Middle East Technical University Northern Cyprus Campus

Computer Engineering Program

CNG491 Computer Engineering Design I

Final Draft

Vaccination Queuing and Management System

Güray Gürkan- 2151967

Ege Erdem- 2328649

Burak Eken- 2352045

Anthony Ikeoffiah Chukwudumebi- 2269447

Supervised by

Idil Candan

Contents

Chapter 1
Introduction

1.1 Problem Formulation

For about two years now, the world has been facing a current pandemic (COVID-19). During this time, various Countries and companies have developed and distributed different types of vaccines to stop the spread. WHO (World Health Organization) has stated the multiple guidelines that individuals should follow to protect them self and their loved ones. Which are:

- Get vaccinated as soon as it's your turn and follow local guidance on vaccination.
- Keep a physical distance of at least 1 meter from others, even if they don't appear to be sick. Avoid crowds and close contact.
- Wear a properly fitted mask when physical distancing is not possible and in poorly ventilated settings.
- Clean your hands frequently with an alcohol-based hand rub or soap and water.
- Cover your mouth and nose with a bent elbow or tissue when you cough or sneeze. Dispose of used tissues immediately and clean hands regularly.

The problem is that most vaccination centers find it difficult to manage the queue in there various center, in some cases some vaccination center are overcrowded which provides an event for the spread of the Covid-19.and other vaccination centers have small number of people. This is due to lack of a good queuing system.

Q is a mobile and tab application that aims to help with the vaccination centers have a good queuing management system to help control the number of people at a given center at a particular time, it will also be used by individuals to easily locate various vaccination center close to them. Due to the wild spread of COVID-19, many people are overloading their vaccination centers, violating the second guideline point of physical distancing. We see our Q application as a solution to this problem, as it contains a queuing management system to help with this specific problem.

1.2 Background Problem

Queuing management system has been used for long period of time and has been applied in various ways, for example when individuals go to the bank there is a queuing system that helps the bank to know which customer to serve next. We are using this queuing System, directly integrated with our mobile application to properly group and manage population size in the various vaccination Center. Lastly the queuing System being on a mobile application is of significant improvement for the Queuing management system as it makes it more flexible for the users to manage their queue, from selecting which queue to enter or deciding if to leave a queue. This queuing system will also have proper time integration to give a user an approximated time for their number in the queue to pop up, based on the calculated time of servicing on a particular vaccination center.

1.3 Aims and Objectives

Q aims to make the vaccination process more efficient for the vaccine center customers and the vaccine centers, thereby helping to get more people vaccinated while maintaining all the necessary guide line outline by WHO(World health organization).

- By providing Queuing Management system to the vaccination centers to assist them in managing the number of individuals at their centers at a particular time, thereby helping them to maintain social distancing among their customers.
- By providing the vaccine customers, with a list of vaccination centers close to them, and the number of customers waiting in the queue at a given time.
- By providing them with an average waiting time on the queue, and notification when their queue number is about to be reach so as to help them easily follow the queuing process while maintaining proper social distancing.
- By providing daily notification to the unvaccinated to encourage them to get vaccinated.
- By providing information to the vaccine customer on the various vaccine present in a given vaccination center, thereby helping to improve the transparence in the vaccination process.

1.4 Stakeholders

- Vaccine Center Employees

- Vaccine Center Customers

Chapter 2
Requirements Specification

2.1 Functional System Requirements

1 Customers shall be able to register to the system.

- Customers shall use their social security number, phone number, and password to register.
- The system shall check if the password is strong enough.
- The system shall look through its database and check if the social security number and phone number exist on the system.
- If not, the system shall verify the phone number by sending a message and link to the social security number for notification purposes.
- If phone number verification is successful, the system shall search the national database to find the citizen according to the citizen's social security number.
- If found, the system shall store the password to the database using a one-way hashing algorithm and link to SSN. If the operation is successful, then the system shall finish the registration process successfully.

2. Customers shall be able to log in to the system after registration is completed.
    - Customers shall use their SSN and password to log in.
    - The system shall check through its database to see whether the given SSN exists.
    - If the SSN exists, the system shall check whether the hashed password is matching with the database.
    - If it is, then log in the user to the system.
3. The system shall display whether the customer has the right to be vaccinated.
    - If the customer has not, then the system shall provide a warning message to the user and disable queue operations.
    - If the customer is vaccinated, the system shouldn't allow entering the queue system.
4. The system shall provide a map component to see their location and the vaccine centers close to them.
    - The system shall ask for the necessary Android permission to display the user's location on the map.
    - The system shall center the map component around the latitude/longitude of the user.
    - The system shall get the latitude-longitude info of every vaccine center to place pins on the map.
    - The system shall add pins on the map for each vaccine center.
    - The system shall calculate the average duration between vaccine center locations and the customer location

5. Customers shall be able to see the list of vaccine centers that they can visit.

- The system shall display the distances and average reach durations between vaccine center locations and the customer location.
- The system shall query the registered vaccine centers in the database sorted by distances in ascending order.
- The system shall list all the vaccine centers in a list according to the user’s city-based location.
- Users shall see which vaccine centers are currently operational and each vaccine centers ’working hours from the list.
- Users shall be able to see the available vaccine types for each currently active vaccine center.
- Users shall see how many people are in the queue and the estimated waiting time supposing s/he enters the line.

6. The system shall display the details of the vaccine center when a vaccine center is selected.

- The system shall display how many people are currently in the queue for the selected vaccine center.
- The system shall calculate and display the average waiting time that the customer will wait to suppose s/he enters the queue.
- The system shall provide a list of possible operations that the customer can do in the vaccine center.
- Once an operation is selected, the system shall provide a button to click and enter the queue for the selected vaccine center.

7. Once a customer is in the queue, they shall see their current status in the vaccine center.

    - The queue shall be in the form of a priority queue where the priorities are distributed based on the emergency of the provided operations
    - The system shall display the current position of the customer and update anytime when a change in the status of the queue occurs.
    - The system shall display the estimated remaining waiting time of the customer depending on the current position and update it continuously

8. The vaccine center shall be able to manage which customers are in the queue
    - Once a Customer has selected a given vaccination center, the system shall update the queuing list of the chosen vaccination center.
    - Once the Customer has been put on the queuing list of the vaccine center, the vaccine center can remove them from the queue or mark them as done (to show that they have been attended to).
    - The vaccine center can decide to fix the maximum number of customers that can be accepted into the query at a given time.
9. The vaccine center shall be able to manage their working days.
    - A vaccine center can select the various days they would like to be operational from the calendar view system.
    - They will be able to set the time and date in any given month based on their decisions.
10. The vaccine center shall manage the profile of their center in the System and the type of vaccine they administer.
    - A vaccine center will be able to change their profile details from their names to their profile image.
    - They will also be able to add and remove the vaccine type they administer.

11. The system shall provide a button to the customer where s/he cancel the operation and leave the queue
12. The system shall provide estimated waiting time to the customer and send notification to the user

2.2 Non-functional Requirements

1. The customer shall not be able to join more than one queue at the same time.
2. The system shall be available all the time regardless of the number of customers running the application.
3. If the person is already vaccinated, the system shall not make an appointment for the user.
4. Response time should be less than 500 ms to the users for better user experience.

2.3 Domain Requirements

1. The system shall reach AWS Gateway and related AWS services to obtain any data when needed.
2. The system shall have access to a national database to confirm SSN and get customers' vaccine status.
3. The system shall have access to Google Firebase and Google Maps services to get users' locations so that nearby vaccine centers and customers' locations can be displayed in the map component.

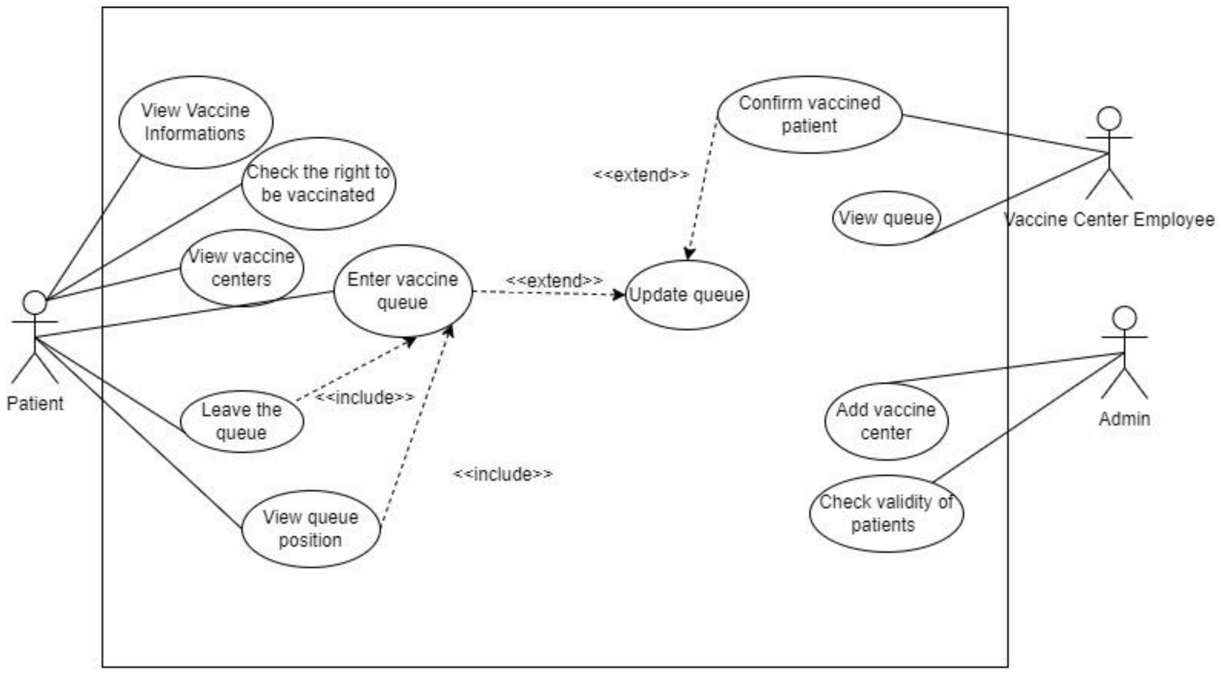
2.4 Assumptions and Justifications

1. We have assumed that we can confirm the SSN of the customers and customer vaccine statuses via the national database.
2. Since in the beginning, we didn't have any data in the system, we assumed that we could manually add data for displaying statistics, estimating waiting time, etc.

2.5 Structured Use Case Diagrams

The use case diagram below shows the actors interacting with the system. It shows in a simple way what actions the users are doing. Since the system is very simple in structure, the use case diagram is quite simple as shown.

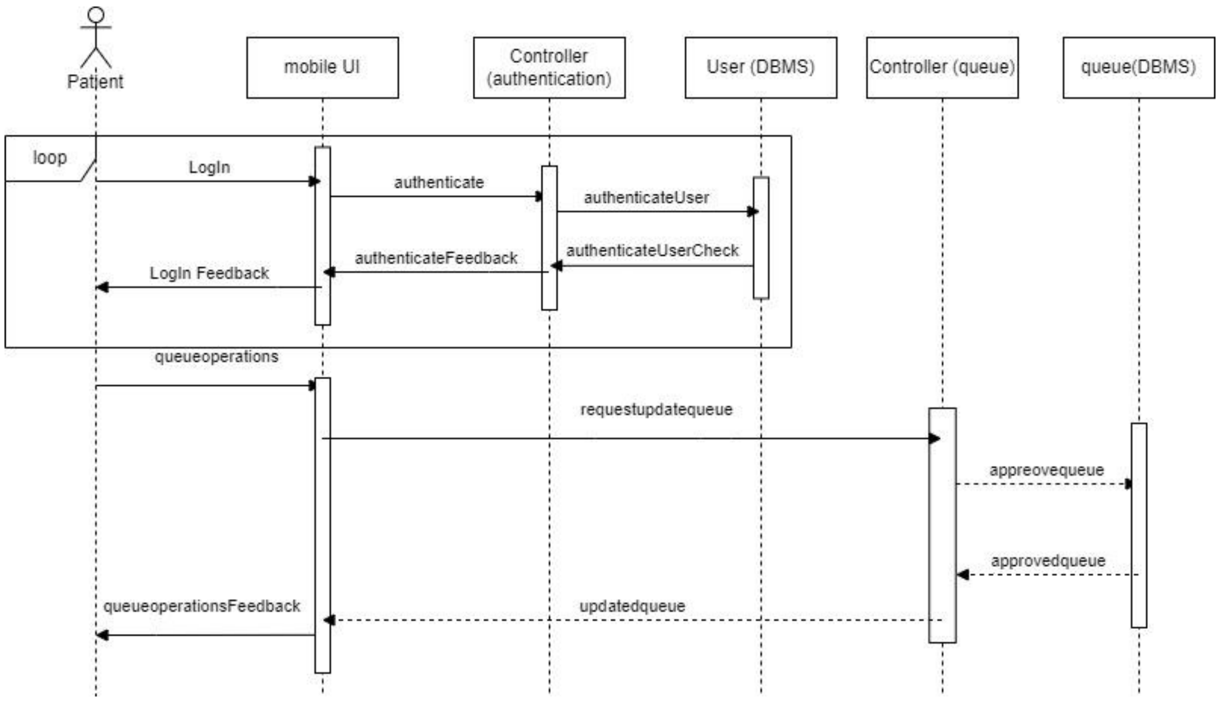
USE CASE	
ACTOR	Patient, Vaccine Center Employee,Admin
PRE-CONDITION	• Vaccine Centers and employees should be added to the system by Admin. • Patient must log in to the system in an approved manner
POST-CONDITION	• At the end of vaccination patient should be saved to the database as vaccined and then patient removed from the queue. • Vaccine center employee should approve if patient vaccined • The system sends a notification message to the patient
PATH	• Patient reviews vaccination status • The patient examines the vaccination centers where he will be vaccinated. • Patient queues to be vaccinated • The queue is updated • Patient can check his remaining time • If the patient gives up, he leaves the queue. • The queue is updated • Vaccination center employee verifies vaccinated patients.



2.6 High Level Sequence Diagrams

Sequence diagrams show the interactions between objects in the order in which these interactions occur. Sequence diagrams describe how and in what order the objects in a system work. These diagrams are widely used by software developers to understand the requirements of existing systems.

For the major use cases sequence diagram would be like this
The patient logs into the system via the mobile UI. The system accepts by authenticating the patient through the user system database. After the patient enters the system, he/she sends a request to queue himself/herself from the queue screen, if the data in the queue is suitable, the request is approved, and the patient is added to the queue.



2.7 Interface Definitions with External Systems

There are four main external systems that the app talks with:

- Google Firebase: Involves the server-side operations of our application. There are various APIs that the Firebase suite provides (all within Google Cloud Services) and we use several of them:

Firebase Cloud Messaging: Firebase Cloud Messaging handles the delivery of these alerts to the relevant devices for our client application to capture, whether we send them manually through the Firebase Dashboard or automatically using the Cloud Functions. We have the option of sending a generic notice to all users or sending alerts to specific persons based on their device, which we store for each user in the database.

Google Maps API: The map will provide with each hospital shown as pins will constitute a huge part of our main interface. Through our Google Cloud Account, we can create Google Maps API Keys, which we can then specify in the application to create the Map instance. For each active hospital, we will create a pin on the map right on the coordinates of that given hospital. The user can zoom in, zoom out through the map to find the most suitable hospital available for him/her.

Amazon Web Services:

- Elastic Cloud Service (ECS)
  Deployment and host purposes are provided by AWS EC2 service. This service provides a virtual machine to manage the whole project. The ubuntu server image is installed on it and managing the server from console.

- Elastic Bean Storage (EBS)
  It provides us a local storage system like for our virtual machine. Hence we use SSD based storage, it is fast and scalable for our project.

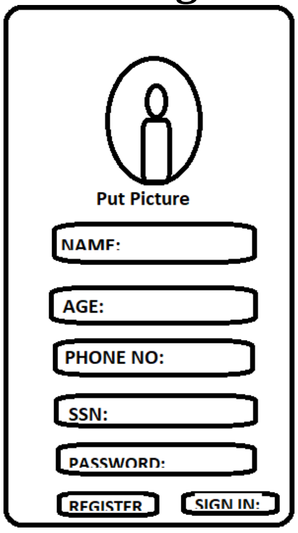
- Docker
  Dockerizing the microservices makes the the product environment isolated from host machine. Each microservice communicates with others via unique bridge network, so the product environment become more secure for attacks. Moreover, maintain and recovering the microservices are more powerful in this system.

- PostgreSQL
  - Each microservice has it’s own database and they are communicating via bridge if it’s needed. Therefore, we also isolated the database and secure the other microservices’ databases for any leak condition.

- LoadBalancer
  - Load balancer provides us a highly maintainable and scalable product environment if we have heavy workloads, traffic, etc. If the virtual machine will have a resource problem at any time, load balancer catch that traffic and maintain it from different machines. This is very critical for high traffic.

2.8 Graphical User Interface

2.8.1 Register Screen



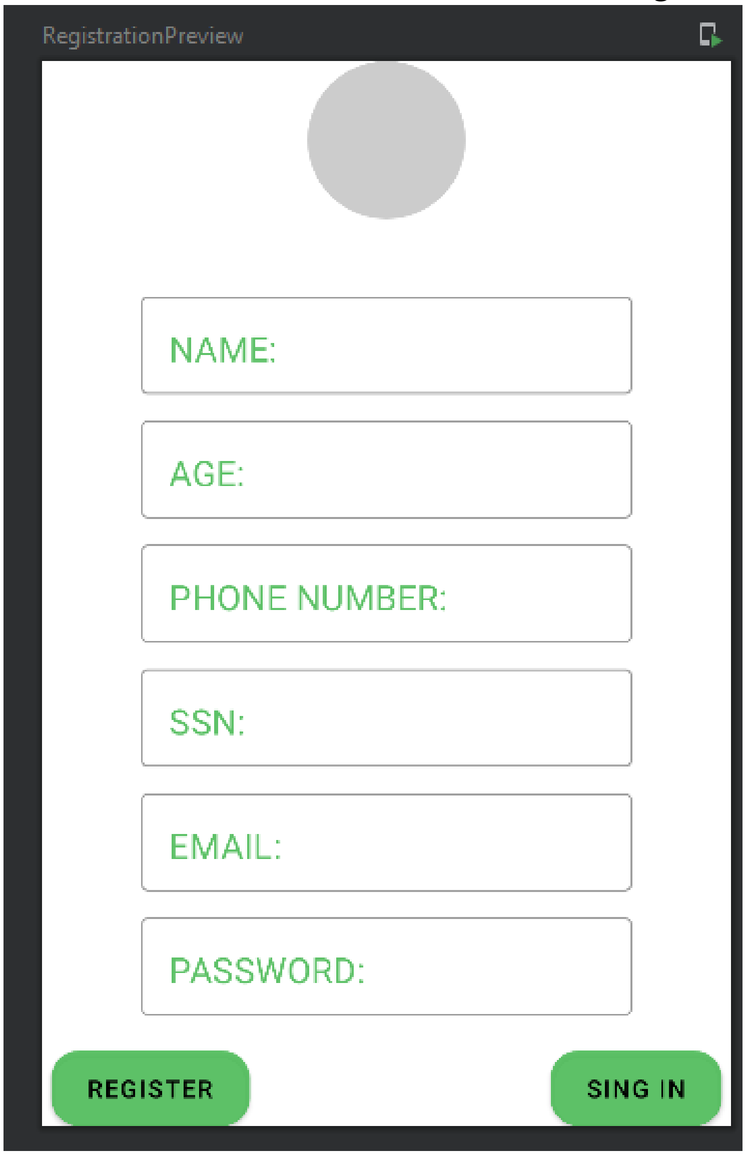
TEXT BOX:

The **NAME, AGE, PHONE NO, SSN, and PASSWORD** are all user input needed in the registration process.

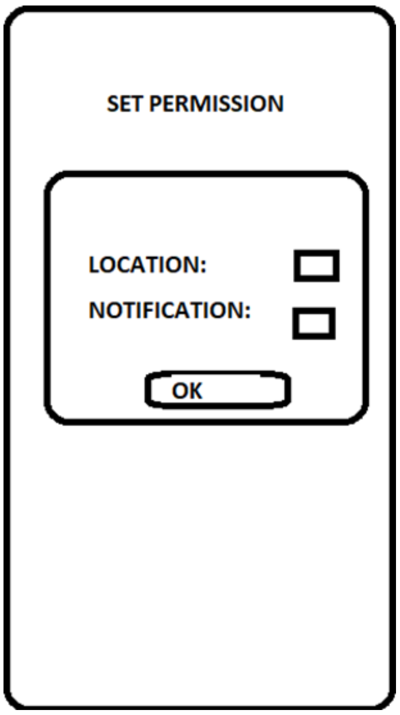
BUTTONS:

REGISTER-> when clicked, checks the accuracy of the user input, checks if user is already in the systems database, adds the user to the system if they are not in the database of the system, then go to the Permission Screen.

SIGNIN-> When click takes the use to the Login Screen.



2.8.2 Permission Screen

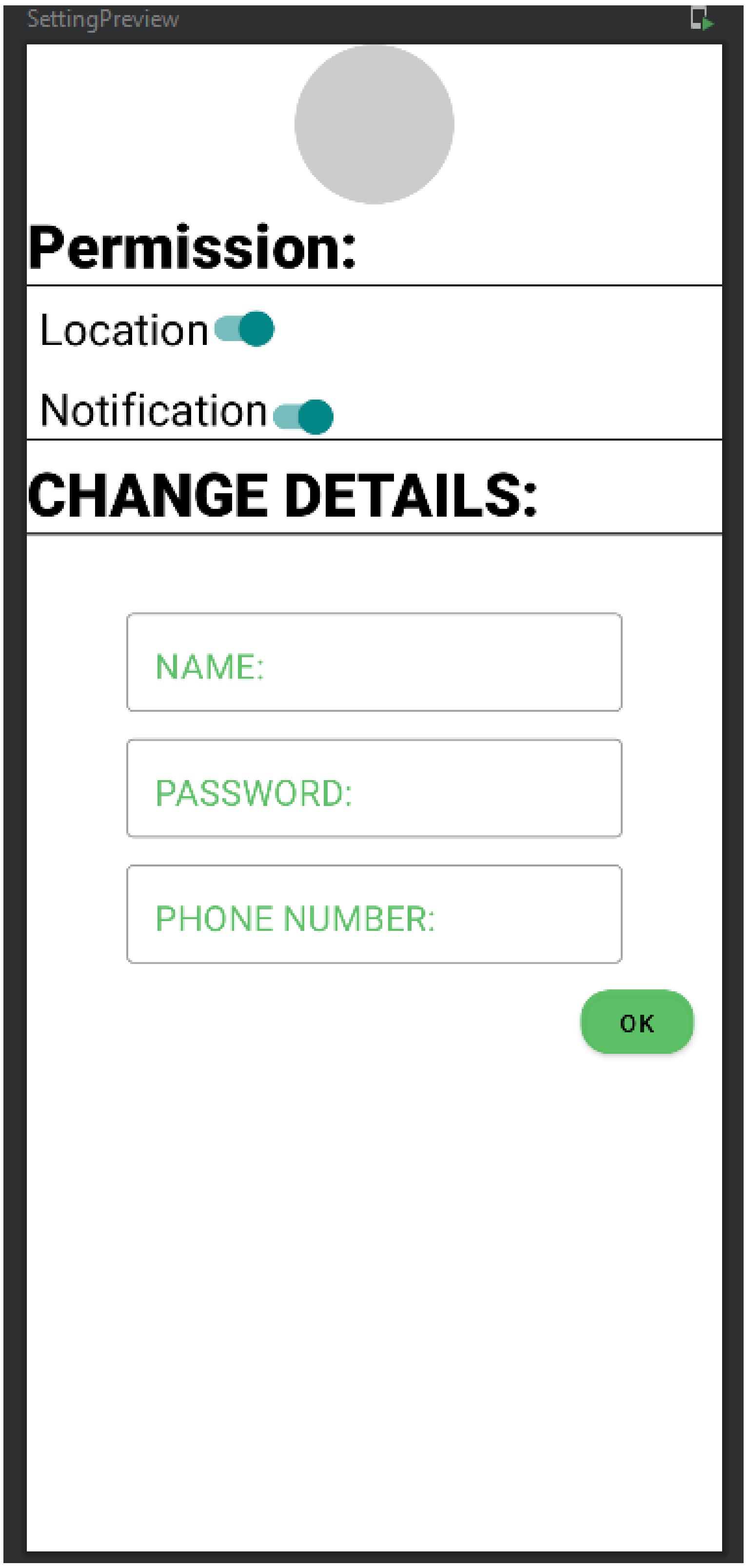


CHECK BOX:

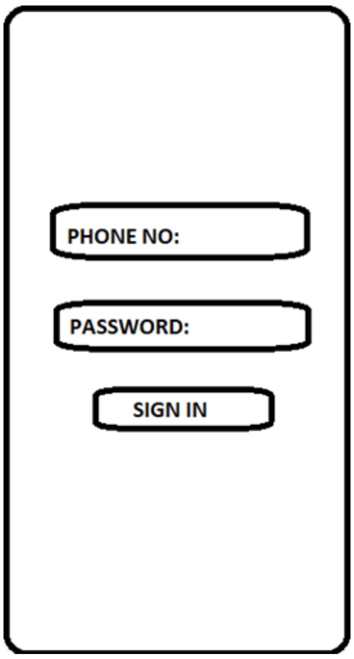
The **LOCATION** and **NOTIFICATION** check box if checked gives a true case for the above permission else a false, if false users would not be able to receive Notification or get proper information about the vaccination centers close to them.

BUTTON:

OK-> When clicked sends the result of the **LOCATION** and **NOTIFICATION** check box to the Android permission system for approval. Then goes to the home screen

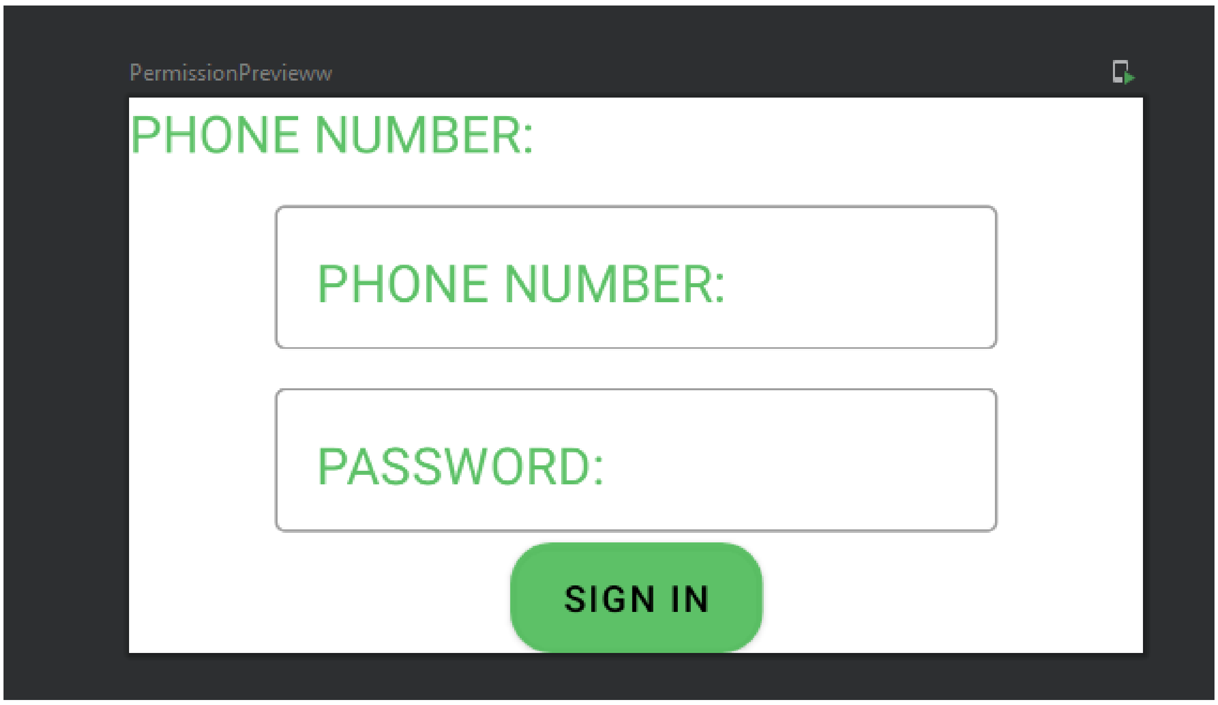

SettingPreview
Permission:
Location
Notification
CHANGE DETAILS:
NAME:
PASSWORD:
PHONE NUMBER:
OK

2.8.3 Login Screen

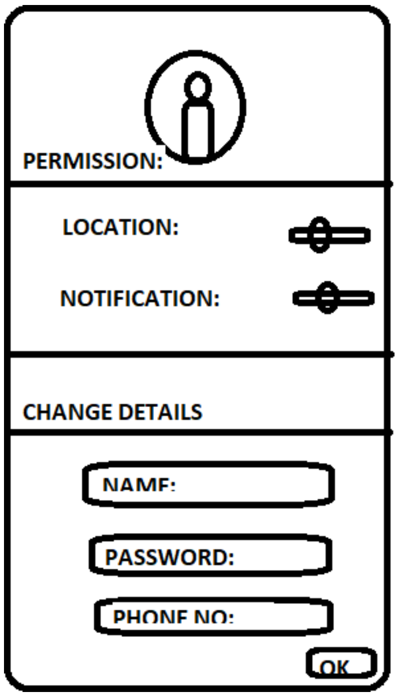


The **PHONE NO and PASSWORD** are all user input needed in the sign in process.

BUTTON:
SINGIN->: when clicked, checks the accuracy of the user input, if false show error message and remain in the sign in screen, else moves to the home screen.


PermissionPrevieww
PHONE NUMBER:
PHONE NUMBER:
PASSWORD:
SIGN IN

2.8.4 Settings Screen



TOGGLE BUTTON:

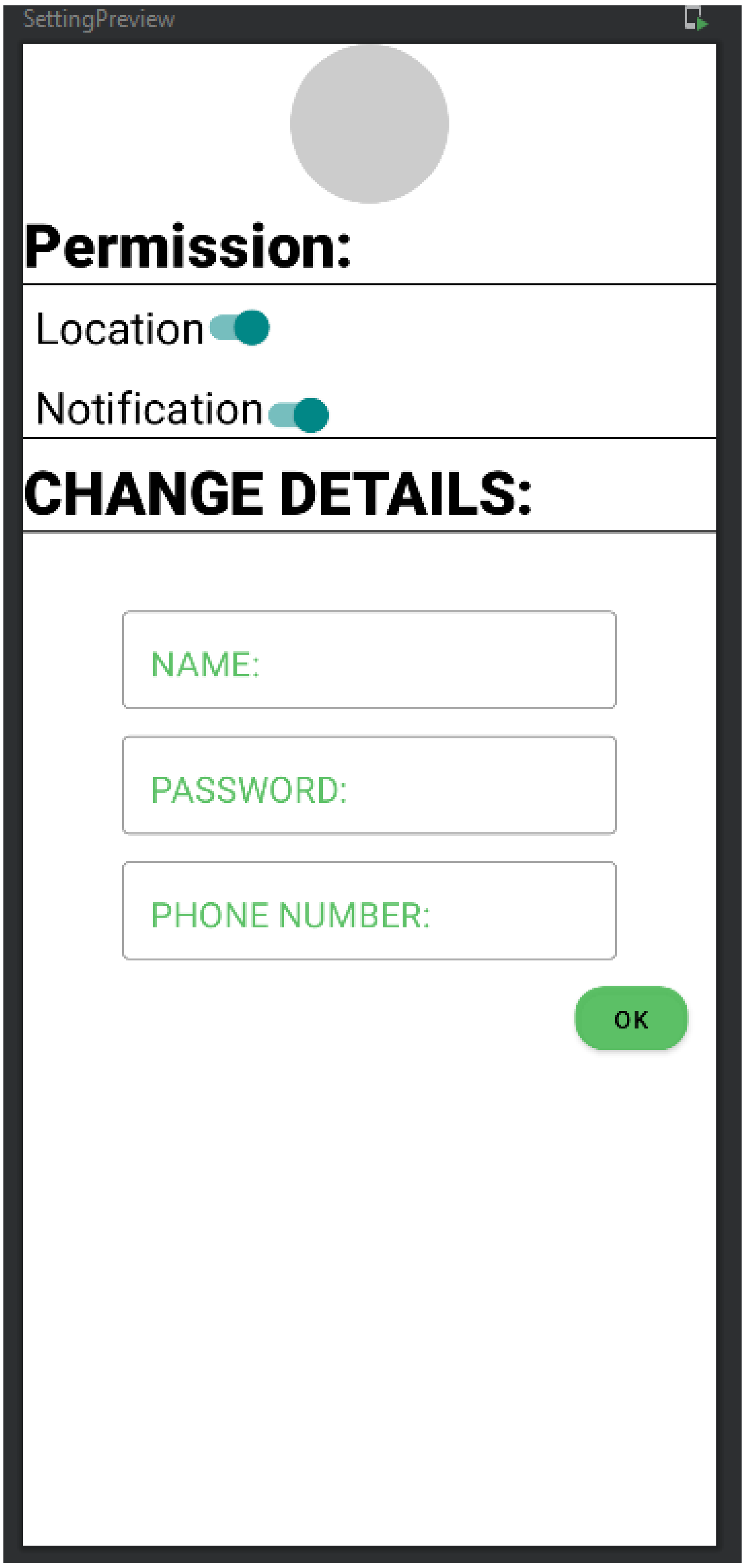
The **LOCATION** and **NOTIFICATION** toggle button is used to change the permission status (Lift for false and Right for true).

TEXT BOX:

The **NAME, PHONE NO,** and **PASSWORD** are all user input.

BUTTON:

OK-> When clicked updates the changes made by the use.



SettingPreview

Permission:

Location

Notification

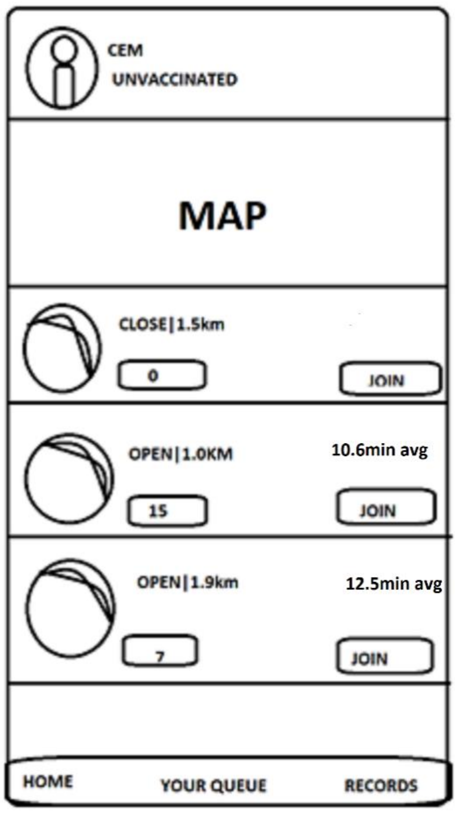
CHANGE DETAILS:

NAME:

PASSWORD:

PHONE NUMBER:

2.8.5 Home Screen



MAP:

Shows a map interface of all the hospitals close to you.

Button:

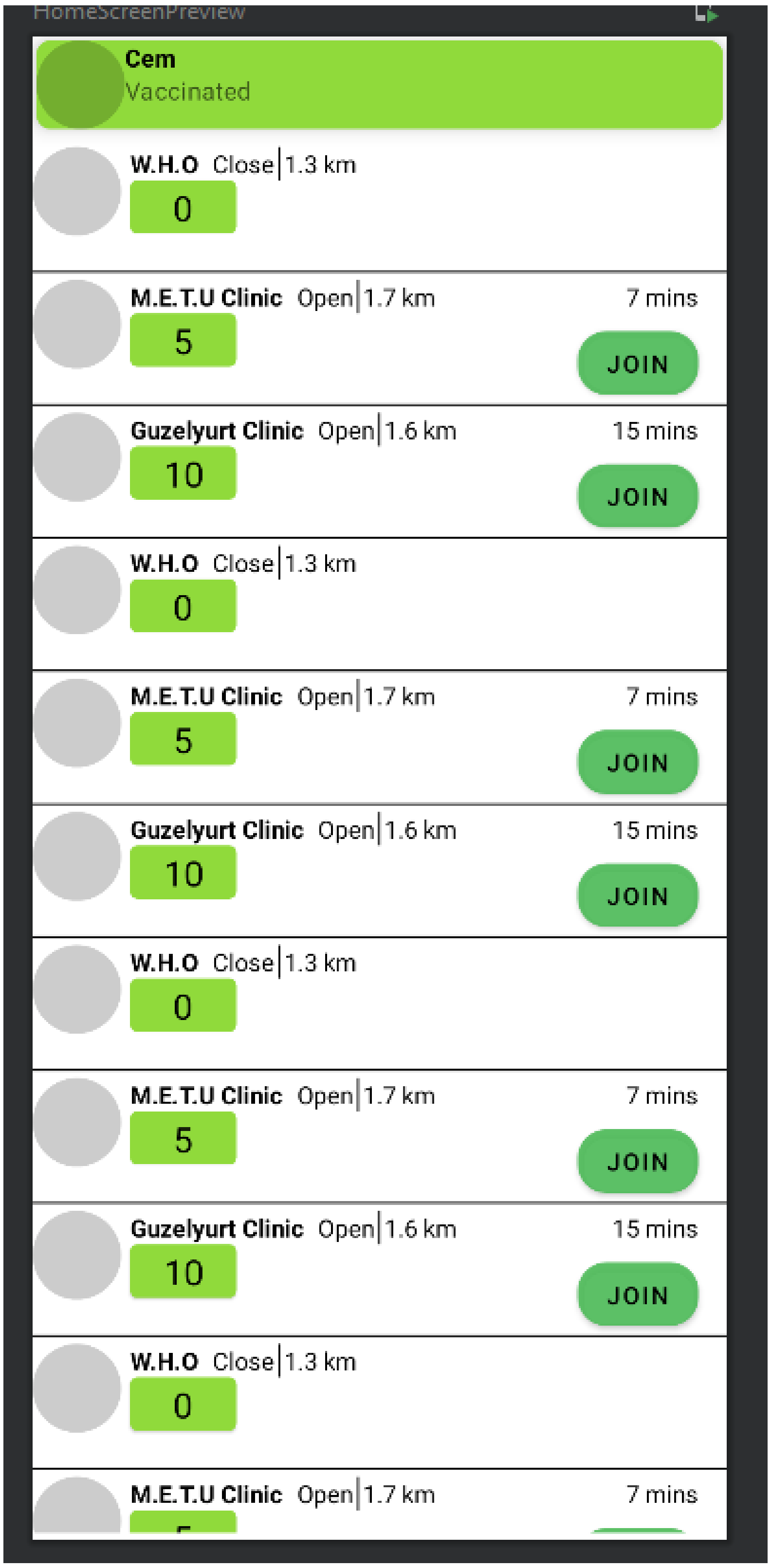
JOIN-> When clicked take the user to the Vaccination center Screen and adds the user to the queue.

Bottom TAPS

HOME-> When clicked takes you to the home screen.

YOUR QUEUE-> When clicked takes you to Vaccination center screen where you are in queue.

RECORDS->When click takes you to the Record Screen.


HomeScreenPreview

Cem
Vaccinated

W.H.O Close|1.3 km
0

M.E.T.U Clinic Open|1.7 km 7 mins
5
JOIN

Guzelyurt Clinic Open|1.6 km 15 mins
10
JOIN

W.H.O Close|1.3 km
0

M.E.T.U Clinic Open|1.7 km 7 mins
5
JOIN

Guzelyurt Clinic Open|1.6 km 15 mins
10
JOIN

W.H.O Close|1.3 km
0

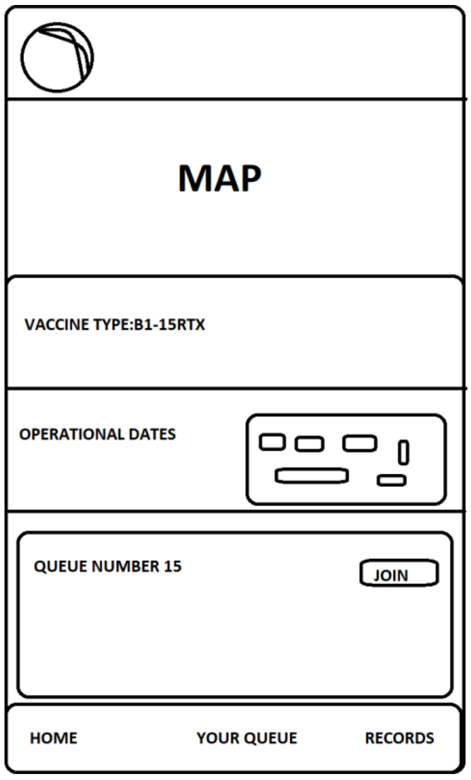
M.E.T.U Clinic Open|1.7 km 7 mins
5
JOIN

Guzelyurt Clinic Open|1.6 km 15 mins
10
JOIN

W.H.O Close|1.3 km
0

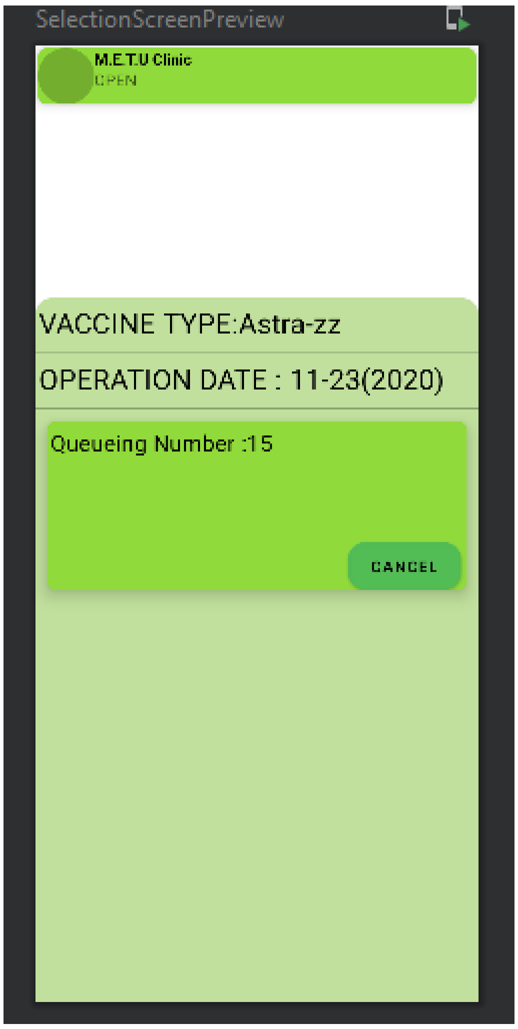
M.E.T.U Clinic Open|1.7 km 7 mins

2.8.6 Vaccine Center Screen



BUTTON:

JOIN-> When Click adds the user to the queue.

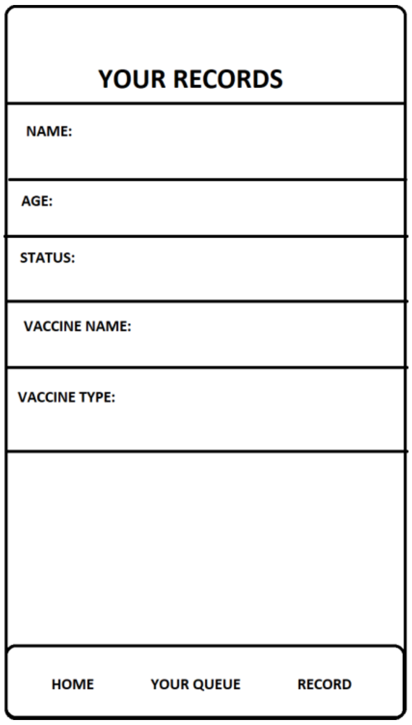


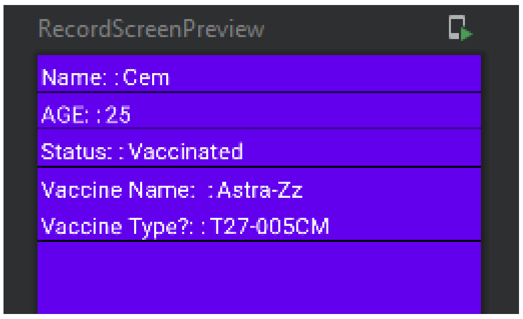
VACCINE TYPE:Astra-zz

OPERATION DATE : 11-23(2020)

Queueing Number :15

2.8.7 Record Screen




RecordScreenPreview

Name: : Cem

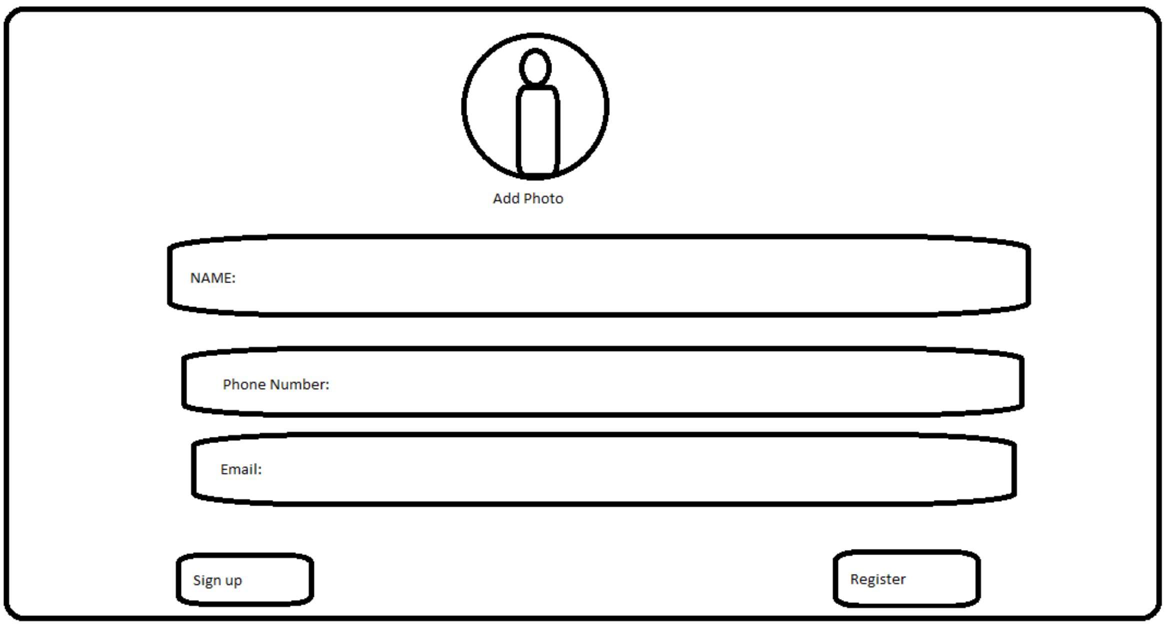
AGE: : 25

Status: : Vaccinated

Vaccine Name: : Astra-Zz

Vaccine Type?: : T27-005CM

2.8.9 Vaccine center login Screen



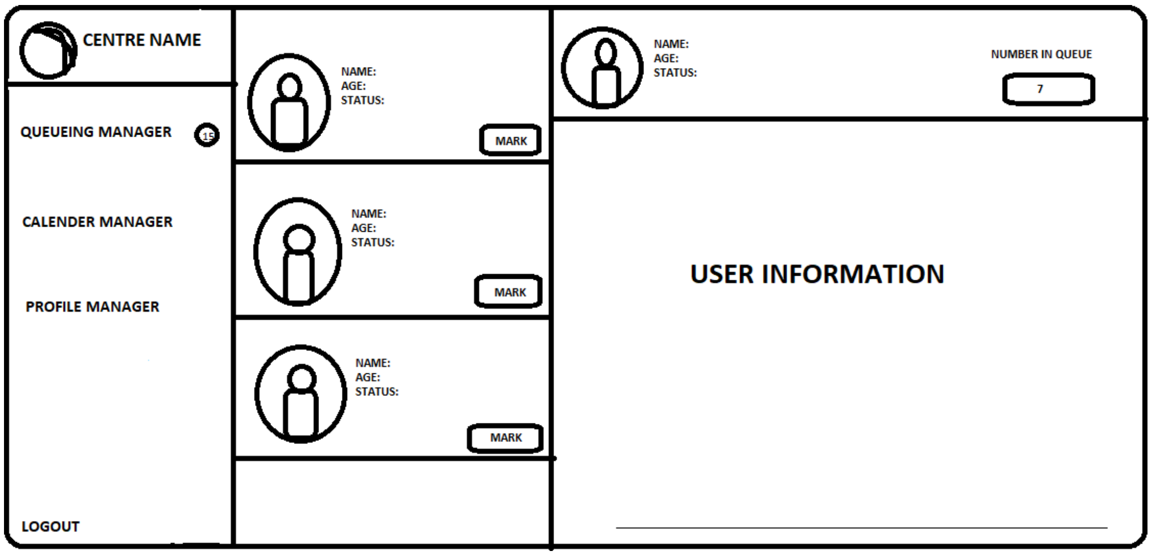
EDIT TEXT:

The **NAME, PHONE NUMBER, EMAIL** are all user input.

BUTTON:

REGISTER-> when clicked, checks the accuracy of the user input, checks if user is already in the systems database, adds the user to the system if they are not in the database of the system, then go to the home screen

2.8.10 Vaccine center Home Screen (after Queuing manager is clicked)



Clickable Text:

QUEING MANAGER-> Show the List of User in the Queue.

CALENDER MANAGER-> Show the Calendar of the Vaccination Center.

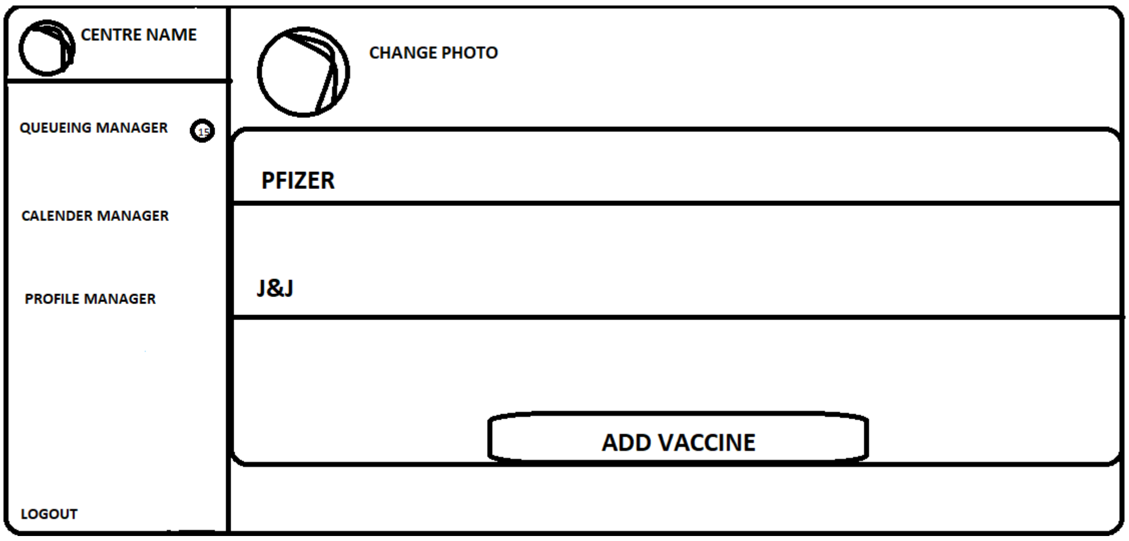
PROFILR MANAGER-> Shows the profile of the Vaccination center.

LOGOUT->Takes the user always from the application.

Button:

MARK-> When clicked takes the Vaccine customer out of the queue.

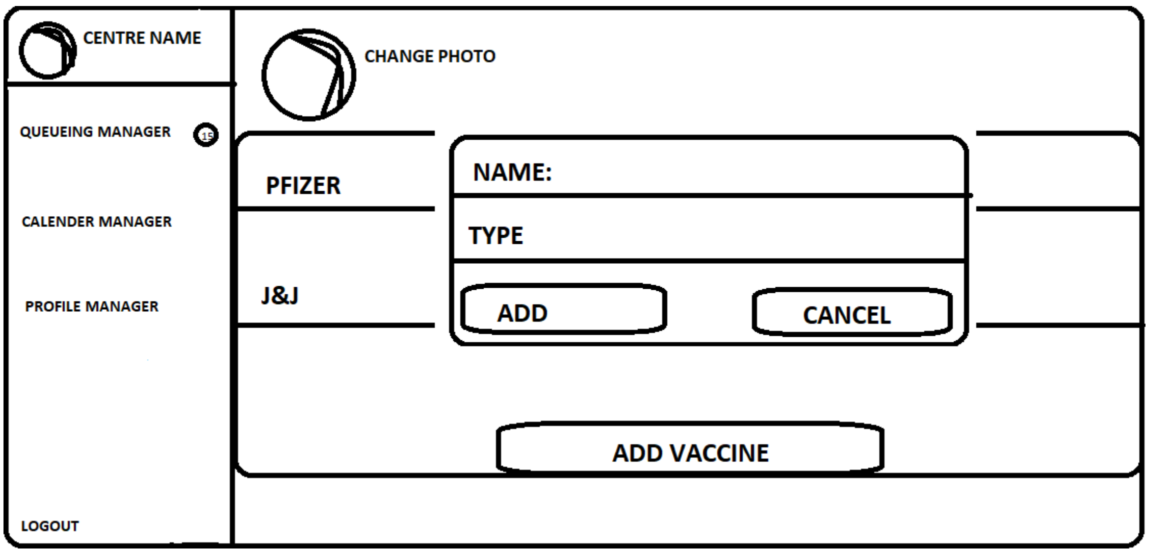
2.8.11 Vaccine center Home Screen **(after Profile manager is clicked)**



BUTTON:

ADD VACCINE-> When vaccine button is clicked a popup card is displayed for the user to add a new vaccine.

2.8.12 Vaccine center Home Screen (after Add Vaccine is clicked)



EDIT TEXT:

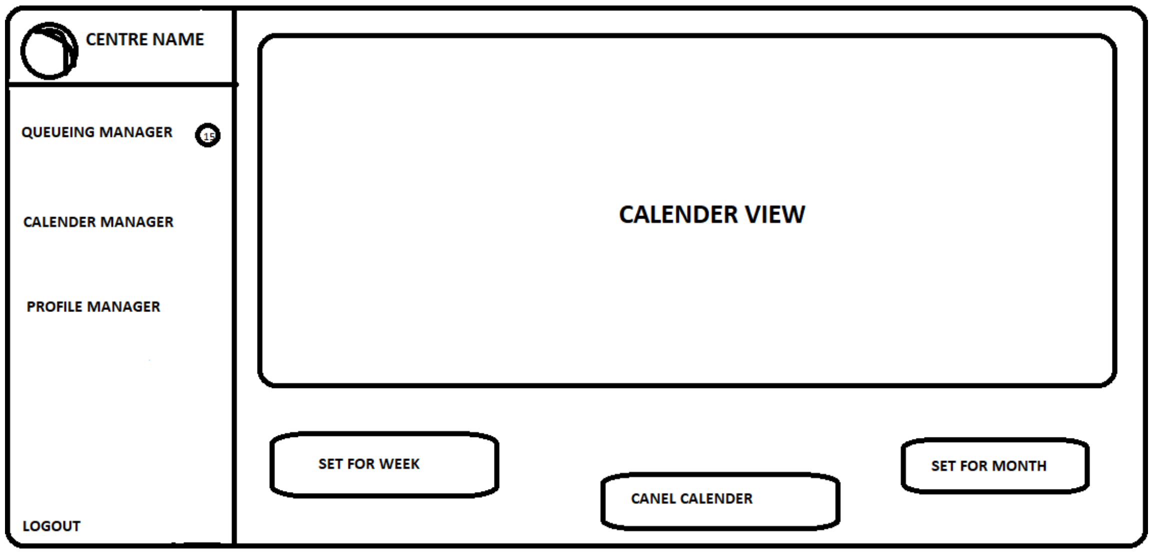
The **NAME, TYPE**, takes the user input.

Button:

ADD->When clicked it adds the Vaccine to the database.

CANCEL->When clicked it removes the selected vaccine from the user’s database.

2.8.13 Vaccine center Home Screen **(after calendar Manager is clicked)**



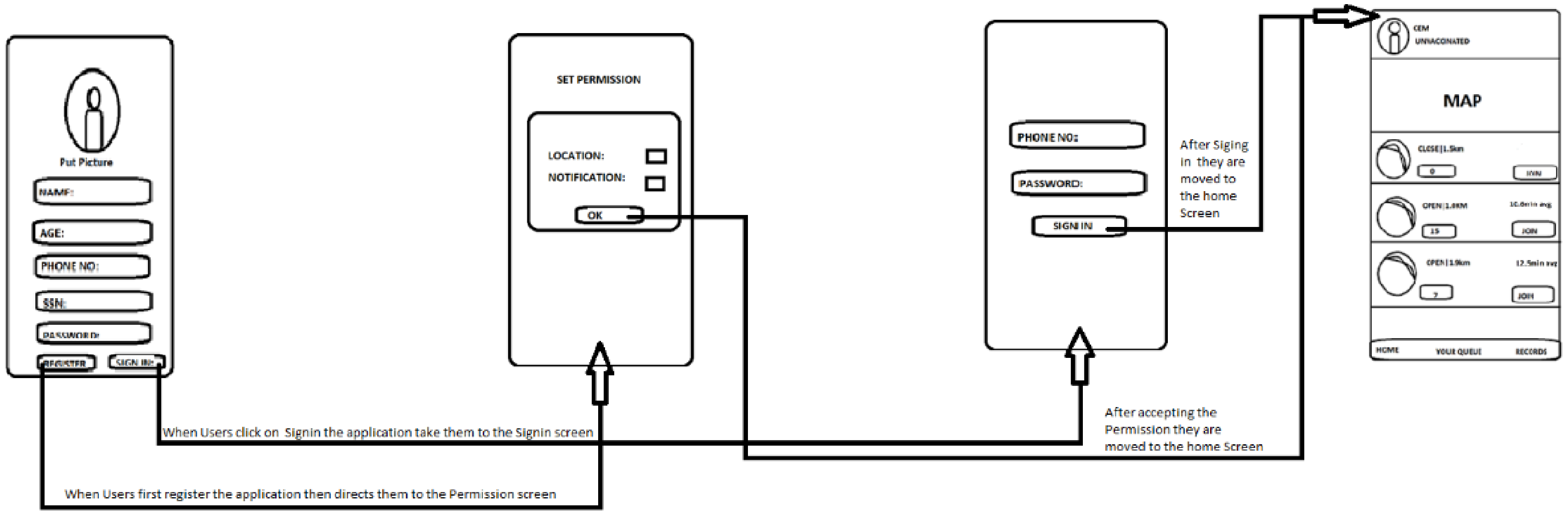
BUTTON:

SET FOR WEEK-> When clicked sets all the working days in a week in the calendar as open and sends it to the data base.

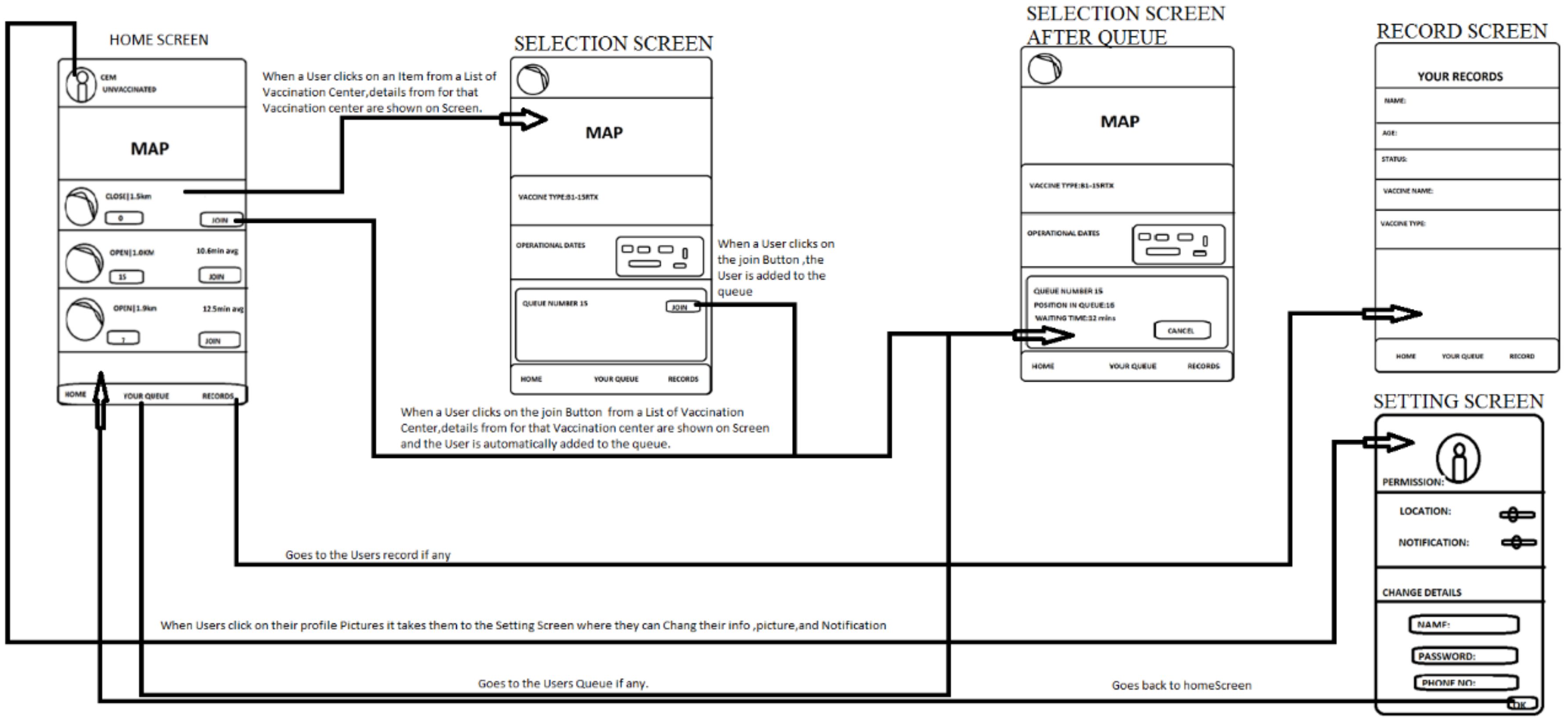
CANEL CALENDAR-> When clicked cancel all the open dates.

SET FOR MONTH-> When clicked sets all the working days in a Month in the calendar as open and sends it to the data base.

REGISTRATION PATHWAY


Put Picture
NAME:
AGE:
PHONE NO:
SSN:
PASSWORD:
REGISTER
SIGN IN
SET PERMISSION
LOCATION:
NOTIFICATION:
OK
PHONE NO:
PASSWORD:
SIGN IN
After Siging in they are moved to the home Screen
CEM
UNVACONATED
MAP
CLOSE|1.5km
0
JOIN
OPEN|1.8KM
10.0min avg
15
JOIN
OPEN|1.9km
12.5min avg
7
JOIN
HOME
YOUR QUEUE
RECORDS
When Users click on Signin the application take them to the Signin screen
After accepting the Permission they are moved to the home Screen
When Users first register the application then directs them to the Permission screen

USER NAVIGATION PATHWAY



When a User clicks on an Item from a List of Vaccination Center,details from for that Vaccination center are shown on Screen.

When a User clicks on the join Button ,the User is added to the queue

When a User clicks on the join Button from a List of Vaccination Center,details from for that Vaccination center are shown on Screen and the User is automatically added to the queue.

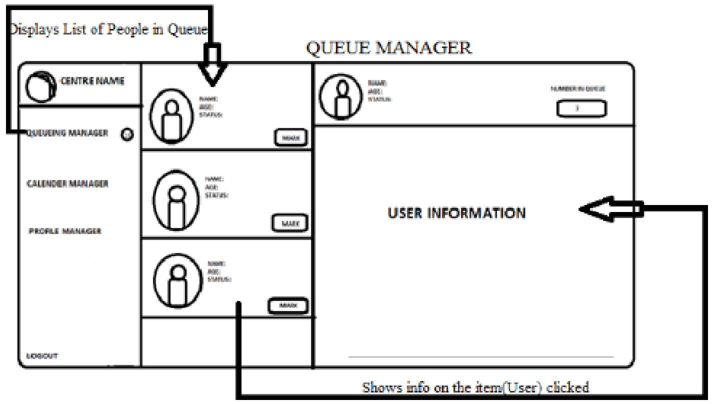
Goes to the Users record if any

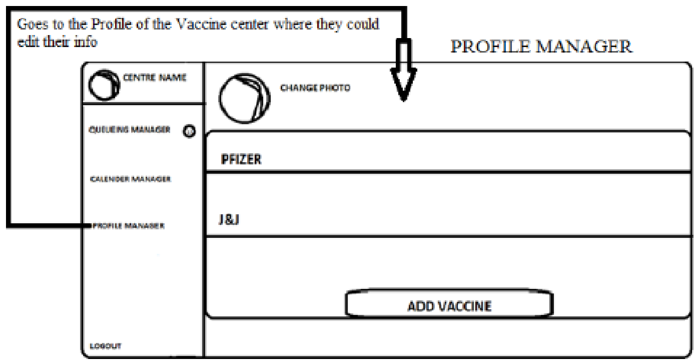
When Users click on their profile Pictures it takes them to the Setting Screen where they can Chang their info ,picture,and Notification

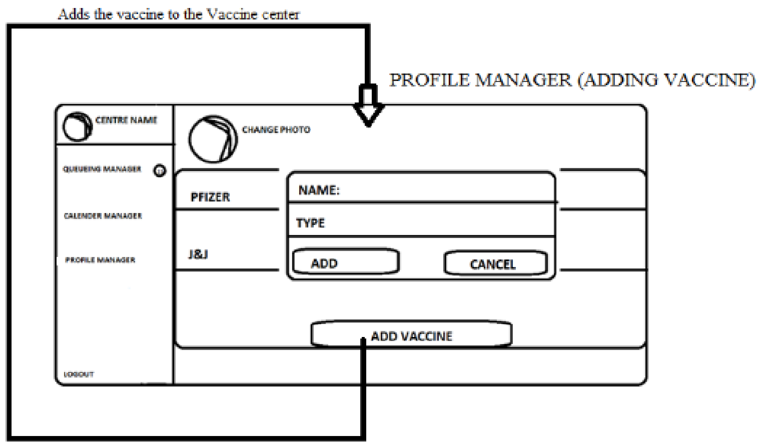
Goes to the Users Queue if any.

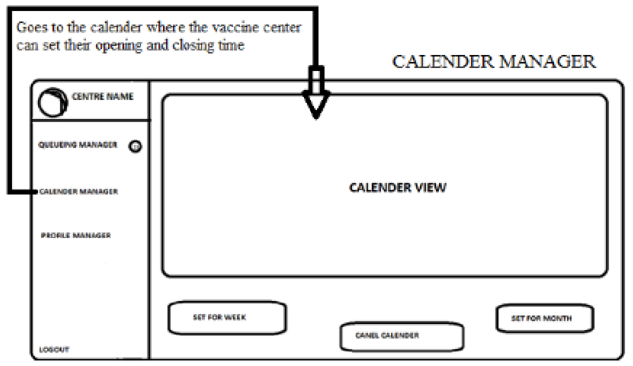
Goes back to homeScreen

VACCINE CENTER PATHWAY





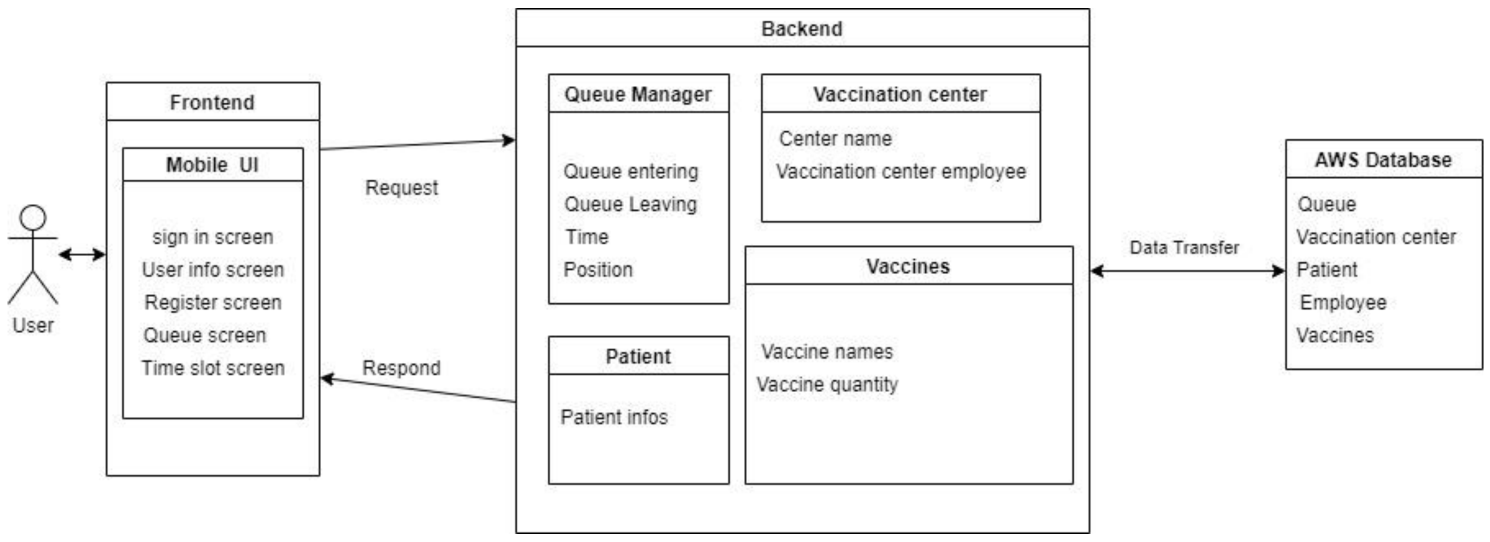




Chapter 3
System Architecture and Models

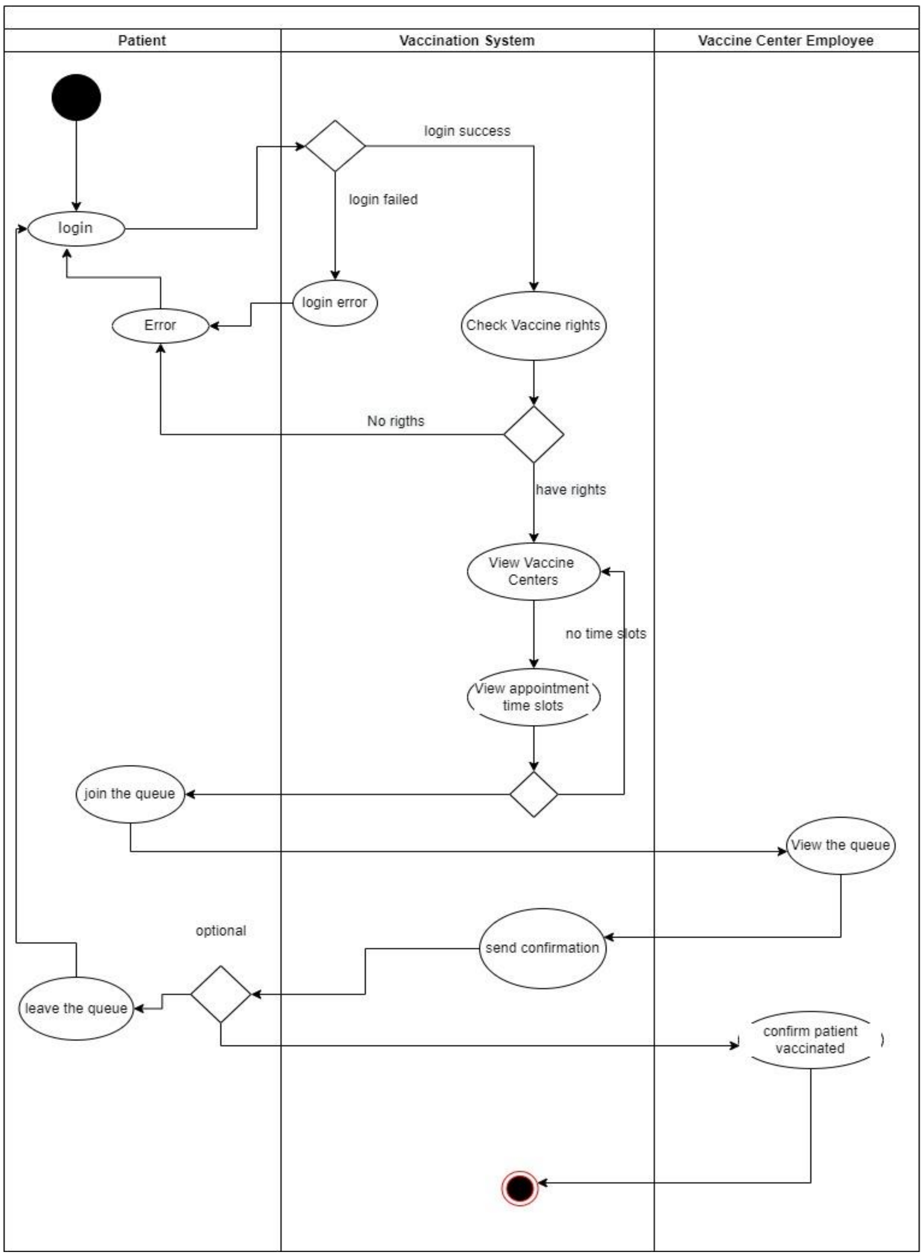
3.1 Architectural Model

Figure shows the architectural model of our application. There are 3 components in our architectural model: the Frontend with which the user interacts, the backend where the actual processes take place in the background include the queue system, and the AWS database where all data is stored. It should be noted that vaccine center employees are natural persons working in the centers and are also users of the application and have separate GUIs. Vaccination center employee are also obligated to confirm whether patients have been vaccinated. Aws Database is required to store all the events and information happening in the background.



3.2. Process Model

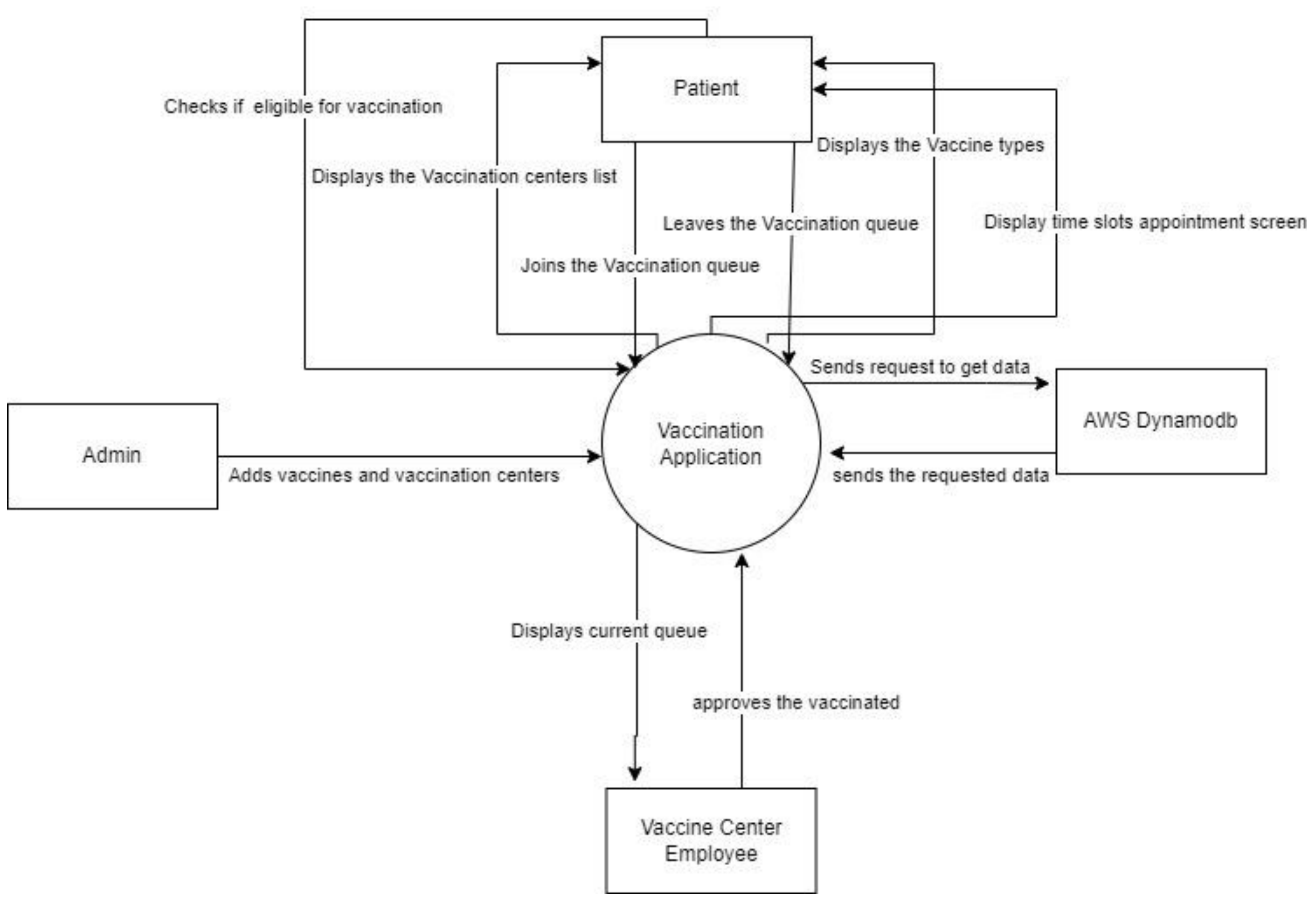
- The patient logs into the system, if the system does not confirm the patient's identity, the patient returns to the login screen.
- The patient checks whether he can be vaccinated. If he does not have the right to be vaccinated, he will be directed to the login screen.
- The patient chooses the vaccination center to be vaccinated.
- The patient chooses the appropriate time to be vaccinated. If all times are full, the patient returns to the back page to select another vaccination centre.
- Patient queues to be vaccinated
- The vaccination center employee sees the people to be vaccinated.
- The system sends a confirmation message notification to the patient.
- If the patient gives up, he leaves the queue and returns to the login screen.
- After the patient has been vaccinated, the vaccination center employee confirms that the patient has been vaccinated.



3.3 Data Flow Models

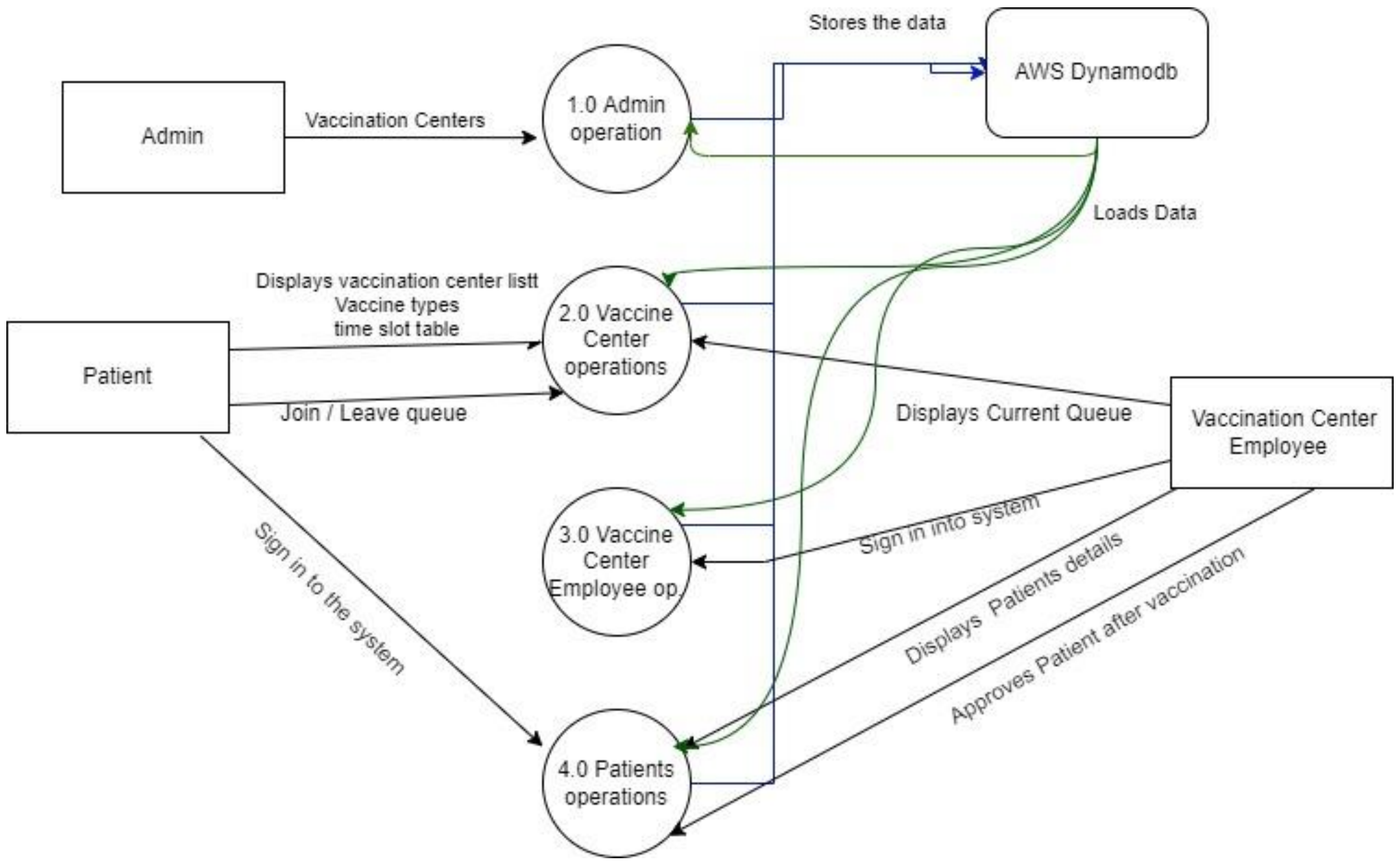
3.3.1 Data Flow Diagram Level 0

- Patient checks if they can be vaccinated
- The system shows the vaccine centers where the patient will be vaccinated.
- The system shows the patient the timeline
- The system shows the patient the vaccines they can get
- Patient queues to be vaccinated
- If the patient gives up, he leaves the queue.
- Admin adds vaccination centers and vaccines to the system
- The system requests the requested data from the database.
- The database sends the requested data to the system.
- The system shows the queue to the vaccine center employee.
- Vaccination center employee approves people who are vaccinated.
- The system also sends a notification message to the patient


Patient
Checks if eligible for vaccination
Displays the Vaccination centers list
Displays the Vaccine types
Leaves the Vaccination queue
Display time slots appointment screen
Joins the Vaccination queue
Sends request to get data
AWS Dynamodb
Admin
Vaccination Application
Adds vaccines and vaccination centers
sends the requested data
Displays current queue
approves the vaccinated
Vaccine Center Employee

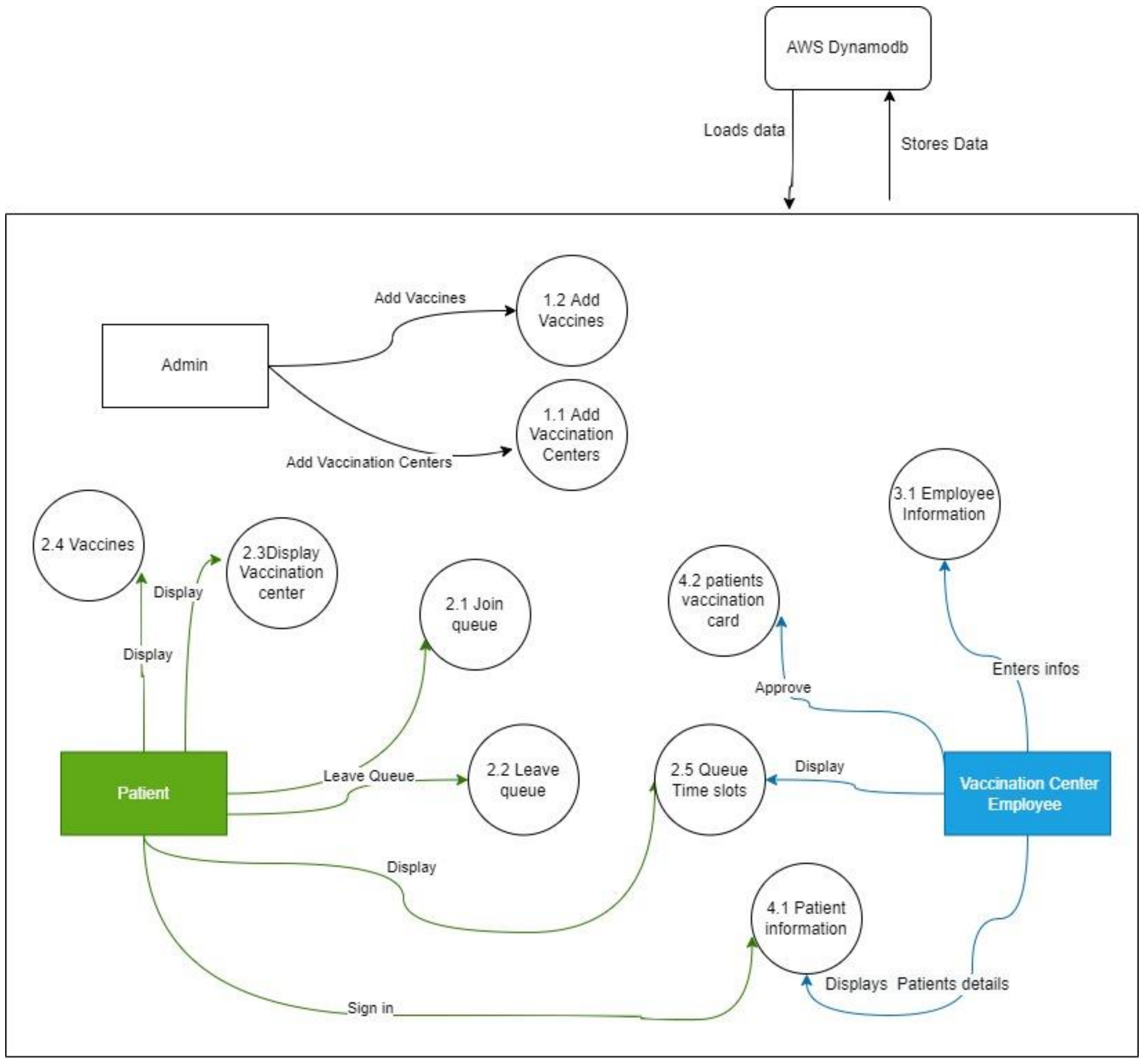
3.3.2 Data Flow Diagram Level 1

In addition to data flow diagram 0, vaccination application in diagram 1 is divided into 4 main extended parts. These are admin operation, vaccine center operation, vaccine center employee operation and patients operation. The vaccine center operation includes the queue system, so the patient interacts with it after registering with the system. After the vaccine center employee logs into the system, he/she interacts with the patient to verify that the patient has been vaccinated, and also with the vaccine center to check the queue system.



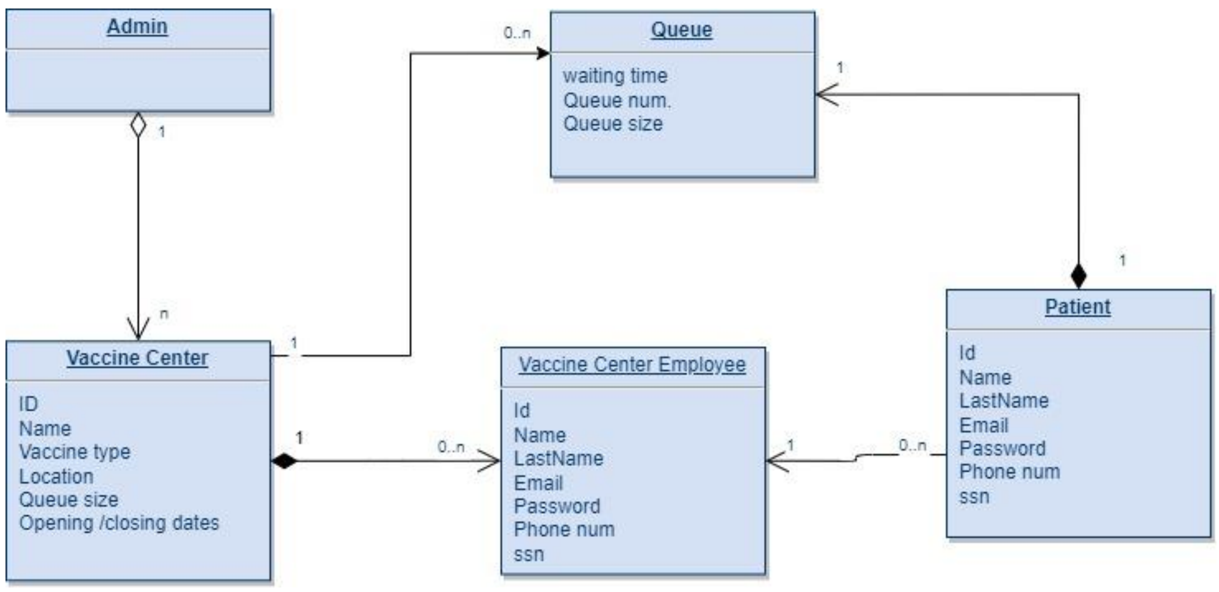
3.3.3 Data Flow Diagram Level 2

Data flow diagram 2 is a more detailed diagram than Data flow diagram 1, and the queue system is shown in more detail.



3.5 Database Design

A simple structure of our AWS Database application is as follows. There are vaccination centers, Patients, and Vaccination center employess. Each of them has their own information. Admin can add n Vaccine Centers, if the admin leaves the application, nothing added will be deleted. Vaccine Centers has n vaccine center employees. If the vaccine center is deleted, the vaccine center employee is also deleted. vaccine center employee is responsible for n patients. It gives approval for vaccination after vaccination. 1 patient can join 1 queue. if the patient is deleted, patient will be removed where he joined in the queue.



Chapter 4
Project Estimation

4.1.1 Function Point Estimation

Inputs

1. Register to the system — Medium Complexity
2. Log in to the system — Low Complexity
3. Change user details setting — Low Complexity
4. Add user vaccination details — Low complexity
5. Employer calendar management — Low complexity
6. List the Vaccine Centers (by taking User location as input) — Medium Complexity
7. Selecting the Action while entering Queue — Low Complexity
8. Emergency queue cancelation — Medium Complexity
9. Emergency entry for the queue — High Complexity

In Total : 5 x Low Complexity , 3 x Medium Complexity ,1 x High Complexity

Outputs

1. Show user records — Low complexity
2. Displaying a Map with the user’s and Vaccine Centers’ locations — High Complexity
3. List of Vaccine Centers with queue status, distance to the user, SSN, phone number, address, etc. — Medium Complexity
4. Progress of Queue will be displayed Dynamically — High Complexity
5. Notification system for queue alerts and reminders — Medium Complexity

In Total: 1 x Low Complexity, 2 x Medium Complexity, 2 x High Complexity

Inquiries

1. Registration process — High Complexity
2. Listing the Vaccine Centers — Medium Complexity
3. Calculating the distance between user and vaccine center — Low Complexity
4. Entering t queue — High Complexity
5. Leave the column — Medium Complexity
6. Show details of vaccination to the user — Low Complexity
7. Emergency queue entry — Medium Complexity
8. Listen to live queue entries and cancellations — High Complexity
9. Show Vaccine Center Statistics for employees — Low Complexity
10. Log and follow user activity process — High Complexity

In Total : 3 x Low Complexity , 3 x Medium Complexity ,4 x High Complexity

External Interface Files

1. Users Collection — Medium Complexity
2. Vaccine Center Collection — Medium Complexity
3. Queues Collection — High Complexity
4. Redis instance Cache — High Complexity

In Total : 2 x Medium Complexity , 2 x High Complexity

4.1.2 Compute Unadjusted Total of Function Points

Inputs: 5 * 2(Low) + 3 * 4(Medium) + 1 * 6(High) = 28

Outputs: 1 * 4(Low) + 2 * 5(Medium) + 2 * 7(High) = 28

Inquiries: 3 * 3(Low) + 3 * 4(Medium) + 4 * 6(High) = 45

Logical Internal Files: None

External Interface Files: 2 * 7(Medium) + 2 * 10(High) = 34

Total UTFP: 28 + 28 + 45 + 34 = 135

4.1.3 General System Characteristics (GSC) Degree of Influence (DI)

	GSC	DI
1	Data Communications	5
2	Distributed Processing	1
3	Performance	3
4	Heavily used configuration	2
5	Transaction Rates	5
6	On-line data entry	5
7	Design for end-user efficiency	4
8	Online updates	4
9	Complex processing	1
10	Usable in other applications	2
11	Installation ease	1
12	Operational ease	1
13	Multiple sites	0
14	Facilitate change	4

Total of Degree of Influence: 38 $\rightarrow$ VAF = TDI $*$ 0.01 + 0.65 $\rightarrow$ VAF = 38 $*$ 0.01 + 0.65 = 1.03

Adjusted Total of Function Points (ATFP): ATFP = UTFP * VAF $\rightarrow$ 135 * 1.03 = 139.05

LOC SIZE METRIC

The languages that will be used with their percentage are as follow:

Programming Language	**Percentage**	**Language Unit size (Mode)**
Kotlin	%60	53
GO	%30	97
SQL	%10	21

Since Kotlin and GO was not on the list we used JAVA and C

(53*0.6) + (97*0.3) + (21*0.1) = 63

LOC = ATFP * Language Unit Size → 139.05 ∗ 63 = 8760.15

The values of Language Unit size (Mode) were taken from:

https://www.qsm.com/resources/function-point-languages-table

https://www.cs.helsinki.fi/u/taina/ohtu/fp.html

Constructive Cost Model (COCOMO)

For estimating the software development effort, we will use the estimated line of code we computed previously. To do that, we will use the basic Constructive Cost Model with the Semi-detached Development mode.

a= 2.4 b= 1.05 c= 0.38

KDSI = ATFP * Language Unit Size/ 1000 → 8760.15/1000 = 8.76

$MM = 2.4 * KDSI^{1.05}$ (for Organic) = 23.4

$TDEV = 2.5 * MM^{0.38}$ = **8.288**

For team of 4;

MM=23.4/4 = 5.85

$TDEV = 2.5 * MM^{0.38}$ = **5 months**

4.1.4 Rough Schedule Estimate

Rough Schedule Estimate = $(ATFP)^{\text{Class Exponent}}$;

Best Case: 6.85

Average Case: 7.94

Worst Case: 9.2

4.2 Milestones

- User Management
- Queue Management
- Database
- Interface Management

4.3 Tasks

1. M1 User Management

    - T1 Registration and Login
    - T2 Customized Screens

2. M2 Queue Management

    - T3 Join queue
    - T4 Leave queue
    - T5 Remove patient from queue

3. M3 Database

   - T6 Database Design
   - T7 Database Creation in AWS
   - T8 Integration of the database with the application

4. M4 Interface Management

   - T9 List of hospitals
   - T10 Search for nearby hospitals
   - T11 See the state of queue
   - T12 Record Screen preview

4.4 Task Allocation of First Semester

Task ID	Task Description	Estimated Time	Team Members
T1	Registration and Login	5 days	Antony
T2	Customized Screens	21 days	Antony,Burak,Ege,Güray
T3	Join queue	7 days	Burak, Ege
T4	Leave queue	7 days	Burak, Ege
T5	Remove patient from queue	7 days	Antony
T6	Database Design	7 days	Güray
T7	Database Creation in AWS	10 days	Güray
T8	Integration of the database with the application	14 days	Güray, Antony
T9	List of hospitals	7 days	Antony
T10	Search for nearby hospitals	14 days	Güray, Antony,Ege,Burak
T11	See the state of queue	7 days	Ege
T12	Record Screen preview	7 days	Burak

4.5 Activity Diagram of Tasks and Milestones

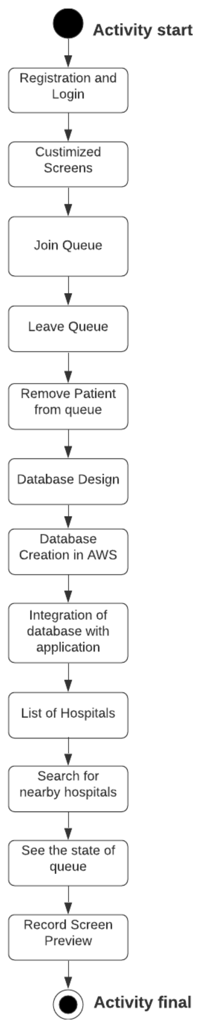


Figure shows the activity diagram which shows the milestones and the tasks of our main work this semester.

Chapter 5 Conclusion

5.1 Critical Evaluation

To properly evaluate the work done so far for the first semester, there are certain key point that would be check example:

- **Was the project planned in the right way?**

During the duration of this semester, we had weekly meetings with our supervisor in which she monitored our progress on the project and gave us useful feed-back. The team also had occasional meets, in which we want over our various part of the project, adjusted, and set deadline for ourselves based on the progress we had made so far. We had concluded that for the first semester we are going to focus on the Vaccination-Customer side of the application which involves a mobile application to connect to the vaccination centers this we have achieved, therefore we can say that our plan was done in the right way.

- **Did we reach all the timing (Deadline) of the project?**

We were a little bit behind our plan based on the deadline we set, this was mainly caused by the workload of other courses we had and finding the time to meet to discuss how to properly integrate the various application functionality subdivided to each team member. We greatly underestimated the workload needed to implement our first planned application functionality. However, after most of the midterms and lab report from our other courses were done, we put full focus on the project, and we were able to make up for lost time and complete the Vaccination Customer application functionality.

- **Does the final application Support the core functionality of the project?**

The final application supports all the core functionality based on the vaccination Customer application part, this includes ability to register, signin see available vaccination centers join a queue etc. We were able to complete the Mobile application design, and backend Implementation, despite the fact that we were new Android development and had to learn while trying to implement and develop the application.

Overall, we as team are proud of the work done so far as this will give us a good starting point when we start implementing the Queuing System.

5.2 Retrospective of First Semester

What went well?

The plan heading into the first semester was to finish the Vaccine Customer application side of our project, this we have achieved. Looking back, we also made the right decision to develop our android application based on android Native development, using android Studio as the development environment. We also made the right decision to use Kotlin programming language which was needed to use the new android declarative UI tool kit (Jetpack compose) which gave us more flexibility in our UI design when compared to XML design toolkit some android applications use. Finally, we had regular meetings with the Supervisor which was very helpful for us in improving our application and keeping us in the right direction.

What went less well?

Due to how relatively new Jetpack compose (UI Toolkit for Declarative UI Design Rendering) is most of the time when we had error there were not enough answer or reference to fix the bug online or in the android jetpack compose documentation page. This in some cases slowed us down, also we are relatively new to developing mobile application, so we had to learn how to develop and use the android studio while developing a Complex application at the same. Overall, we were able to overcome these challenges and catch up with time to complete our Customer Vaccination mobile application.

What should we do?

This first semester was more or less a crash course on android mobile development, but it taught us a lot. Having been through this learning process we would now be better effective in our planning and timing over the period of next semester. Also seeing that we be on a holiday over before the next semester begins, we plan to fully get familiar with android studio and complete the Vaccine Center GUI.

5.3 Future Work

Vaccination Center GUI: This would be starter and completed during the holidays, our plan is to focus on the GUI of the Vaccination Center and connect it to the backend Data base over the duration of the holidays. We would also make sure that both the Vaccination Center GUI and the Customer Vaccine Center GUI and all their system functionality work accordingly to the intended design.

General GUI Improvement: After both applications has been implemented, we would focus on making the design more appealing for our Users. During this period, we would polish the various UI component from the Text to the Buttons, cards, and other various Layout and themes we used in our design. We hope to complete this process during the holiday that is between this semester and next.

Queue System Integration: The queue System would start at the starting of next semester and occasional feedback would be given to us on our design, algorithm, and implementation by our supervisor.

Testing and Debugging: During the process of the queue System Integration occasional system testing and debugging would be carried out both for the queuing system and mobile application.

System application Simulation: The simulation part would be done after all the above-mentioned tasks are completed. The simulation is need for us to effectively check how our application would run in the real world in the hands of real user, it also checks how much work load can our application handle and how the simulation would manage millions of entry to it, for now further discussion would be needed to fully test all the scenarios that may occur during the lifetime of our application ,this would be discussed in future meeting