRequest() will send the request to the nurse and doctor for approvals.
iii. updateStatus() will update the status as approved or denied made by the nurse or the doctor.
iv. sendToPharmacy(pharmlocation) is a method that will be called if the status is updated to approved. This will send the request to the pharmacy location of the patient's choosing.



9. **Appointment Class:** This class is for each appointment that is created and identified uniquely by the appointment ID.

Attributes: The date is going to be a date data type and the time is going to be a date/time data type. The appointment ID is going to be an integer data type. The remaining attributes including provider, type, kind, location, and reason for appointment are all going to be character data types.

Methods:

i. updateAppointmentInformation(appointmentID) is a method that allows nurses and doctors to update the appointment with the information collected during the appointment. The appointment ID is passed in order to update the correct record.
ii. updateCalendar() is a method that updates the calendar with the date, time, and patient name to the calendar when an appointment is created.
iii. updateDetails(appointmentID) when there are changes to an appointment, updateDetails method will update the appointment information using the appointment ID to make sure the correct appointment is being updated.

Traceability Matrix

Req't	PW	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11	UC12	UC13	UC14	UC15	UC16	UC17	UC18
1	5	X																	
2	5		X																
3	5			X															
4	5				X														
5	5					X													
6	3						X												
7	5							X											
8	5								X										
9	5									X									
10	3										X								
11	3											X							
12	5												X						
13	5													X					
14	1														X				
15	3															X			
16	5																X		
17	2																	X	
18	2																		X
Max PW		5	5	5	5	5	3	5	5	5	3	3	5	5	1	3	5	2	2
Total PW		5	5	5	5	5	3	5	5	5	3	3	5	5	1	3	5	2	2

Description of Requirements with Test Cases

Req-ID	Requirement	Priority Weight	Test Case ID	Test Case Description
1	Log In	HIGH	TC01	Login with invalid username and password
			TC02	Login without inputting username and password
			TC03	Login with valid credentials
			TC04	Login as patient
			TC05	Login as Front Desk Staff
			TC06	Login as Nurse
			TC07	Login as Doctor
2	Update Information	HIGH	TC08	Patient updates information
			TC09	Front Desk updates information
			TC10	Nurse updates information
			TC11	Doctor updates information
3	Cancel Appointment	HIGH	TC12	Patient cancels appointment
			TC13	Front Desk cancels appointment
			TC14	Nurse cancels appointment
			TC15	Doctor cancels appointment
4	Reschedule Appointment	HIGH	TC16	Patient reschedules appointment
			TC17	Front desk reschedules appointment
			TC18	Nurse reschedules appointment
			TC19	Doctor reschedules appointment
5	Schedule Appointment	HIGH	TC20	Patient schedules appointment

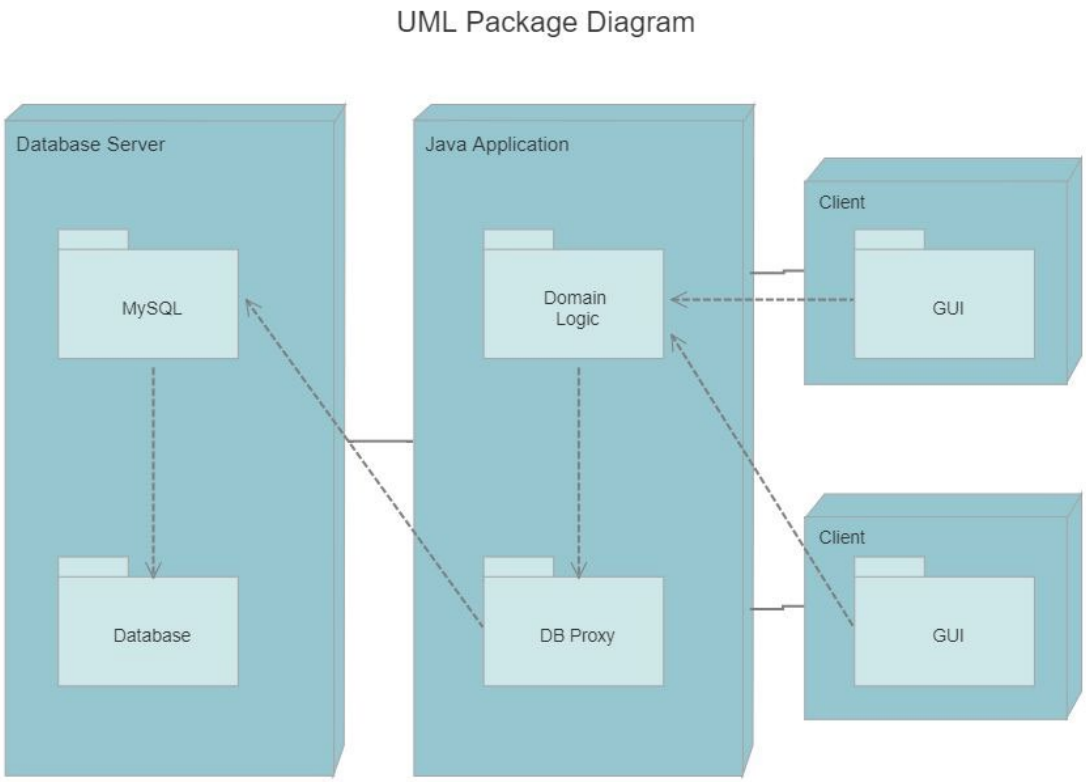
			TC21	Front desk schedules appointment
			TC22	Nurse schedules appointment
			TC23	Doctor schedules appointment
6	Pre-Appointment Information	MED	TC24	Patient completes pre-appointment information
7	Staff Patient Search	HIGH	TC25	Front desk searches for patient
			TC26	Nurse searches for patient
			TC27	Doctor searches for patient
8	Staff Appointment Update	HIGH	TC28	Nurse updates patient appointment page
			TC29	Doctor updates patient appointment page
9	Adding New Patient	HIGH	TC30	Front desk adds a new patient
			TC31	Nurse adds a new patient
			TC32	Doctor adds a new patient
10	Medication Refills	MED	TC33	Patient requests medication refills
11	Medication Approvals	MED	TC34	Nurse approves medication refills
			TC35	Nurse denies medication refills
			TC36	Doctor approves medication refills
			TC37	Doctor denies medication refills
12	Prior Appointments	HIGH	TC38	Patient views prior appointments
			TC39	Nurse views prior appointments
			TC40	Doctor views prior appointments
13	Staff View Patient Schedule	HIGH	TC41	Front desk views patient schedule
			TC42	Nurse views patient schedule
			TC43	Doctor views patient schedule
14	Staff View Staff Schedule	LOW	TC44	Front desk view staff schedule
			TC45	Nurse views staff schedule
			TC46	Doctor views staff schedule
15	View My Health Page	MED	TC47	Patient views My Health Page
			TC48	Nurse views My Health Page
			TC49	Doctor views My Health Page
16	Log Out	HIGH	TC50	Click CLOSE on login menu
			TC51	Patient clicks Exit in application menu
			TC52	Patient clicks Log Out in application menu
			TC53	Front Desk clicks Exit in application menu
			TC54	Front Desk clicks Log Out in application menu
			TC55	Nurse clicks Exit in application menu
			TC56	Nurse clicks Log Out in application menu
			TC57	Doctor clicks Exit in application menu
			TC58	Doctor clicks Log Out in application menu
17	Request new password	MED	TC59	user clicks forgot password link
18	Create new password	MED	TC60	Customer support generates temp password
			TC61	Customer support sends email link to create new password

Section III- System Architecture and System Design

Architectural Styles

The architectural style for the patient tracker system will be the client-server model. All of the data of the appointments, patient information, and health records will be kept on a server. All the users that have access to the desktop application will be able to connect to MySQL database to be able to access the information.

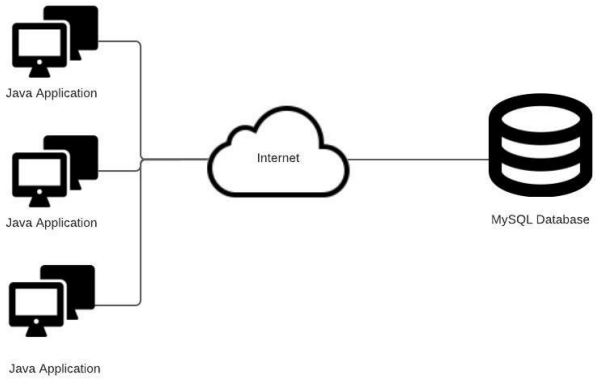
Identifying Subsystems



Each client will access the GUI for the Java Application. The Java Application includes all the domain logic, which allows the client to make the decisions on what they plan to execute. This would include, updating information, scheduling an appointment, cancelling an appointment, or rescheduling an appointment. The domain logic depends on the DB Proxy, which connects to the MySQL Database Server. MySQL will access the database containing all the data for the application.

Mapping Subsystems to Hardware

This application will be downloaded on any computer that wants access to the application. There will be a MySQL database running on one computer that all the other computers will have access to by connecting to the database. All the computers that have downloaded the application will connect to the internet, where they will be able to log in and access the database data for the program. Below is an image on what this would look like.



Persistent Data Storage

This application does require saving data constantly. From the user updating information, to creating appointments, the application will require frequent saving. For persistent data storage, the application will use relational databases in order to keep up with the saving and updates to the data. MySQL will be the relational database used in this application.

Network Protocol

The network protocol being used by this application is Java JDBC. This network protocol was chosen because Java JDBC works well with MySQL. MySQL will allow the application to be stored in a database on the server. Java JDBC will be able to connect to MySQL to access the information.

Global Control Flow

Execution Order

This application is event driven. The application will only act if the user clicks button or enters information.

Time Dependency

This application is real-time. As updates are created, the data will automatically do those updates. This also is the same for creating appointments, adding patients, and updating the appointments. If multiple users are looking at the same page and someone updates the information, the other user may have to refresh the page in order for the information to update, but overall, the time dependency should be real-time.

Hardware Requirements

The user will need a color display using Windows (Vista to Windows10), Max OS X, or Linux. Below is a broken down list depending on the operating system.

Windows

- Windows Server 2008 R2 SP1 (64-Bit)
- Windows Server 2012 and 2012 R2 (64-Bit)
- RAM: at least 128 MB
- Disk Space: at least 124 MB **note for Java Update the user will need 2MB
- Processor: at least 266 MHz processor

Mac OS X

- Administrator privileges for installation
- 64-Bit browser

Linux

- Oracle Linux 5.5+[1], 6.x, or 7.x
- Red Hat Enterprise Linux 5.5+[1], 6.x, or 7.x
- Suse Linux Enterprise Server 10 SP2+, 11.x or 12.x
- Ubuntu Linux 12.04 LTS, 13.x , 14.x, 15.04, 15.10

Section IV – Algorithms and Data Structures

Data Structures

This system currently uses arrays to read information from a text file until MySQL database is able to connect to Java. Currently the text file is set up as all the information listed with commas in between each index. The code will then search the file for a specific index. Each line starts a new array, which helps with looking for a username and password in order to enter the program. This text file is only temporary until MySQL is updated and ready to connect, then the code will be able to search through the database versus a text file.

All the appointments are set up as array objects as well that include all the information gathered from scheduling an appointment. These will save to a text file (for testing) and eventually automatically create a new record in the MySQL database.

Arrays were more flexible to use in order to search for specific information. Having a username for the person text file as the first index helps when searching for a specific person. The code also sets up how it will process in MySQL by being able to search records to find specific people or appointments.

Section V – User Interface Design and Implementation

The layout design hasn't changed too drastically from the mock ups in Report #1. However, the calendar page is better than the design. Overall this will help make the calendar page more user friendly by having add buttons on every date. Also, when an appointment is made, the number of appointments for that day will increase. Once the staff member clicks the View button, the information regarding all the appointments that day will show up on the right hand column.

Also, to schedule the appointment, I originally had buttons for the patient to choose the kind and type into radio buttons. This will allow the data to be saved more easily as well as more user friendly for the user so they can see which option is selected.

For the My Health page, radio buttons will also be implemented for the different options for the charts. This will allow the user to see which one is selected versus a button click.

Section VI – Design of Tests

Demo 1 Test Cases

Req-ID	Requirement	Priority Weight	Test Case ID	Test Case Description	Status
1	Log In	HIGH	TC01	Login with invalid username and password	Passed
			TC02	Login without inputting username and password	Passed
			TC03	Login with valid credentials	Passed
			TC04	Login as patient	Passed
			TC05	Login as Front Desk Staff	Passed
			TC06	Login as Nurse	Passed
			TC07	Login as Doctor	Passed
2	Update Information	HIGH	TC08	Patient updates information	Passed
5	Schedule Appointment	HIGH	TC9	Patient schedules appointment	Passed
			TC10	Front desk schedules appointment	Passed
			TC11	Nurse schedules appointment	Passed
			TC12	Doctor schedules appointment	Passed
7	Staff Patient Search	HIGH	TC13	Front desk searches for patient	Passed
			TC14	Nurse searches for patient	Passed
			TC15	Doctor searches for patient	Passed
9	Adding New Patient	HIGH	TC16	Front desk adds a new patient	Passed
			TC17	Nurse adds a new patient	Passed
			TC18	Doctor adds a new patient	Passed
16	Log Out	HIGH	TC19	Click CLOSE on login menu	Passed
			TC20	Patient clicks Exit in application menu	Passed
			TC21	Patient clicks Log Out in application menu	Passed
			TC22	Front Desk clicks Exit in application menu	Passed

			TC23	Front Desk clicks Log Out in application menu	Passed
			TC24	Nurse clicks Exit in application menu	Passed
			TC25	Nurse clicks Log Out in application menu	Passed
			TC26	Doctor clicks Exit in application menu	Passed
			TC27	Doctor clicks Log Out in application menu	Passed

The above chart shows the test cases that will be tested during the first demo. Currently, everything seems to be working and does the task.

The bottom-up integration strategy will be used for this demo. The program is based on restricted access, so to implement that, the log in views of patient, front desk staff, nurse, and doctor were all implemented first. This program is focused on storing patient information and storing it with the access to edit and update information. Patients are also able to schedule their own appointments through this application as well. Staff will also have the capability to schedule appointments for patients. Staff also has the ability to search for a patient and add a new patient to the system. Going through the demo I will start with implementing the patient side of the program then move onto the front desk staff, nurse, and then end with the doctor side.

Section VII – Project Management and Plan of Work

Project Coordination and Progress Report

The following Use Cases have been implemented for the first demo.

Log In
Update Information
Schedule Appointment
Staff Patient Search
Adding New Patient
Log Out

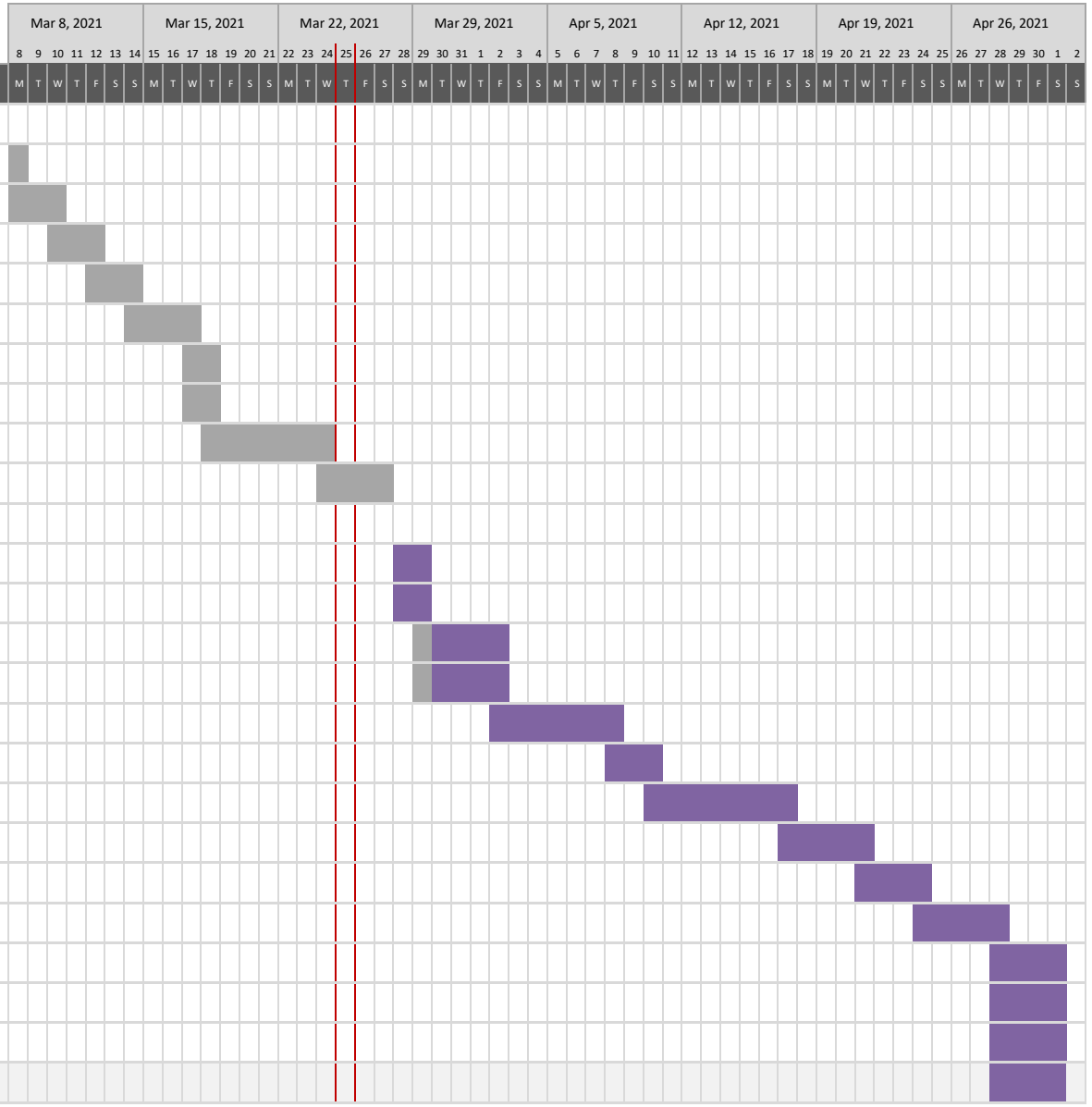
All of the above use cases are fully functional. After the live demo, we will be moving on to cancelling appointments by patients and staff members, which is currently being worked on. There are also some minor kinks that are being worked out including the search patient feature providing any patients with similar names instead of being exact with the search. Also, when a user logs in the log in screen stays hidden behind the program. I want that to be able to close fully, so that is also in the works. I also want to put restrictions on the text fields like sizes and alerts where the information has to be filled out in order to proceed.

Plan of Work
Patient Tracker

Siara Saylor

ject Start: Mon, 3/8/2021

lay Week: 1



TASK	PROGRESS	START	END
Phase 1			
High Priority Designs	100%	2/16/21	3/8/21
Main screens	100%	3/8/21	3/10/21
Save Data	100%	3/10/21	3/12/21
Pull Up Data	100%	3/12/21	3/14/21
Add Patient	100%	3/14/21	3/17/21
Patient Search	100%	3/17/21	3/18/21
Update Data	100%	3/17/21	3/18/21
Schedule Appointment-patient view	100%	3/18/21	3/24/21
Schedule Appointment-staff view	100%	3/24/21	3/27/21
Phase 2 - After Midterm			
Cancel Appointment – patient	25%	3/28/21	3/29/21
Cancel Appointment – Staff	25%	3/28/21	3/29/21
Reschedule Appointment – Patient	25%	3/29/21	4/2/21
Reschedule Appointment – Staff	25%	3/29/21	4/2/21
Staff Update Appointment	0%	4/2/21	4/8/21
Prior Appointments	0%	4/8/21	4/10/21
Design all MED Priority pages	10%	4/10/21	4/17/21
Pre Appointment Page	10%	4/17/21	4/21/21
Medication Refills	0%	4/21/21	4/24/21
Medication Approvals	0%	4/24/21	4/28/21
My Health	0%	4/28/21	5/1/21
Request new Password	0%	4/28/21	5/1/21
Forgot Username	0%	4/28/21	5/1/21
Staff Calendar		4/28/21	5/1/21

References

Nishadha. (2021, March 9). *UML Diagram Types Guide: Learn About All Types of UML Diagrams with Examples*. Retrieved from Creately: https://creately.com/blog/diagrams/uml-diagram-types-examples/
This reference assisted me in researching different types of UML diagrams other than sequence diagrams. I found Activity Diagrams and was able to implement those to go along with each sequence diagram.

Wikipedia. (2021, Feb 24). *SOLID*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/SOLID
This reference was used to be able to research more design principles. I was able to research which ones I could use for my project and more details on how to implement them.