

VIRUAL QUEUE SYSTEM

Submitted in partial fulfilment of the requirements for the award of degree of
BACHELOR OF ENGINEERING IN COMPUTER SCIENCE & ENGINEERING



Submitted to:

MR.TARUN KUMAR

Submitted By:

RATNMAM KUMAR YADAV

19SCSE1010327

PRAKHAR DEV

19SCSE1010316

School of Computing Science and Engineering
Greater Noida, Uttar Pradesh

TABLE OF CONTENTS

S.No.	CONTENT	PAGE NO.
1.	Abstract	3
2.	Introduction	4
3.	Literature Survey	5
4.	Project Design	9
5.	Modules description	17
6.	Result	21
7.	Conclusion	24
8.	References	24

ABSTRACT:

The covid -19 pandemic has been a traumatic and painful experience for the whole world. Now , the world after struggling with this virus for nearly two years ,the hardworking scientists have brought the people with a sigh of relief by the covid -19 vaccines. Since, everyone wants to get vaccinated as soon as possible, vaccination centres get overcrowded. This can further lead to another harmful wave of the covid-19 virus as it is highly contagious. So overcrowding needs to be avoided for the well being of everyone. People have been seen standing in queues and lines outside the vaccination centres waiting for their turn and this leads to large crowds being formed which is extremely dangerous. So in this paper our motive is to design a virtual queue system which implements virtual queues and will ensure social distancing by letting the people know their turn or number in the line or queue virtually.

KEYWORDS:

Healthcare System , Web Development, Queue.

INTRODUCTION:

After one and a half year of covid -19 pandemic when the vaccines finally hit the vaccination centres, large crowds of people could be noticed outside of the vaccination centres each one of them waiting for their turn to get vaccinated. This can further lead to the spreading of the covid-19 virus which could lead to another wave of the virus. So for these kind of situations a virtual queue system can a very useful solution. Virtual queueing takes full advantage of digital solutions and devices and help smooth out the rough corners of the waiting experience. The system takes into account all customers currently in queue and calculates the average wait time, text messages alert the customers of their turn, and many more features are offered in this particular virtual queuing system so that the user can have a stress free and relaxing experience.

Benefits of Virtual Queues:

1. Enhance customer flow.
2. Decrease wait time and service time.
3. Reduce perceived wait time.
4. Maintain the rules of fair queuing.
5. Monitor and improve staff performance.
6. Increase operational efficiency.
7. Offer a more personalized service experience.
8. Eliminate customer anxiety.
9. Increase customer satisfaction.

LITERATURE SURVEY

- Virtual Queue as a tool for social distancing:

Virtual queuing is about managing customer journeys in the digital space, so they have minimal close interactions with other people. Customers do virtual check in and wait remotely, and staff can also keep their distance when serving. A few ways virtual queuing solutions can help with social distancing:

- Regulating the number of people visiting your premises
- Keeping close interactions to a minimum
- Reducing the time customers need to spend on-site
- Managing customer flow

➤ Virtual Queuing in Healthcare:

With a virtual queue management system, healthcare facilities can manage patient flow, limit the number of people on premises, etc. This can be done by using allowing patients to wait somewhere else and only enter the healthcare facilities when it's their time and optimize staff allocation to improve the service. Better staff allocation and more efficient service can also contribute to the increase of patient satisfaction and reduction in staff's stress levels.

By significantly reducing wait time, virtual queue systems help deliver an exceptional patient experience. In addition to that, they also improve the experience for caregivers. Real-time virtual queues help improve the operational efficiency at your hospital or clinic. Hospital staffers spend less time managing the waiting room by letting the patients and the system take care of registrations — and more time delivering high-quality care. The staff can communicate with patients through text messages. Patients are notified of delays and estimated wait times, reducing frustration and improving the visitor flow.

➤ Uses of Virtual Queues in other sectors:

- Retail:

This is the most basic use case, as virtual queuing applies itself incredibly well to retail. The benefits that retail gets from virtual queue management are also fairly straightforward. Reduced wait times and decreased perceived wait times are two important perks that help keep frustration and anxiety at bay. Moreover, real-time analytics help improve the team's productivity and make informed decisions based on the actual data.

- Banks:

Despite the rapid growth of online banking, in-person banking remains strong 60% of American still prefer opening a new account in person to using a digital device. And where there are physical lines, there is anxiety, confusion and dissatisfaction. Virtual queuing helps banks eliminate the drawbacks of physical waiting lines and give time back to their customers.

- Universities:

Traditional, outdated ways of queue management, such as stanchions and paper tickets, don't let you effectively manage long student lines. Students expect high-quality service and tend to be more vocal about not receiving satisfactory experiences. Virtual queues reduce physical traffic in the registrar's office, library and other key locations at educational facilities. The administration can plan their workload more efficiently while preventing service bottlenecks from forming. Employing a virtual queue system allows your staff to focus on their tasks, and your students to focus on their studies.

- Government offices:

Wait time is one of the biggest universal criticisms applied to government offices. Having a virtual queuing tool in place helps mitigate that while also providing a more personal experience. A self-service sign-in solution drastically improves the citizen flow, not only reducing overall wait times, but also cutting down on operational costs. Real-time updates and text messages keep visitors informed, while in-depth service metrics help understand the situation better. Employees get insights into the actual numbers regarding no-shows, walkaways, service duration, etc. Government offices have to keep public perception and visitor satisfaction in mind, and virtual queuing is the easiest way to satisfy the crowds.

- Industries which can benefit from virtual queuing solutions in Covid-19 situation:

In the current environment where only essential businesses and service providers remain open, virtual queuing solutions can offer great advantages in:

- Public sector: Government agencies and services that need to stay open
- Healthcare: Hospitals, clinics, and other healthcare facilities
- Retail: Grocery stores, pharmacies, other essential business stores

But businesses and service providers must be prepared for the significant influx of customers/visitors when restrictions are eased. Businesses and organizations like retail stores, banks, gyms/sport centers, and government offices can improve their efficiency with a virtual queue system, which will greatly help when the in-branch footfall increases. You can also implement appointment booking to further manage the customer flow, avoid unscheduled visits and minimize waiting times.

PRODUCT DESIGN

- Key-Features:
- Just scan the QR code.
- Enter the virtual world of queues and wait for your turn to arrive.
- Timely notifications/sound alerts will keep the user updated about his position/Time Left in the Queue.
- Automated Check-in Authentication System for Following the Queue.
- Reduces Crowding to Great Extent.
- Efficient Operation with Minimum Cost/No additional hardware required.
- Completely Contactless

- For front end we will use React JS:

React.js is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications. It's used for handling the view layer for web and mobile apps. React also allows us to create reusable UI components. React was first created by Jordan Walke, a software engineer working for Facebook. React first deployed on Facebook's newsfeed in 2011 and on Instagram.com in 2012.

React allows developers to create large web applications that can change data, without reloading the page. The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in the application. This corresponds to the view in the MVC template. It can be used with a combination of other JavaScript libraries or frameworks, such as Angular JS in MVC.

React JS is also called simply to React or React.js.

React.js properties includes the following

- a) React.js is declarative
- b) React.js is simple
- c) React.js is component based
- d) React.js supports server side
- e) React.js is extensive
- f) React.js is fast
- g) React.js is easy to learn

- Reasons for using React Js:

- a) Simplicity

ReactJS is just simpler to grasp right away. The component-based approach, well-defined lifecycle, and use of just plain JavaScript make React very simple to learn, build a professional web (and mobile

applications), and support it. React uses a special syntax called JSX which allows you to mix HTML with JavaScript. This is not a requirement; Developer can still write in plain JavaScript but JSX is much easier to use.

2. Easy to learn

Anyone with a basic previous knowledge in programming can easily understand React while Angular and Ember are referred to as ‘Domain-specific Language’, implying that it is difficult to learn them. To react, you just need basic knowledge of CSS and HTML.

3. Native Approach

React can be used to create mobile applications (React Native). And React is a diehard fan of reusability, meaning extensive code reusability is supported. So at the same time, we can make IOS, Android and Web applications.

4. Data Binding

React uses one-way data binding and an application architecture called Flux controls the flow of data to components through one control point – the dispatcher. It's easier to debug self-contained components of large ReactJS apps.

5. Performance

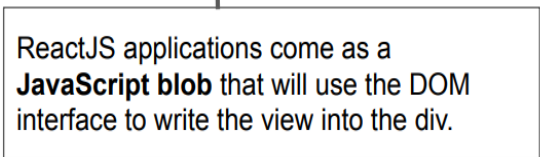
React does not offer any concept of a built-in container for dependency. You can use Browserify, Require JS, EcmaScript 6 modules which we can use via Babel, ReactJS-di to inject dependencies automatically.

6. Testability

ReactJS applications are super easy to test. React views can be treated as functions of the state, so we can manipulate with the state we pass to the ReactJS view and take a look at the output and triggered actions, events, functions, etc.

- ReactJs Web Application page:

```
<!doctype html>
<html>
  <head>
    <title>CS142 Example</title>
  </head>
  <body>
    <div id="reactapp"></div>
    <script src="./webpackOutput/reactApp.bundle.js"></script>
  </body>
</html>
```



ReactJS applications come as a **JavaScript blob** that will use the DOM interface to write the view into the div.

- Steps to create a React app:

Step 1. Create a simple HTML file.

```
01. <html>
02.   <head>
03.     <title>Let's React with npm</title>
04.   </head>
05.   <body>
06.   </body>
07. </html>
```

Step 2. Import React library

```
01. <html>
02.   <head>
03.     <title>Let's React with npm</title>
04.     <!-- Load React Libraries -->
05.     <!--
06.     - Note: when deploying, replace "development.js" with "production.min.js". -->
07.     <script src="https://unpkg.com/react@17/umd/react.development.js" crossorigin>
08.     </script>
09.     <script src="https://unpkg.com/react-dom@17/umd/react-
10.     dom.development.js" crossorigin></script>
11.   </head>
12.   <body>
13.   </body>
14. </html>
```

Step 3. Placeholder for React Component

```
01. <body>
02.   <div id="root"></div>
03. </body>
```

Step 4. Create a React Component

```
01. class HelloClass extends React.Component
02. {
03.   render()
04.   {
05.     return React.createElement('div', null, 'React without npm');
06.   }
07. }
```

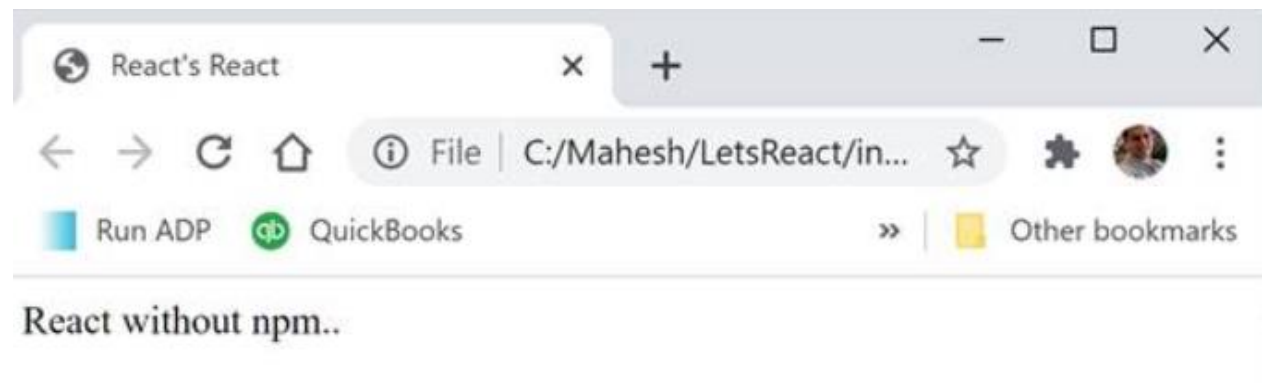
Step 5. Call React Component

```
01. ReactDOM.render(  
02.   React.createElement(HelloClass, null, null),  
03.   document.getElementById('root')  
04. );
```

Step 6. Complete code

```
01. <html>  
02.   <head>  
03.     <title>React's React</title>  
04.     <!-- Load React. -->  
05.     <!--  
06.     - Note: when deploying, replace "development.js" with "production.min.js". -->  
07.     <script src="https://unpkg.com/react@17/umd/react.development.js" crossorigin>  
08.   </script>  
09.     <script src="https://unpkg.com/react-dom@17/umd/react-  
10.     dom.development.js" crossorigin></script>  
11.   </head>  
12.   <body>  
13.     <div id="root"></div>  
14.   <!--  
15.   - This is embedded JavaScript. You can even place this in separate .js file -->  
16.   <script>  
17.     window.onload = function()  
18.     {  
19.       class HelloClass extends React.Component  
20.       {  
21.         render()  
22.         {  
23.           return React.createElement('div', null, 'React without npm..');  
24.         }  
25.       }  
26.       ReactDOM.render(  
27.         React.createElement(HelloClass, null, null),  
28.         document.getElementById('root')  
29.       );  
30.     };  
31.   </script>  
32. </body>  
33. </html>
```

STEP 7. RUN REACT CODE



- For building OTP API we will use Flask.

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.

Flask is part of the categories of the micro-framework. Micro-framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins. In the case of Flask, its dependencies are:

- I. Werkzeug a WSGI utility library

II. jinja2 which is its template engine

- Whole thing will be built on node server.
- Implementing Queue Data structure in MongoDB.

MongoDB is an open-source document database and leading NoSQL database. MongoDB is written in C++. This tutorial will give you great understanding on MongoDB concepts needed to create and deploy a highly scalable and performance-oriented database.

This tutorial is designed for Software Professionals who are willing to learn MongoDB Database in simple and easy steps. It will throw light on MongoDB concepts and after completing this tutorial you will be at an intermediate level of expertise, from where you can take yourself at higher level of expertise.

Before proceeding with this tutorial, you should have a basic understanding of database, text editor and execution of programs, etc. Because we are going to develop high performance database, so it will be good if you have an understanding on the basic concepts of Database (RDBMS).

➤ Architectural Design:

In our case the main program consists of the homepage which is the consists of login and register components to pass onto the next subroutine. The subroutine has the task to take registration details of the user and merchant

and store them in database for access in the next subroutine according to the type of registration. The subroutine 2 is for user's page and subroutine 3 is for merchant page.

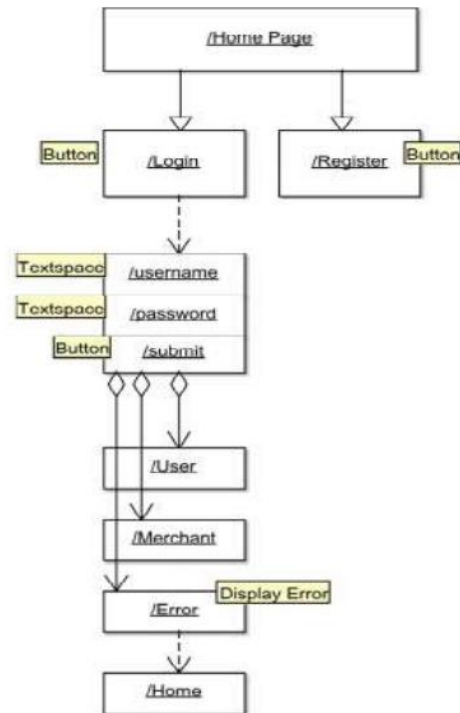
The architecture of a software system has a noteworthy effect on framework execution, effectiveness, security, and viability. It is the essential antique for conceptualization, building, and overseeing of the framework a work in progress.

MODULES DESCRIPTION

➤ Module 1:

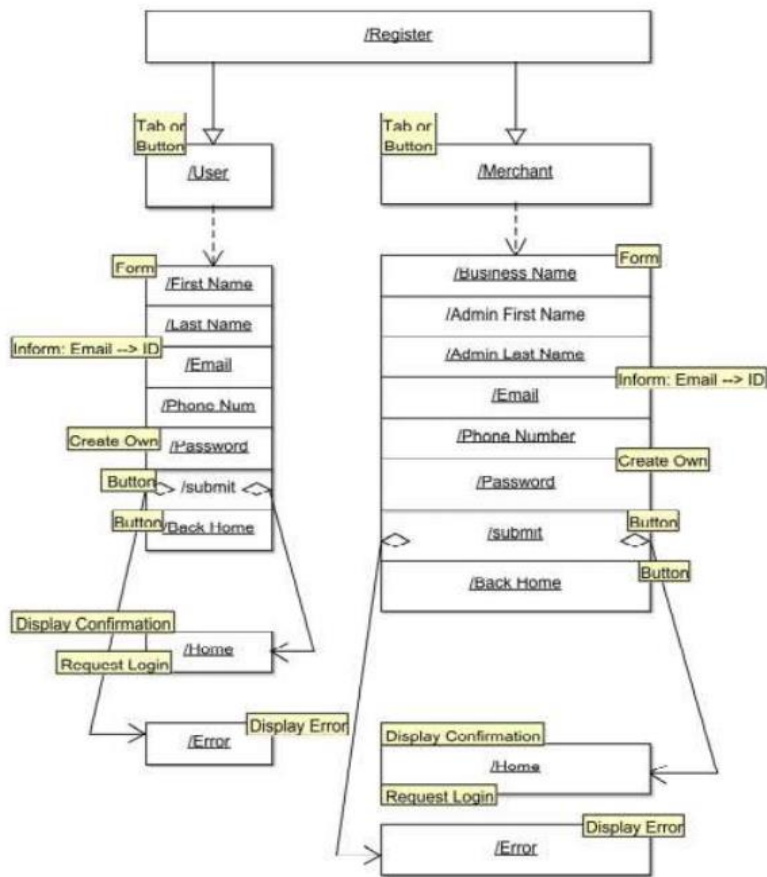
The login and register is the main component which is going to be implemented in module 1. The homepage (module 1) consists of all the necessary contents in login which is the username and password. It also includes path to direct to the registration page. The figure below is the view page for homepage.

- The login will direct the person to either merchant or user page.
- The register will direct the person to register as a user or a merchant.



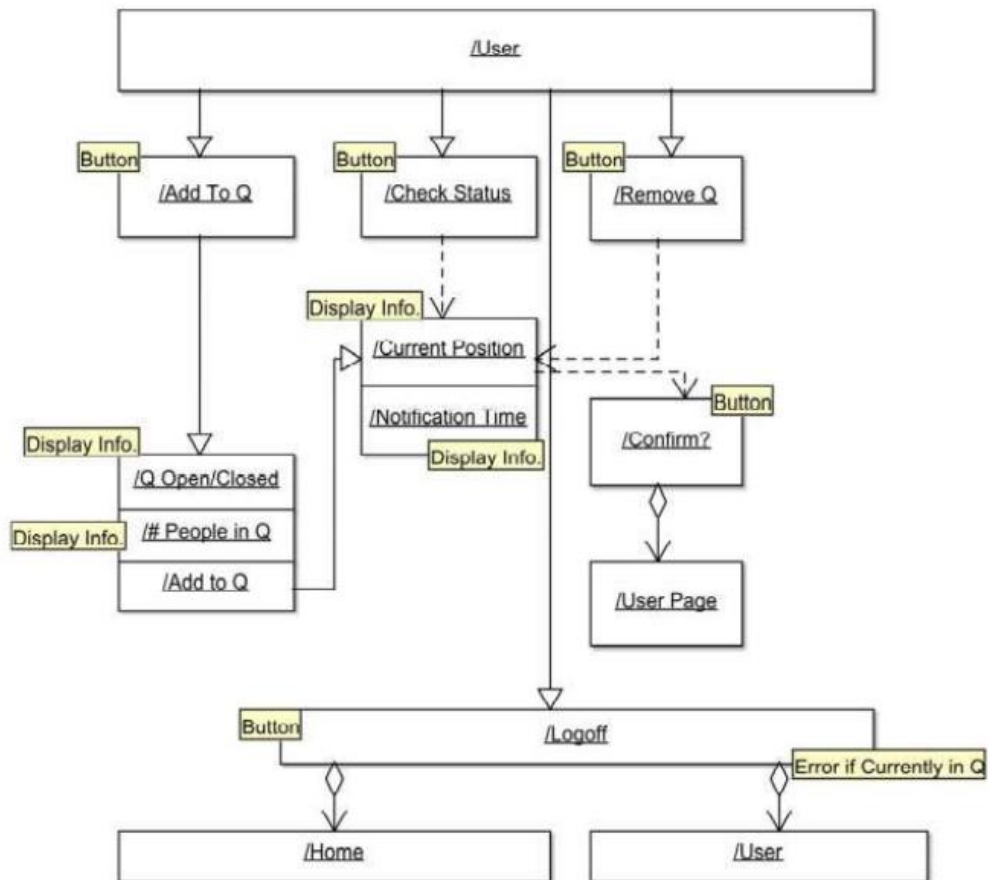
➤ **Module 2:**

The module 2 is the registration page which consists the registration of the user and the merchant. The user registration consists of first name, last name, email, phone number and password. The user registration consists of first name, last name, email, phone number password, business name and business address. After the individual has filled all the detailed and clicked on submit button their respective registration page then the details are collected and stored database using Mongoose. Then the individual is directed to the login page. If the individual enters wrong details, then error message pop up.



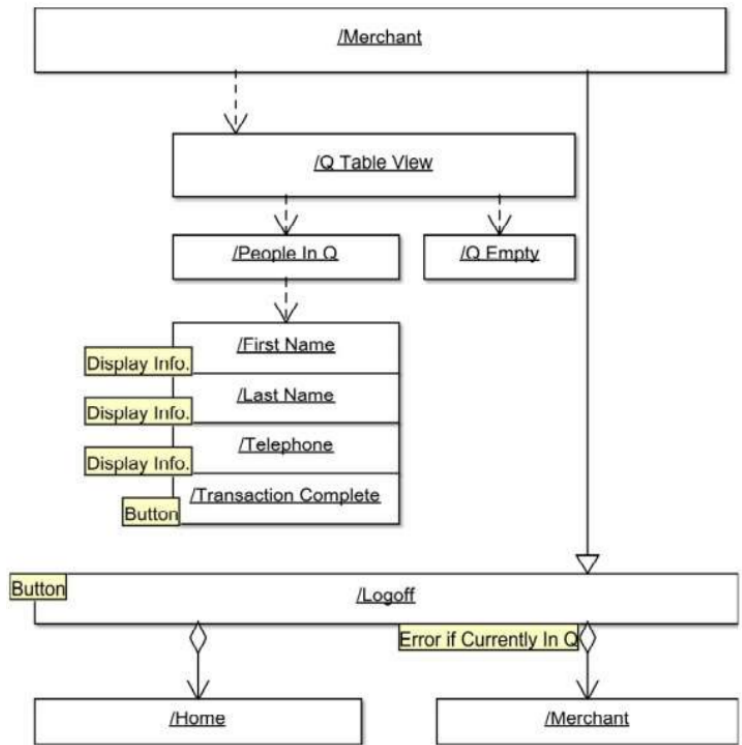
➤ **Module 3:**

This module 3 consists of the user's page components. It consists of tasks like to select the business, check status, add to queue and remove from queue. The user first would be greeted with a hello message and then an option to select which queue the user wants to be in. After the user has selected the merchant he wants to be in then the user has the ability to check the status of the queue and add to queue. The add to queue will add the user to the queue and display the position of the user, the average time for one transaction and expected time. The user has the option to remove himself from the queue.



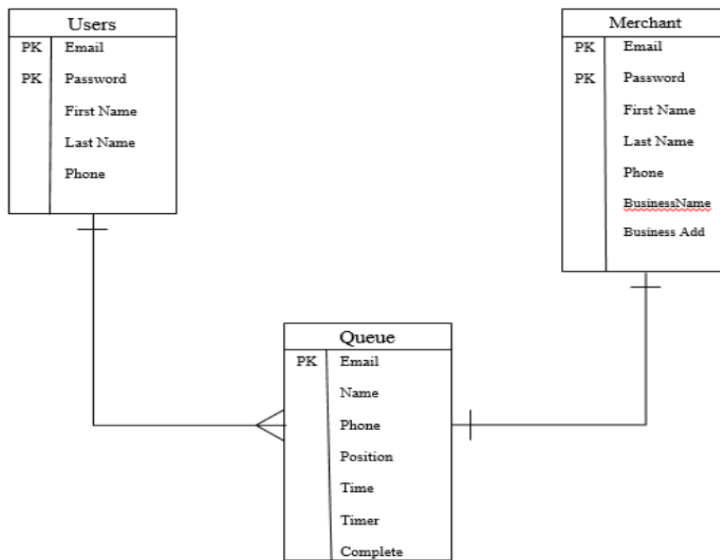
➤ **Module 4 :**

The module 4 consists of the merchant’s page functions. If the individual has logged in as a merchant, then the queue list would be displayed to the individual. The queue list consists of user’s position, name, email, telephone number, time, timer for transaction and a complete button. The merchant has to start accepting users in ascending order. The timer is the time taken for each transaction which would stop when the merchant clicks on the complete button. The timer is allocated so the merchant can wait for a predefined second if the user doesn’t show up. It is also used to calculate average time of each transaction.



➤ **Module 5 :**

E-R Diagram for Virtual Queue System :-

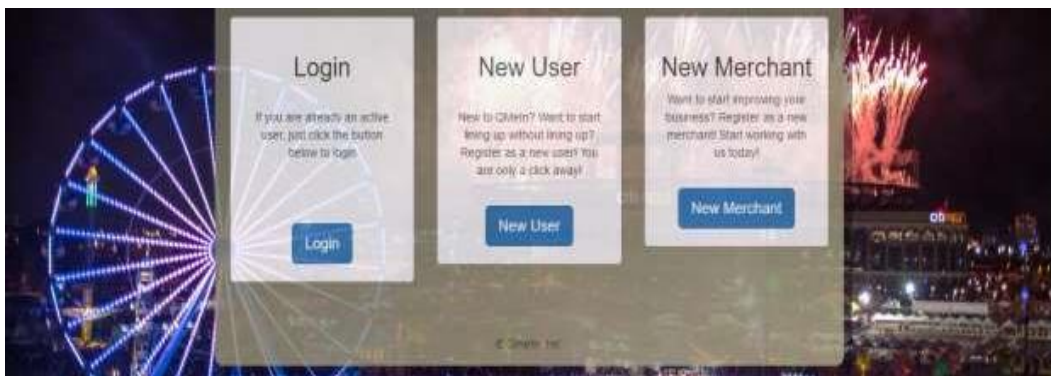


RESULT

The target of the application was that it must be accessible from any platform. Thus the web browser was the most convenient platform for developing this application. The user can access web from any device.

➤ UI Samples:

1. Homepage:



2. Login :



3. Register :

User Registration Form

First Name
Enter First Name Here

Last Name
Enter Last Name Here

Email
Enter e-Mail Address Here

Phone Number
Enter Your Phone Number Here

Password
Enter Password Here

Confirm Password
Confirm Password Here

© 2010 Webkul, Inc.

Merchant Registration Form

First Name
Enter e-Mail Address Here

Last Name
Enter Last Name Here

Email
Enter e-Mail Address Here

Phone Number
Enter Your Phone Number Here

password
Enter Password Here

confirm Password
Confirm Password Here

Business Name
Enter Business Name Here

Business Address
Enter Business Address Here

City
Enter Business City Here

State
Enter Business State Here

4. User :

Hello, **PRATHMESH PARDHIYE!**

You are in the Queue.

There is an 1 user(s) in queue.

Your position in queue is 1

With an average transaction time of 2 minutes, your average wait time is 0 minutes.

5. Merchant:



CONCLUSION

The product implemented will result in smooth working of the places or tasks that require queue management. Ex. – Hospital management, covid vaccinations, Government management, business management, etc.

REFERENCES

- [1] Worthington, D. J.” Queueing models for hospital waiting lists.” Journal of the Operational Research Society 38.5 (1987): 413-422.
- [2] Green, Linda.” Queueing analysis in healthcare.” Patient flow: reducing delay in healthcare delivery. Springer, Boston, MA, 2006. 281-307.
- [3] A. Komashie, A. Mousavi, P. J. Clarkson, and T. Young, “An integrated model of patient and staff satisfaction using queuing theory,” IEEE journal of translational engineering in health and medicine, vol. 3, pp. 1–10, 2015.

- [4] M. Abusair, M. Sharaf, H. Muccini and P. Inverardi.” Adaptation for situational-aware cyber-physical systems driven by energy consumption and human safety.” Proceedings of the 11th European Conference on Software Architecture: Companion Proceedings. 2017.
- [5] Hermanto, Rinda Parama Satya, and Ariadi Nugroho.” Waiting-Time Estimation in Bank Customer Queues using RPROP Neural Networks.” *Procedia Computer Science* 135 (2018): 35-42.
- [6] Guo, Jing, Caixia Zhang, and Xu Zhang.” Smart queue system based on RFID technology for theme park.” 2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS). IEEE, 2016.
- [7] Ghazal, Mohammed, Rania Hamouda, and Samr Ali.” An iot smart queue management system with real-time queue tracking.” 2015 Fifth International Conference on e-Learning (econf). IEEE, 2015.
- [8] Cowdrey, Kevin WG, et al.” Applying queueing theory for the optimization of a banking model.” *Journal of Internet Technology* 19.2 (2018): 381-389.
- [9] Kyritsis, Athanasios I., and Michel Deriaz.” A Machine Learning Approach to Waiting Time Prediction in Queueing Scenarios.” 2019 Second International Conference on Artificial Intelligence for Industries (AI4I). IEEE, 2019.
- [10] Keilson, J., and L. D. Servi.” A distributional form of Little’s law.” *Operations Research Letters* 7.5 (1988): 223-227.
- [11] Dharmawirya, Mathias, and Erwin Adi.” Case study for restaurant queuing model.” 2011 International Conference on Management and Artificial Intelligence. 2011.
- [12] Dixie Thamrin “Virtual queuing system”,2020.
- [13] Dr. Lidia Morrison “Software Design Dewscription”,2017.

