Linnéuniversitetet
Kalmar Växjö

2DV609 -
Project Course in Software Engineering

Final Release

Authors: Adam Rashdan, Kalid Diriye, Nitin Pasikanti, Ramin Jafari & Rashed Qazizada
Supervisor: Mirko D'Angelo & Mauro Caporuscio
Semester: Spring 2020
Group: 4

Contents

1 Introduction

The Final release is a combination of the previously submitted documents, Requirements Document and Design Document.
This document contains the updated information about the application which include Requirements, Test Cases, software architecture, design details, UML diagrams. More detailed description of the architecture and system components will be described throughout subsequent sections of the document.
The source code for the web application is also provided in a separate file.

2 Topic

This project is about creating a web application for booking medical appointment. By using this application, users can virtually book an appointment to their nearest Medical centre.
The application is named as 'Medical Appointment'.

2.1 Description

The application provides a time-saving solution to the end-customers which is designed to provide customers an opportunity to book an appointment without the need of physical presence. The application is applicable and it can supports all browsers for instance Google Chrome, Firefox or Opera…

The application is simple and has a user-friendly layout. The web application comes with a self-test for Covid-19 which helps users perform an analysis of their present health condition by going through a set of questions. After the questionnaire is complete a brief report is presented stating if the user is Coronavirus positive or not based on the symptoms and the required solutions are shown for each of the two cases.

3 Purpose

The purpose of this document is to provide a detailed description about how the web application is constructed. This document demonstrates that design meets the requirements specified in the Project Requirements documentation as well as the Design document.

4 Stakeholders

- Teachers- Test the application and grade the project
- Customers/ Peers- Use the application for appointment booking
- Admin/ Secretary- Keep a track of the bookings made by the user as well as appointment doctors to patients.

5 Tools

- React- The framework use to create the web application. It is a part of the JavaScript library for building user interfaces.

- WebStorm/ VS Code- The editor for writing and compiling the JavaScript code.
- Firebase- for storing the admin and user personal information like login details as well as the user's bookings.
- JMT [Java Modeling Tool]- to create Queuing Network Model and for Workload Analysis.

6 Functional Requirements

Requirement 1: Registration of new members

- New customers/ members can become a part of the virtual world of Medical Appointment system by creating an account and gaining privileges like booking an appointment for health checkup.
- Members can also see a list of their bookings.

Requirement 2: Registered members can login to the system

- Once a member is registered into the system, the member can make an appointment for checkup.
- Keep track of the list of appointments the member previously made.
- Registered members can create, read, update and delete [C.R.U.D] an appointment.

Requirement 3: Members can book an appointment for a check up

- The members can book an appointment for check up by logging in.
- The members must pick a date and choose an available time slot to confirm the booking.

Requirement 4: Members can view the list of booking

- Members can view the list of appointments they made after logging in.

Requirement 5: Members can delete a booking

- Members can view a list of their bookings in the homepage and can delete them.

Requirement 6: Members can edit their personal details

- Members can login and view their personal details like username, email address, home address etc.
- Members can edit or update their personal details.
- The details are also updated in the database accordingly.

Requirement 7: Admin can login to the system

- Admin can login to the system and get an overview of the bookings made by the members.
- Admin can also see the list of doctors.
- Admin can assign a user's appointment with a doctor for the medical checkup.

Requirement 8: Admin can view the appointments made by all the users

- Admin can login to the system and get an overview of all the appointments made by the users.

Requirement 9: Admin can assign the user appointment with a doctor.

- Admin gets an overview of the appointments made by users and can choose to assign a member with a doctor

Requirement 10: Admin can delete a booking

- Admin gets an overview of the appointments made by the user and can choose to delete a booking if the nurse for that appointment is not available

Requirement 11: Admin can edit their personal details

- Admin can view thier personal details like username. email address, password etc.
- Admin can edit user details
- The details are also updated in the database accordingly

Requirement 12: Admin can access list of users

- Admin can view the list of users that are registered with the system after logging in.
- The users are shown in a table from first user to the latest user.

Requirement 13: Admin can see the list of doctors

- Admin can view the list of doctors registered with the system after logging in

Requirement 14: Admin can edit the details of the doctors

- Admin can view the list of doctors registered with the system and can edit their details.
- The edited details are saved in the database.

Requirement 15: Invalid user request

- If a user or admin makes a request to a page that does not exist, then the webpage throws an error request 404, Page not found.

Requirement 16: Self-test for Covid-19

- The homepage of the web application comes with a test for covid-19
- The page takes the user though a list of questions where user addresses his present health condition to find out if he has symptoms of corona
- Towards the end of the test a brief report is shown about the precautions he must take if the user is not infected or the health line number that he must call if he is infected.

7 Non- functional requirements

1. The login page shows an error message if the incorrect login details are entered
2. The webpage should be simple, easy to use and navigate.
3. While booking an appointment, the application must only show available time slots for booking.
4. After booking an appointment, the application must show a message that the booking was successful.
5. The components of the webpage must be scalable with the webpage. like the navbar, sidebar, footer etc.
6. The components used to build the application must be reusable in other areas of the application.
7. When the application loads, the present date is shown on the webpage.
8. The application must load within in 3 seconds.
9. The user gets a notification to his email address after a successful appointment booking.

8 General priorities

The priority guide is sorted by hierarchy, with a strong focus content-first, from top to bottom and without layout specifications. Figure below shows the structure.

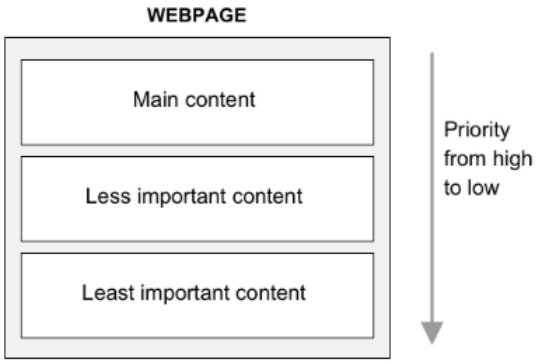


Figure 1: Structure of priority guide

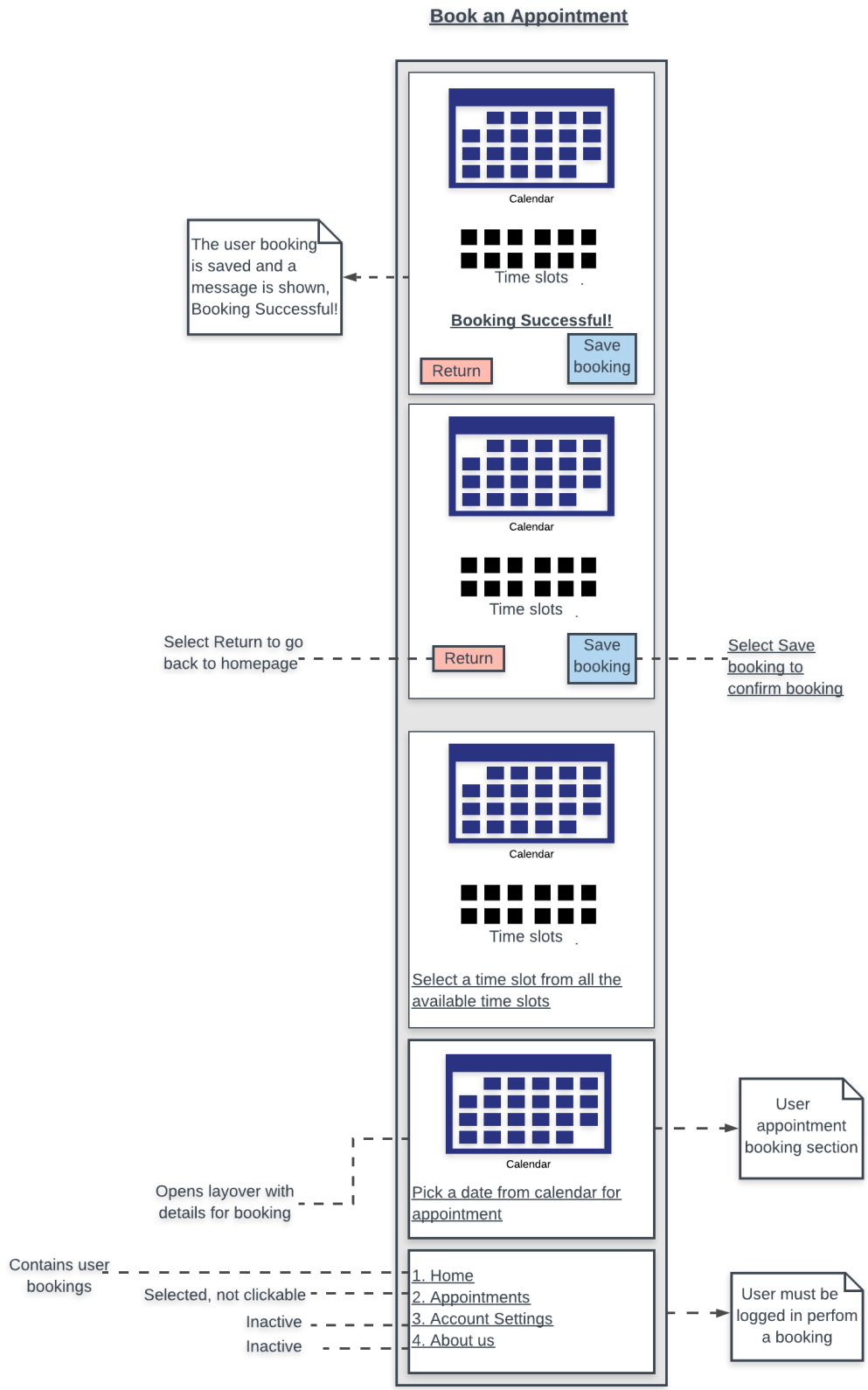
We believe that the priority goals of this project is to implement the functional requirements of the application.

To do so, the user has to follow some steps which can be of a lower priority and then followed by higher priority steps.
For instance, if the user has to create an appointment the steps below must be followed:

1. Open the list of options in the sidebar of userpage
2. Then choose ‘Book an Appointments’ option.
3. Select a date for appointment
4. Select a time slot
5. Finally, choose save changes button.

The initial steps are least important so they fall under the lower part of the figure mentioned above while the main content which is booking an appointment is shown in the top most part of the figure .

To explain the priority in detail, we implement the Functional Requirement 3 which is Book an Appointment. The figure flows from bottom (least important content) to top (main content)


Book an Appointment
Calendar
Time slots
Booking Successful!
Return
Save booking
The user booking is saved and a message is shown, Booking Successful!
Calendar
Time slots
Select Return to go back to homepage
Return
Save booking
Select Save booking to confirm booking
Calendar
Time slots
Select a time slot from all the available time slots
Calendar
Pick a date from calendar for appointment
User appointment booking section
Opens layover with details for booking
Contains user bookings
Selected, not clickable
Inactive
Inactive
1. Home
2. Appointments
3. Account Settings
4. About us
User must be logged in perfom a booking

9 System Overview

This section describes the system's Design details like the design goals, functionality and architecture. It also includes a high level description of the approach used to develop the system design.

Goals

Towards the beginning of the project we faced challenges and shortcomings in managing the team, deciding the topic for the project, software, programming language, tracking the time spent and reporting. Finally, we decided to work on creating a web application that can be used for medical appointment booking and this is based on React framework.

We had no knowledge of React and everything was new to us. This was one of the main challenges. Therefore, we had to spend quite a lot of time trying to learn and understand how react works.
So, we decided that the main goal is to first implement the functional requirements for the application and then add extra features upon that.

Functionality

The web application can be used to virtually book appointments for medical chekup and we believe that an approach like this can improve the quality of service and save time.
The application is compatible with most of the web browsers like Chrome, Firefox, Opera to name a few.

Since the application runs on React framework and supported by JavaScript, it is compatible with IDE's such as WebStorm, VS Code etc. The user data in this application is stored in the Firebase and can be retrievd when necessary for instance, to login or to fetch bookings.

One of the primary benefits of React is, it significantly reduces page load time through faster rendering speed, adapts it performance in real-time based on current user traffic, features that are otherwise not perfectly handled by most frameworks.

Architecture

The application implements the Client-Server architecture. For the application to run, the server should be up and running on a specified port.

Code structure for User Interface

- The code base was the most tricky part to handle as we have more than 20 classes.
- We decided to re-architect the code by creating various modules. The architectural pattern that was implemented was Multi-layer pattern. In this pattern we diviede the code into different layers/ modules and each corresponds to a different service or integration.
- Since each layer is separate, it is easy for us to control and make changes in the web application than having to tackle the entire architecture.
- The classes were divided into the following layers/modules as shown below:

I. Assets: for images used in the webpage
II. Components : all the parts that make up the webpage
    - Admin
    - Authentication
    - Main page
    - Navigation
    - User
III. Containers : The application
    - App
IV. Database : The data that is entered in the webpage. It is a backend component
    - Firebase

- All the dependences of the code are implemented in the package.jason file.
- We installed various libraries from React while creating the website. They are:
    - react-icons--save
    - burger-menu
    - calendar
    - firebase--save
    - npm install

- For the application styling, we used tachyons css frame work along with the normal css.

Backend structure (Database)

- The back-end part of the website is implemented using Firebase.

- Firebase is form Google which provides applications with features like authentication, hosting of apps, cloud storage and realtime database.
- In the firebase, we created various collection which include:
    I. Users: contain the details of the user like first name, last name, password, email address, phone number, personal number and house address.
        - The user email address and password was used for logging into the user account.
        - The Admin's details are stored within the users collection

    II. Booking
        - The user's appointments are stored here.
        - They include details like date of booking, reason for booking and time slots chosen for booking.
        - The user booking is shown after logging in to the user account in the webpage
    III. Nurses
        - The Nurse's details are also stored within the users collection

Design Principles

The following design principles were used to guide the code structure of the application.

1. Divide and conquer
    - The workload was equally divided among the team members and each person worked on a different component.
    - Each individual subsystem is smaller and hence it was easier for us to understand, replace or change the code.

2. Increase cohesion where possible
    - A module keeps the components together that are related to each other
    - It keeps out the other things

3. Increase reusability where possible
    - Almost all the aspects of the system can be used again in other contexts.
    - The design is simplified as much as possible.

4. Reuse existing designs and code where possible
    - All the designs were reused and hence it saved a lot of time.

5. Design for Portability
    - The application is simple and easy to use. It is compatible with most web browsers.
    - It works on all the OS

6. Design for flexibility
    - The code is designed such that it can undergo change in the future.
    - There is cohension.
    - Code reusability etc

10 Design constraints

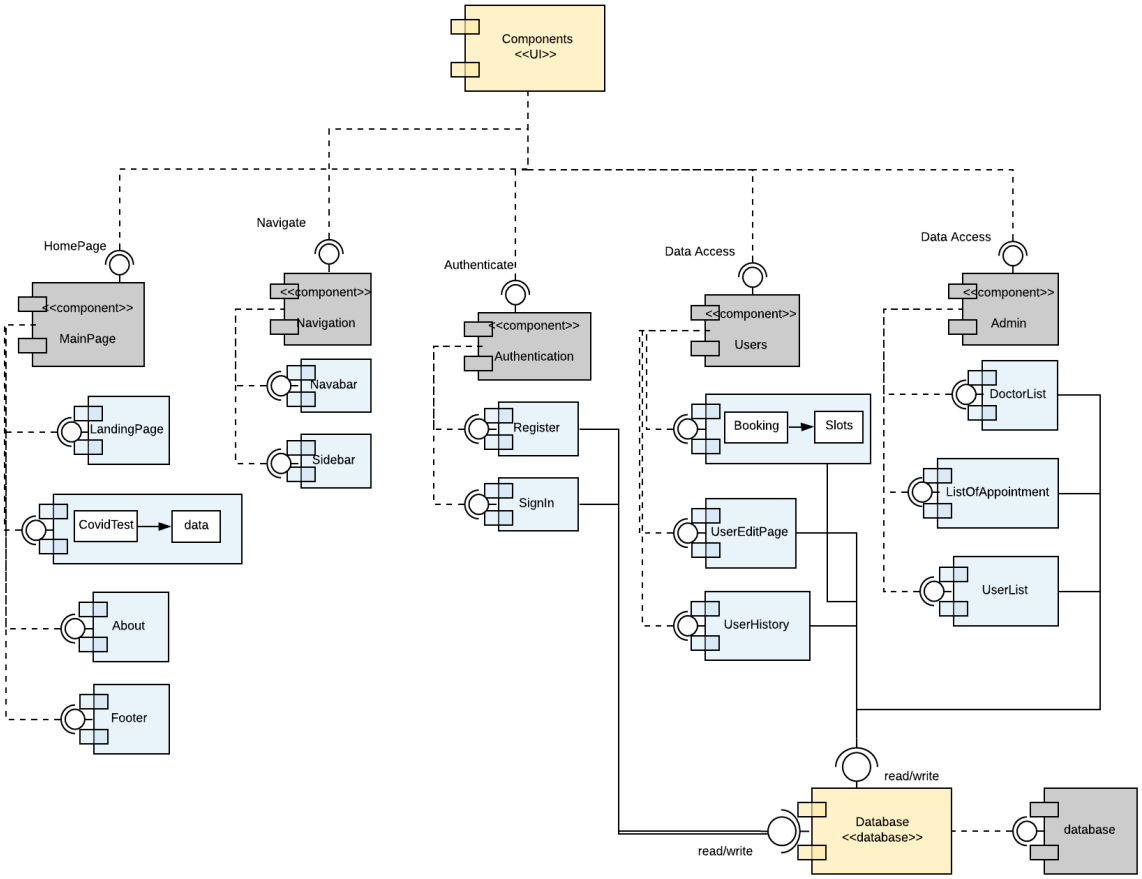
We identified several constraints while designing the project. To date, the following constraints have been identified:

- Decision about the project topic
    - Initially we implemented an Android application but after the second iteration we decided to implement web application
    - The time frame to design the web application and meet all the requirement was limited as we changed the topic two weeks after the project had started.

- Lack of experience
    - The team was new to React and JavaScript. It took us about 1 week to learn React and how to design web pages using React.
    - However, we managed to finish the project within the deadline.

- The design of the code must be in a mannaer such that it can be easily tested and favours reusability of components within the webpage.

UML Component Diagram



11 Checklist based Requirements analysis

Checklist for Requirements based analysis	Requirements															
	Req 1	Req 2	Req 3	Req 4	Req 5	Req 6	Req 7	Req 8	Req 9	Req 10	Req 11	Req 12	Req 13	Req 14	Req 15	Req 16
Requirements testability	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Combined requirements	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No6
Requirement Gold- Plating	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No
Use of non-standard hardware	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No
Conformance with business goals	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Premature Design	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Requirements realism	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Requirements ambiguity	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No

12 Classification on faceted approach

Classification on Faceted approach	Requirements															
	Req 1	Req 2	Req 3	Req 4	Req 5	Req 6	Req 7	Req 8	Req 9	Req 10	Req 11	Req 12	Req 13	Req 14	Req 15	Req 16
System	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
User interface	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Database	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Communication	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No

13 Systematic Validation of the requirements

Requirements Validation	
Are the requirements complete?	YES
Are the requirements consistent?	YES
Are the requirements comprehensible?	YES
Are the requirements ambiguous?	NO
Is the requirements document structured?	YES
Are the requirements traceable?	YES
Does the requirements document as a whole?	YES

14 Test Cases

14.1 Requirement

Requirement 1: Member Registration	
Test Case ID	TC1
Test Priority (Low/Medium/ High)	High
Test Title	Successfully register a new customer
Description	• New members can become a part of the virtual world of Medical Appointment system by creating an account and gaining privileges like booking an appointment for health checkup
Pre-condition: User has opened the sign-up form	
Test Steps	1. Open the Sign-up form on the homepage of the application 2. You are presented with empty blanks to fill in for registration. 3. Type in a valid Username 4. Provide a valid email address 5. Provide a valid personal number 6. Provide a valid password 7. Provide your house address 8. Click the Register button
Expected Result	The member is registered in the database and is presented with a confirmation message to his email address
Status (Pass/ Fail)	Pass
Comments	• Registration passes only if valid input is provided. **Example**: • The username should not contain numbers or special characters • The email address should be valid

14.2 Requirement

Requirement 2: Registered members can login to the system	
Test Case ID	TC2
Test Priority (Low/Medium/ High)	High
Test Title	Verify user login with a valid username/ email address and password
Description	Once a member is registered into the system, the member can make an appointment for checkup.
Pre-condition: User is registered within the system and has a valid username and password	
Test Steps	1. Open the application and choose the sign-in button 2. The login page is displayed 3. Provide the valid username or email address 4. Provide valid password 5. Click the login button
Expected Result	The user should be able to login to his account successfully and is navigated to the dashboard
Status (Pass/ Fail)	Pass
Comments	• The user is validated with the database and successfully logged in.

14.3 Requirement

Requirement 3: Members can book a time slot for a checkup	
Test Case ID	TC3
Test Priority (Low/Medium/ High)	High
Test Title	Book a time slot
Description	Test if a customer can book a time slot for a checkup.
Pre-condition: User has logged into his account	
Test Steps	1. Follow TC2 2. The dashboard is presented with 3 options 3. Select the 'Book' option. 4. You are presented with a calendar containing a color scheme. The calendar is shown in a weekly format along with date and time. 5. The colors represent the following • Red for not available. • Orange indicating that only a few slots are available. • Green indicating that about 70% of the slots are available for booking. 6. Select an available slot by choosing the date and time of your choice. 7. After choosing the slot you are redirected to the description page. 8. Write a brief description explaining your health issue in less than 50 words. 9. Click the submit button
Expected Result	The system registers the booking and sends a message to the medical center and a confirmation email to the customer.
Status (Pass/ Fail)	Pass
Comments	None

14.4 Requirement

Requirement 4: Members can view the booking	
Test Case ID	TC4
Test Priority (Low/Medium/ High)	High
Test Title	Check if a member can view the booking
Description	Test if a customer can view the booking after he is logged in to his account
Pre-condition: User has booked an appointment	
Test Steps	1. Follow TC2. 2. The user is taken to the homepage 3. The homepage contains the booking made by the user. 4. The home button takes user to the booking they made.
Expected Result	The booking made by the user is shown on the homepage
Status (Pass/ Fail)	Pass
Comments	None

14.5 Requirement

Requirement 5: Members can modify or delete a booking	
Test Case ID	TC5
Test Priority (Low/Medium/ High)	High
Test Title	Check if a member can modify or delete his booking
Description	Test if a customer can change or delete his previous booking after he is logged in to his account
Pre-condition: User has booked appointment previously	
Test Steps	1. Follow TC2. 2. The dashboard is presented with 3 options. 3. Select 'Modify or Delete a Booking' option. 4. From the list of the bookings, select the booking that you want to modify/delete. 5. Modify the booking by changing the date and time. 6. Delete a booking by choosing the delete button. 7. A confirmation dialog box appears. 8. Click the save changes button.
Expected Result	A particular booking is updated/ modified according to the user's requirement or it is deleted if the user chose to delete it
Status (Pass/ Fail)	Pass
Comments	• The changes made by the user are also updated in the database.

14.6 Requirement

Requirement 6: Members can edit their details	
Test Case ID	TC6
Test Priority (Low/Medium/ High)	High
Test Title	Check if a member can edit their details
Description	Test if a customer can edit his personal details after he is logged in to his account
Pre-condition: User is logged in	
Test Steps	1. Follow TC2. 2. The dashboard is presented with 3 options. 3. Select ‘Account Setting’ option. 4. The user details are presented which include first name, last name, email etc. 5. Select the update button 6. A form is displayed where the details can be edited 7. Enter the new details 8. Click the save changes button.
Expected Result	User details are updated and the new details are shown
Status (Pass/ Fail)	Pass
Comments	• The changes made by the user are also updated in the database.

14.7 Requirement

Requirement 7: Admin/ Secretary can log in to his account	
Test Case ID	TC7
Test Priority (Low/Medium/ High)	High
Test Title	Verify admin login with a valid username and password
Description	Test admin login into the medical booking system.
Pre-condition: Admin's details registered with the system and has a valid username and password	
Test Steps	1. Open the application and choose the sign-in button. 2. The login page is displayed 3. Provide a valid username or email address.. In this case it is admin@admin.com' 4. Provide valid password. In this case it is 'password' 5. Click the login button.
Expected Result	The admin should be able to login to his account successfully and is navigated to the dashboard.
Status (Pass/ Fail)	Pass
Comments	• The admin is validated with the database and successfully logged in. • The admin can change the bookings made by the users

14.8 Requirement

Requirement 8: Admin can view the appointments made by all the users	
Test Case ID	TC8
Test Priority (Low/Medium/ High)	High
Test Title	Verify if admin can view the appointments made by all the users
Description	Admin can view the appointments made by all the users
Pre-condition: Admin is logged in	
Test Steps	1. Follow TC7 2. The admin is directed to the homepage where he can see all the appointments made by the users.
Expected Result	The admin should be see all user appointments.
Status (Pass/ Fail)	Pass
Comments	• The user bookings are retrieved from the database

14.9 Requirement

Requirement 9: Admin can assign user appointment with a doctor	
Test Case ID	TC9
Test Priority (Low/Medium/ High)	High
Test Title	Verify if admin can assign user appointment with a doctor
Description	Admin can assign user appointment with a doctor
Pre-condition: Admin is logged in	
Test Steps	1. Follow TC7 2. The admin is directed to the homepage where he can see the list of appointments made by user 3. Towards the right side of every appointment, there is a Assign a doctor button. 4. Select that button on the desired appointment. 5. From the dropdown menu select a doctor that you would like to appoint. 6. Select the save button.
Expected Result	A doctor is appointed to the selected patient
Status (Pass/ Fail)	Pass
Comments	• Admin is validated with the database and successfully logged in • The appointments are updated in database

14.10 Requirement

Requirement 10: Admin can delete a booking	
Test Case ID	TC10
Test Priority (Low/Medium/ High)	High
Test Title	Verify if admin can delete a booking
Description	Admin can delete a booking made by the user if the doctor for that appointment is not available.
Pre-condition: Admin is already logged in	
Test Steps	1. Follow TC7 2. The admin is directed to the homepage where he can see the list of appointments made by user 3. Towards the right side of every appointment, there is a delete button in red color. 4. Select that button on the desired appointment.
Expected Result	The selected appointment is deleted from the list
Status (Pass/ Fail)	Pass
Comments	The booking list is also updated in the database.

14.11 Requirement

Requirement 11: Admin can edit his/her personal details	
Test Case ID	TC11
Test Priority (Low/Medium/ High)	High
Test Title	Verify if admin can edit his/her details
Description	• Admin can edit his/her details after logging in
Pre-condition: Admin is already logged in	
Test Steps	1. Follow TC7. 2. The dashboard is presented with options. 3. Select 'Account Settings' option. 4. The admin details are presented which include first name, last name, email etc. 5. Select the update button 6. A form is displayed where the details can be edited 7. Enter the new details 8. Click the save changes button.
Expected Result	Admin personal details should be changed
Status (Pass/ Fail)	Pass
Comments	• Admin's details are also updated in the database accordingly
Requirement 12: Admin can access list of users	
Test Case ID	TC12
Test Priority (Low/Medium/ High)	High
Test Title	Verify admin can access list of users

Description	Admin gets an overview of all the users registered in system
Pre-condition: Admin is already logged in	
Test Steps	1. Follow TC7. 2. The dashboard is presented with options. 3. Select ‘Users List’ option. 4. The admin is presented with list of all the users and their details who are registered in the system.
Expected Result	The admin views all the users registered in the system.
Status (Pass/ Fail)	Pass
Comments	• The user details are retireived from the database.

14.13 Requirement

Requirement 13: Admin can see list of doctors	
Test Case ID	TC13
Test Priority (Low/Medium/ High)	High
Test Title	Verify if admin can see list of doctors
Description	Admin can see the list of doctors
Pre-condition: Admin is already logged in	
Test Steps	1. Follow TC7 2. The admin is directed to the homepage 3. From the sidebar select the ‘Doctors List’ option 4. The doctors registered in the system are shown
Expected Result	The admin views a list of all doctors registered in the system
Status (Pass/ Fail)	Pass
Comments	The doctor list retrieved from the database.

14.14 Requirement

Requirement 14: Admin can edit the details of the doctors	
Test Case ID	TC14
Test Priority (Low/Medium/ High)	High
Test Title	Verify if admin can edit the details of the doctors
Description	Admin can edit the details of the doctors
Pre-condition: Admin is already logged in	
Test Steps	1. Follow TC7 2. The admin is directed to the homepage 3. From the sidebar select the ‘Doctors List’ option 4. The doctors registered in the system are shown 5. Select the ‘edit’ button towards the right side of the doctor's name. 6. A form is displayed showing the details of the user that can be changed. 7. Enter the required details 8. Select the update button.
Expected Result	The respective doctor’s details are updated.
Status (Pass/ Fail)	Pass
Comments	• The doctor’s details are also updated in the database. • Updated details are shown when requested again.

14.15 Requirement

Requirement 15: Invalid user request	
Test Case ID	TC15
Test Priority (Low/Medium/ High)	High
Test Title	Page not found
Description	If a user or an admin makes a request to a page that does not exist, then the webpage throws an error request 404, Page not found
Pre-condition: booking system website is up and running	
Test Steps	1. Open the application and homepage is displayed 2. The search box on the top of the page shows the path of the current page we are in 3. Change the path to a random name or to a path that is not present in the code. 4. The application displays a message showing. 'Error 404, Page not Found'
Expected Result	Page not found errors will be shown
Status (Pass/ Fail)	Pass
Comments	The user provided an invalid path

14.16 Requirement

Requirement 16: Self-test for Covid-19	
Test Case ID	TC16
Test Priority (Low/Medium/ High)	High
Test Title	Self-test covid-19
Description	Homepage loads with Covid-19 test
Pre-condition: booking system website up and running	
Test Steps	1. Homepage. 2. Click on “Do an online test 3. List of questions are shown regarding the symptoms of Covid-19 4. Select suitable answer based on your health condition 5. Click next 6. Repeat 4 and 5 7. At the end, a brief report will be shown about the precautions/measures to take if he has a possible symptom of Coronavirus
Expected Result	The user will see the test results and an opportunity to do the test again
Status (Pass/ Fail)	Pass
Comments	• Unregistered users can also take the test

15 Performance modeling and evaluation

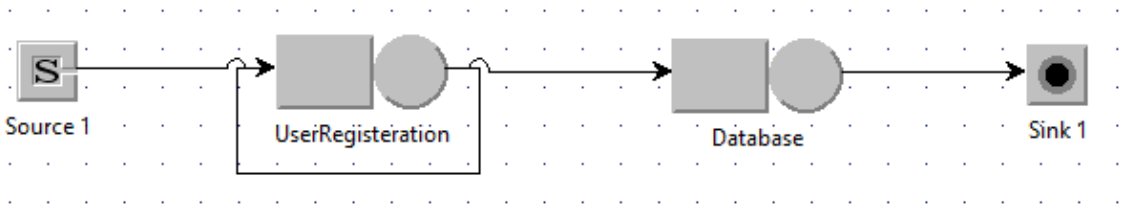
The performance evaluation was done against one functional requirement to test how the system would behave. We used JMT to show the workload analysis and to create queuing network model.

Reason why we measure against one functional requirement?

- The system as a whole cannot be shown in queuing network model because it would increase the complexity.
- If the entire system is taken, the workload results are hard to produce because of the problems with probabilistic routing.

Therefore, this analysis can be performed on any functional requirement.
We chose one functional requirement and gathered all the components it would need to perform a user request. Then we ran the simulation in JMT and obtained the results which are presented below.

Model



Scenario

The model above represents the functional requirement of User Registration.
A new user makes a request for registering into the system, he is directed to the User Registration server where the user enters correct details 90% of the time and then the request goes to the Database server where the detailes are stored for later user like signing into the system. But if the user enters incorrect details in the User Registration server which happens 10% of the time then it has a self loop. Finally, after handling the request it finishes the process.

Worload Analysis

- All the values taken here are based on assumptions.
- Since, this application was within the scope of the course, we assumed that the number of users that can login to the application at a specific time would be just one.

User Registration

Arrival rate (λ)= 1/second [Number of requests that arrive per second]

Service time (S)= 0.75 [Time taken to complete one request]

Resources= 1 [Number of resources of the server]

Probabilistic routing= 0.9 towards Database if all details are correct
0.1 returns if there are errors in user details

Database

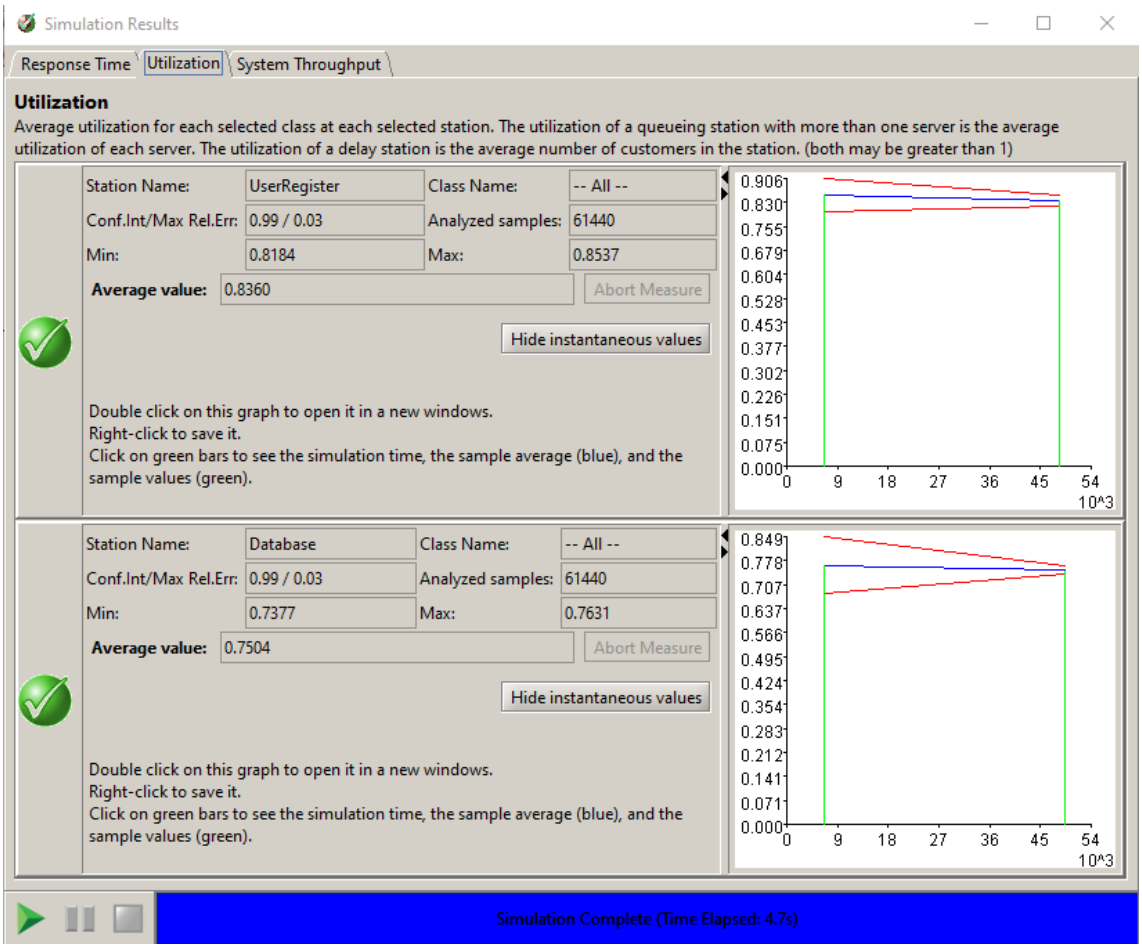
Arrival rate (λ)= 1/second [Number of requests that arrive per second]

Service time (S)= 0.75 [Time taken to complete one request]

Resources= 1 [Number of resources of the server]

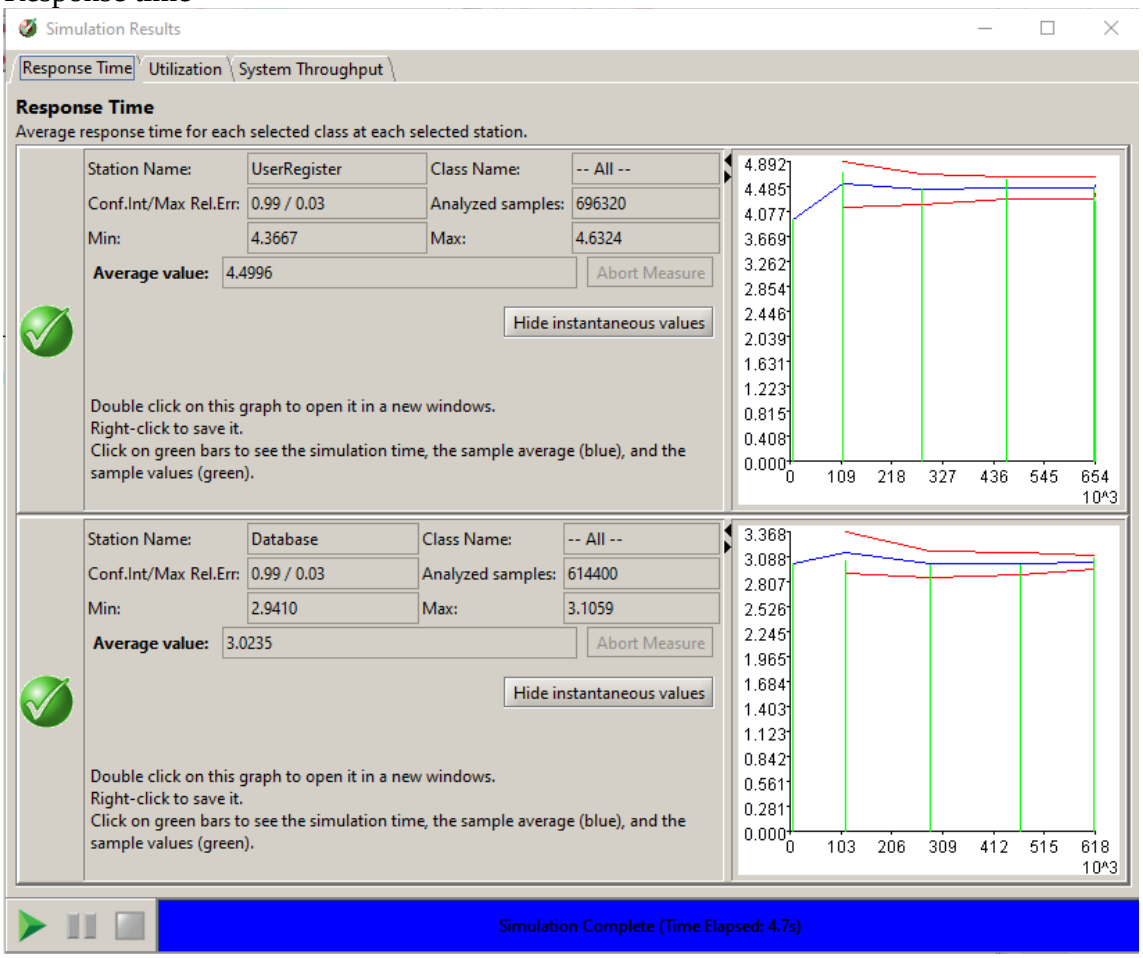
Simulation results

Utilization



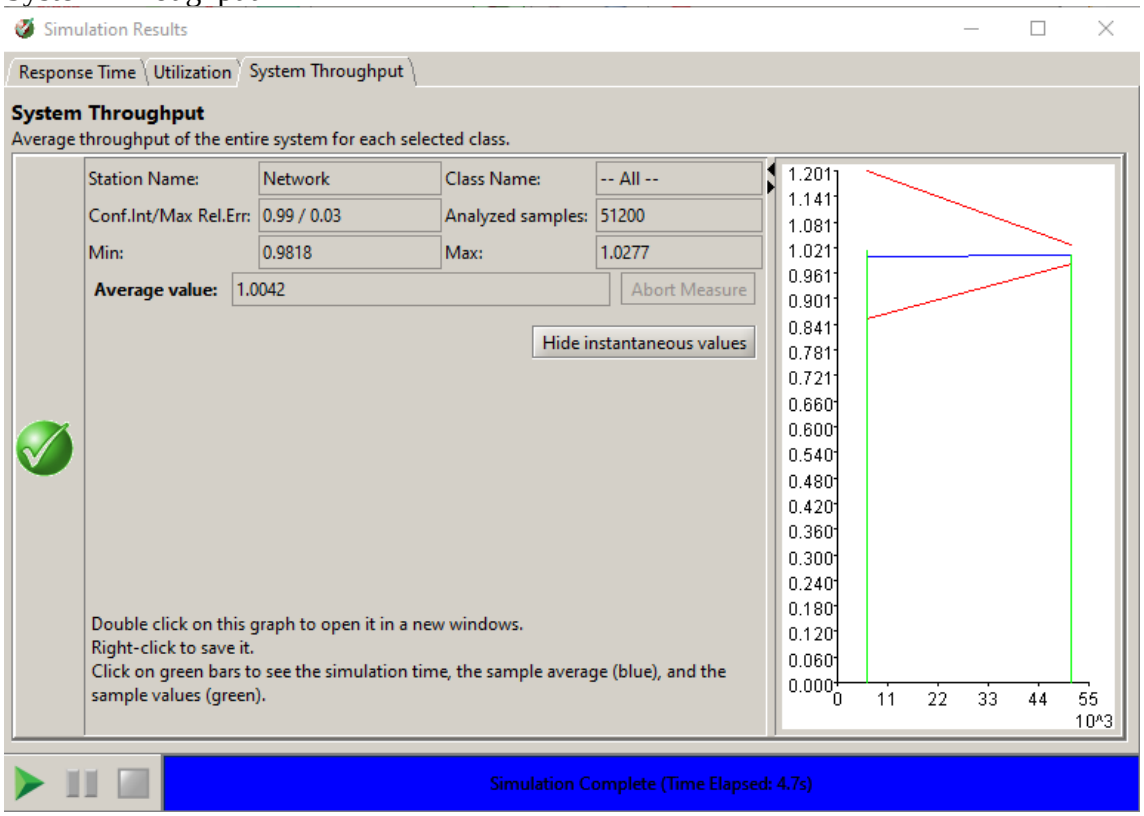
In the above diagram we observe that the utilization of the User Registration server is 0.8 which is nearly 80%. The utilization of the User Registration server is 0.75 which is nearly 70%.

Response time



In the above diagram we observe that the response time (time taken to respond to a user request) of the User Registration server is 4.49 and that the response time of the User Registration server is 3.02.

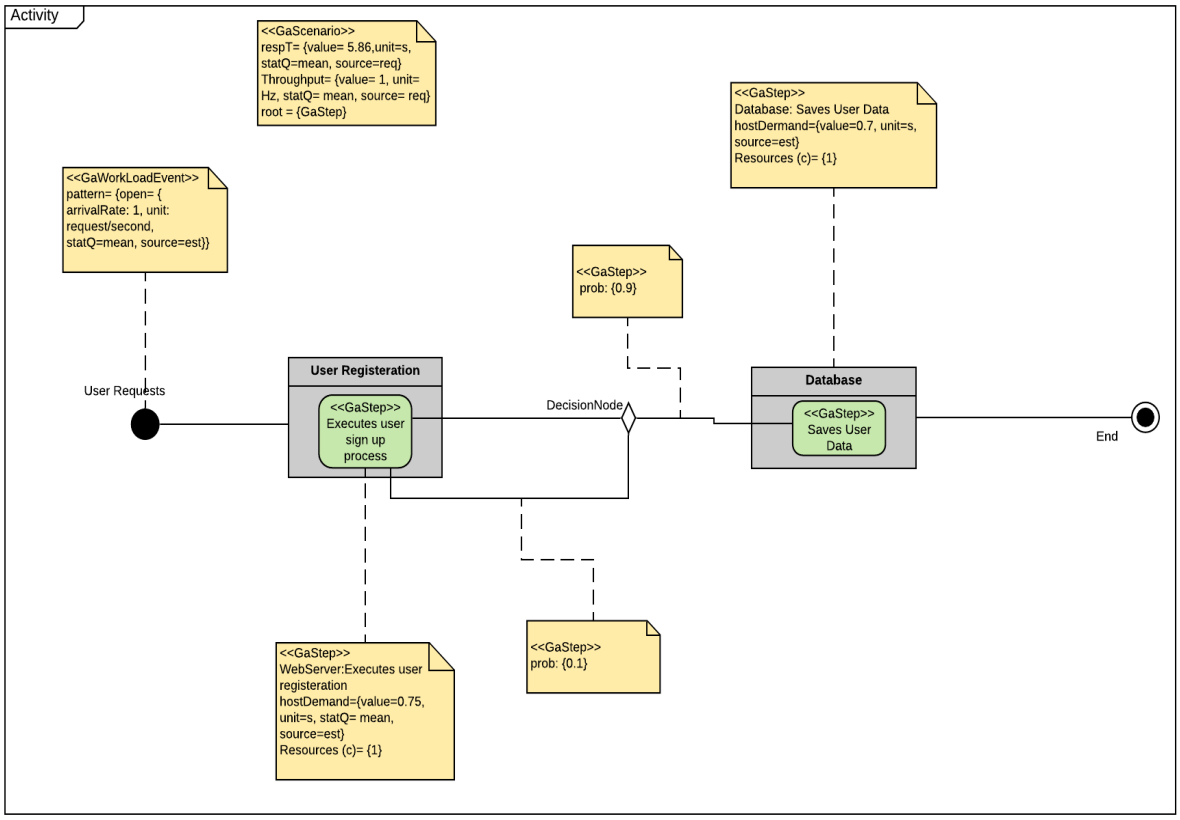
System Throughput



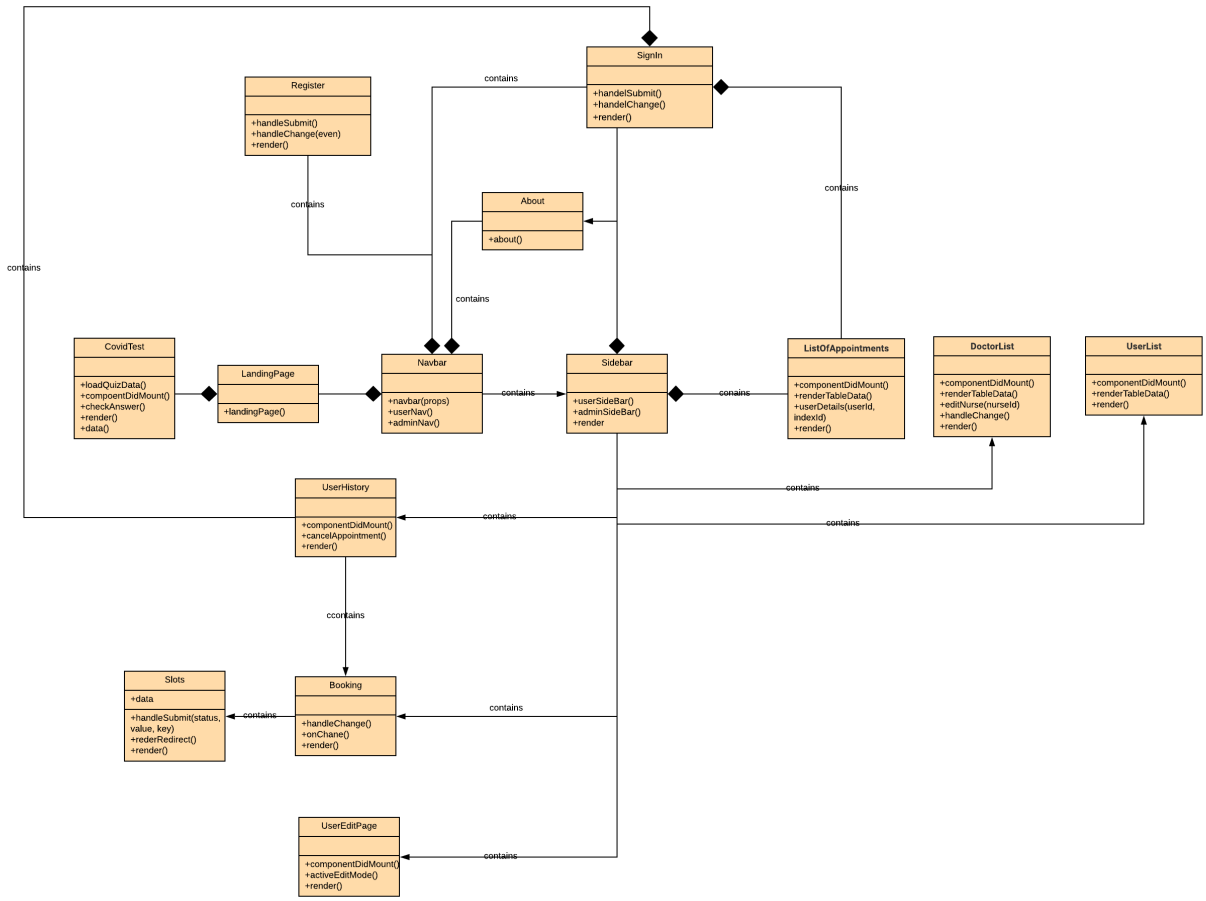
In the above diagram we observe that the system throughput (number of requests completed in one second) is 1.0042

MARTE

The activity diagram below is for user registration. The performance of the system is shown below. We see that the hostDemand for the UserRegisteration and Database is 0.75. While the response time and the utilization of the system are shown in the above pictures.



16 UML Class Diagram



In the above class diagram, we observe that the class diagram has only associations relationship between the classes. This is because in the code base we only import one class within another class and hence, this does not cause dependencies between the classes.

THE END