INTERNATIONAL JOURNAL OF SOCIAL
Sciences and Information Technology
ISSN 2412-0294 www.ijssit.com

http://www.ijssit.com

VIRTUAL QUEUE MANAGEMENT SYSTEM

[1*] **Ndung'u Michael Ngugi**
micqualngugi96@gmail.com

[2**] **Josphat Karani Mwai**
jkaraniw@gmail.com

[3**] **Mwangi Njomo**
mwanginjomo2018@gmail.com

[1] *Student, Bachelor of Science in Information Technology, Kirinyaga University, Kenya*
[2, 3] *Tutorial Fellows and Researchers, Kirinyaga University*

Abstract

Waiting in line is an important issue in the context of services owing to its immense impact on operation capabilities and satisfaction of customers of organizations. The mode of ordering queues and how long customers should wait for a service or product has laid the base for determining the most optimal queuing solution that can give a tradeoff towards minimizing operation cost incurred while optimizing queues, and also minimizing the time taken by customers waiting to be served. As the current economies gradually transition from platform dependent to service oriented, it has become trivial to thoroughly grasp how to effectively manage waiting lines and in turn improve on the customer satisfaction in an organization. This project has examined the shortfalls of traditional and current approached to queue management, studied the types and applications of various queue management systems, and finally formulated a general approach and methodology to design and develop the virtual queue management system.

Keywords: *virtual queue management, mobile queues, online tickets, virtual waiting lines*

INTRODUCTION

Having to wait for services in waiting lines has become an anticipated part of modern life, people can be found waiting to be served at banks, retail stores, post offices and even waiting on hold on telephone calls to access service operators. Queueing and waiting in line causes inconvenience and frustration to those waiting to be served. The sheer length of the waiting line can be enough to discourage customers from pursuing valuable services, or even cause customers having important events to either miss or delay them. In today's competitive environment, businesses are occasioned by the need for improving customers satisfaction, normally actuated by having better service operation capabilities to gain competitive edge in the marketplace [1]. Often, this change has led to increased importance and attention towards service operations management. The need to improve customer interaction and service has given rise to CRM innovations that allow new systems to be developed for managing and analyzing customer interactions and data throughout the customer lifecycle [2]. The virtual queue management system is one such system which allows queue management to be provided as a service to customers and eliminates traditional queue ordering rules that are currently designed and implemented in organizations.

I. BACKGROUND

A waiting line occurs within a system involving customers arriving for a product or serving, queuing when the service is not immediate then leaving the queue once the customer has been served. The queuing theory lay the base for finding the most optimal solution to queue management. By definition, 'a queue entails a line of people or Vehicles waiting in succession in turn to be attended to or to proceed [3]'. Sometimes, in places having a lot of people queuing, there may be a system of numbered tickets and when someone arrives, they take a number and wait in line. A Queue management system is a software for queuing customers and gathering their feedbacks, monitoring real-time information and speeding up of services. Currently, queues are implemented in three ways:

i. **Standard Linear Queues:** This form of queuing represents the normal or standard queue system where each service desk or cashier has a separate line [3].

ii. **Single Line Queues:** Also referred to as a Call Forward System, a single line groups customers and then feeds them to multiple cashiers or service areas. Often, Individual service stations are allocated to different service personnel part of one long counter or desk [3].

iii. **Dispersed or "Digital" Queues:** This type of queue management disperses waiting lines by offering a ticketing management system [3].

The order of service of queues specifies the order in which customers are chosen for service within a queue. Among the disciplines under this category are:

i. First Come, First Served (FCFS): this mode is commonly applied real-world situations, such as tellers in a bank.

ii. Last Come, First Served (LCFS): This mode acts a reverse order service given to customer against their arrival.

iii. Priority Discipline (PD): Under this discipline, customers are classified into categories, then each category is given different priorities.

Traditionally, queues are generally FCFS or LCFS and attempts to re-ordering a tradition queue by cutting through lines creates social friction [4]. As a result, queues tend to have long waiting hours, not only due to inadequate service propagation among staff but also due to missed or miscalled tickets that usually lead to unnecessary wastage of time.

PROBLEM STATEMENT

Having to physically queue in line or waiting for hour in waiting rooms and crowded lobbies has grown to be a major community problem. It's strange that with all the advancement in technology, this mode has not changed much over the last millennium. The model of waiting in lines is very inefficient, customers have to be in line in order to be served even if the line may have more than a hundred customers ahead. Despite this fact, service-based organizations that have the longest waiting time are making very little effort to improve on it. Needed is a system which allows queue management to be provided as a service and customers can accurately estimate their wait times and keep them updated as the queue progresses without having to wait in the premise or crowded lobbies.

DEVELOPED SYSTEM

The developed system provides queue management as a service to businesses and organizations. The system allows customers to book services online or within the premise using an in-house device which communicates to a web-based API handling booking requests and dispersing bookings to allocated services. Customers with an android app can use Google maps API to get around locations and view places implementing the service queues. The application can then show profiles, and services offered with the option to get online tickets. Once assigned a "digital token", a customer is updated on the progress of the queue in real time. The mobile app tracks the estimated time for waits and gives notifications on number of people also waiting for their turn. Timely updates such as push notification on android or IOS and SMS using messaging gateways are sent to customers communicating their time to access a service. This enables customers to wait for their turns in any location of their choosing.

The developed system also implements a web-based application that monitors and tracks customers in queues. Agents manning the queues can call customers and provide specialized service. The system also captures key data based on the engagement and can be used in profiling customers and reaching out later based on the contact details given during registration. The developed system enables service-based environment achieve greater efficiency in handling customer queues and offer innovative ways to interact with customers. A novel application of the virtual service queues would be a parking lot company that needs to keep track of vehicles in the parking lot; customers can easily view remaining slots and reserve parking while on the go.

MAIN OBJECTIVE OF THE DEVELOPED SYSTEM

The key purpose of the system is to develop a cloud and mobile based system that addresses the above-mentioned problems and allow queue management to be provided as a service thereby minimizing waiting lines in service-oriented environments.

SPECIFIC OBJECTIVES OF THE DEVELOPED SYSTEM

The project was to achieve the following objectives:

- To analyze how existing system implement service queues.
- To develop a system that allows the use of virtual queues in web-based and mobile platforms.
- To implement USSD and messaging services for notifications.
- To provide security of the information gathered by the system.

CONCEPTUAL FRAMEWORK

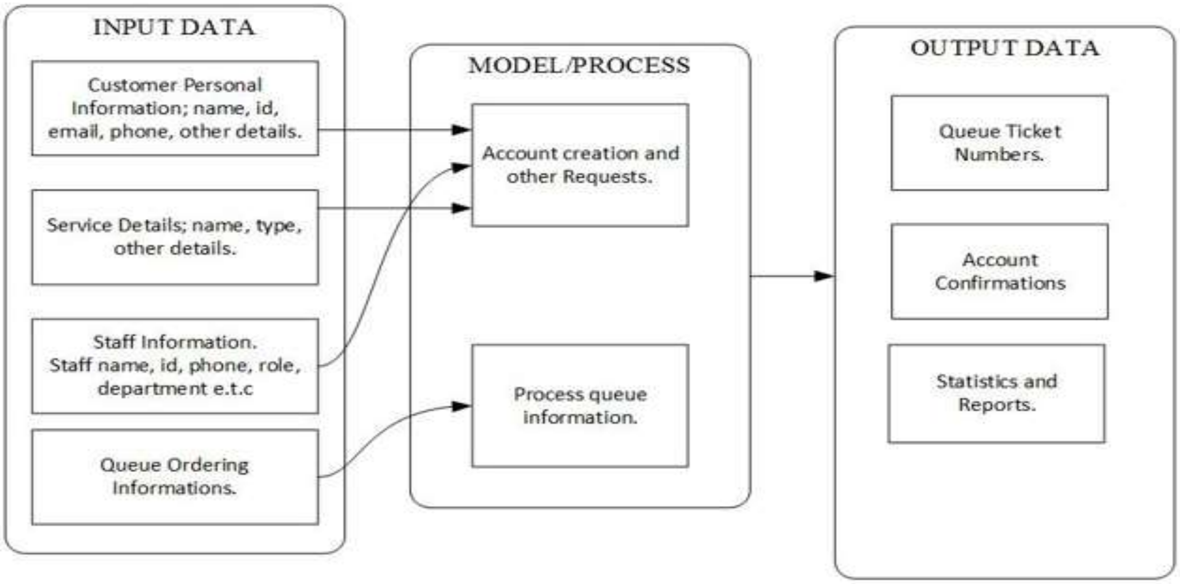

INPUT DATA
Customer Personal Information; name, id, email, phone, other details.
Service Details; name, type, other details.
Staff Information. Staff name, id, phone, role, department e.t.c
Queue Ordering Informations.
MODEL/PROCESS
Account creation and other Requests.
Process queue information.
OUTPUT DATA
Queue Ticket Numbers.
Account Confirmations
Statistics and Reports.

Figure 1 The conceptual framework

JUSTIFICATIONS

The developed system allows multiple customer profiles to be handled impartially and in a fair manner and also provides a way for customers to effortlessly and easily access services without having to wait in line. The developed system also provides timely updates and notifications alert customers as they move to the front of the queue. Fast and efficient queue scheduling enables them feel empowered and in control of their time. As a result, customers can browse and shop while they wait, which was to in turn increase impulse purchases.

LIMITATIONS

The project had the following projected limitations:

- Most features of the system will available on smart devices, other devices will to rely on third party API.
- Less technologically savvy people may be overwhelmed with an advanced system.

SCOPE

The scope of the developed system Project was planning, designing, developing, and implementing a Restful API where all virtual queues would be managed and a mobile application for booking tokens. The system would also provide access to an administrator dashboard with key metrics for showing various reports and statistics on status of queues and also permissions that would enable authorized users to update, manage, and view queues. Administrative dashboards would also contain daily queue statistics, call centers, Department data, counter and staff data and Reports view. The mobile app was to provide a list of service centers and booking options.

II. LITERATURE REVIEW

INTRODUCTION

This literature review was undertaken primarily to underpin the implementation of Virtual queue management systems, how they are structured, designed and implemented in many different organizations. It provides the systematic background information, key features and implementation in the organizations across the country and in other parts of the world.

TRADITIONAL SYSTEMS

Traditional queuing solutions involve linear queues where customers wait in line physically. Not only is it the oldest form of queuing, but also the most common way for managing queues and waiting. There are several variations of Linear queues:

Single Queue Single Service Point (SQSSP)

This variation offers a First in First out model, customers are served in the order in which they arrive. Typically, the SQSSO is found in service centers having few customers such as fast food restaurants and coffee shops.

Multiple Queue Multiple Service Points (MQMSP)

This variation segments customers based on their service needs such that those requiring longer service times are separated from those with smaller requirements.

Single Queue Multiple Service Points (SQMSP)

Services are distributed in a logical manner and gives priority to customers to entered the queue first. Slow customers may affect the overall delivery but every customer will be directed to the available counter.

CURRENT SYSTEMS

WAVETEC: VIRTUAL QUEUE MANAGEMENT SYSTEM

The queue management system provided by wavetec offers standard and customizable features for queue management. These features are geared towards solving queuing problems faced by organizations by offering diverse queuing solutions. Businesses have options for basic, enterprise or mobile queuing system with each able to manage and disperse crowds effectively. Customers are able to seamlessly journey through the system through their virtual and linear queuing systems and also access an appointment booking option. Wavetec also comes with premium services which integrate enterprises that offer sophisticated or multi branch and multi region services with their distributed queuing solutions. Customers interact with the systems via inhouse channels such as self-service desks, websites or a mobile app and receive a ticket.

Key Features

i. Accurate prediction of waiting times from TV monitors and notifications.

ii. Business are able to customize the queue solutions to meet their needs.

iii. Wavetec systems can easily integrate with other preexisting platforms.

iv. Customers are engaged with digital signage whilst they wait in line.

CASE STUDY

Huduma Kenya: This is a one stop shop initiative by the Kenyan government aimed at transforming the delivery mechanism of public service. Following an understanding of customer footfalls and retail services offered at Huduma centers, Wavetec was able to install complete queue management system comprising of dual printer touch screen device, customer dispatching and calling and display system. The system is built to customize the specialized workflows prevalent at Huduma Centers. As traffic in these Huduma centers increased with over 20 public services, more counters were needed to cater for the high influx using a systematic and automated queueing system.

Banks in Kenya: There had been an increase in the number of customers in banks at their various branches. This brought about problems of dissatisfied customers who complained of inefficient services. KCB (Kenya Commercial Bank), one of the leading banks in Kenya, unveiled the first queuing solution and distributed it to their service centers in an attempt to disperse long queues.

QLESS: MOBILE QUEUE MANAGEMENT SYSTEM

QLess a company that offers software as a service (SAAS) provides a queue management system that helps eliminate waiting lines. The system is developed to enable customers wait in a virtual line instead of a physical one. Once a customer joins one virtual line, the customer receives an SMS text, notification, audio-visual or voice call to keep them notified as they proceed to the service area.

QLess allow manages to allocate users to a virtual line in the following ways; A touchscreen device or website can be located in the physical location of the key and customers are able to join the queue from this device; Using a mobile app provided by QLess, users can get virtual token; sending SMS text or receiving a call through an automated voice call. Business using QLess obtain reports and statistics on their busiest hours, customers cancelling or returning.

Key Features

i. QLess offers virtual queues accessed via websites, a mobile app or onsite kiosk.

ii. Appointment booking enabling easy walk in if scheduled and reducing long gaps especially on busy days.

iii. Timely updates enabling interactive communication. This is through SMS text and app notifications to alert customers on the status of the queue CASE STUDY

CASE STUDY

Medical professional at a Nevada-based (USA) Renown Medical Group needed to improve their patient service flow. After several queuing options, they finally settled on QLess. This helped patients skip waiting rooms by booking queues online and join virtual lines either from home or work enabling them arrive just in time for their appointment. The integration was very effective such that customers would be alerted 15 minutes in advance of appointment either by text or notification and also disperse all bottlenecks in service areas. Staff could then monitor and manage real time data provided by the system, and also obtain information on delays to better control expectations and minimize frustrations.

Q-SYS: QUEUE MANAGEMENT SYSTEM

Q-SYS is part of the RIANA Group of companies, which is a leading business solutions provider in Kenya. Q-SYS has been identified as one of the strongest brands in the region by London based The Centre for Brand Analysis and has achieved the status of Super brand East Africa. Q-SYS allows businesses to track, monitor, regulate and control visitor movements within business facilities. Q - SYS is currently implemented in service centers in banks, hospitals, embassies, government offices, ticket counters, or any place where clients queue up to access specialized services.

Key Features

Q-SYS offers management of customer wait times, manages front desk, and reporting, digital signage and real time monitoring. With the tools provided, business can obtain real-time access to performance metrics such as customer in flow, customer wait time, average service time, employee efficiency and customer feedback.

CASE STUDY

Q-Sys has been able to offer industry solutions for the healthcare, retail and finance sectors along with reliable follow-up tools. KCB (Kenya Commercial Bank), equity Bank, Alpha Commercial Bank, National Bank of Rwanda (BNR) and several other firms are some of the clients currently implementing service queues provided by Q-SYS.

SKIPLINO: MOBILE QUEUE MANAGEMENT APPLICATION

Skiplino is a multilingual system for queue management allowing customers to be managed efficiently and with easy. This is made possible by offering an integrated cloud based queueing solution that monitors data relating to queue in real time and obtain customer feedback.

Key Features

i. Tracks how long customers have been waiting in line and provide instant feedback. This helps avoid preventable customer complaints that can damage the company reputation.

ii. Branch management monitoring regardless of geographical distance from headquarters.

iii. Fast and easy response time to demands from visitors and customers.

LACUNAS

Uneven Queue Distribution

Customers can be discouraged to wait in line when on premise, they find a single long queue. This situation may be caused by lack of planning on lines and queue distribution. This problem can easily be solved by incorporating such a system that can reliably predict queueing patterns and evenly distribute customers to queues.

Service Based Queue Delivery

The current industry is a service dominating one. Despite this, most queue implementation are still offered inhouse. This entails that huge investments in infrastructure and logistics must be planned beforehand to implement a queueing system. With a mobile one, services can be added on the fly and businesses and organization can access the Virtual queue services by simply creating an account.

Enhanced HCI

This refers to how the deployment of the system is done, identified in the user interface and the overall user experience (UI/UX). The current standard for implementing queue management systems use desktop applications and traditional mobile applications. The system was to take advantage of the growing trends in user experience to enhance customer interaction and give a natural feel of use, instead of feeling like an automated process.

CONCLUSION

In conclusion, the key subject areas researched were current implementation of virtual queues that form the basis of the system. This chapter provided a general context of the information gathered by discussing systematic background information, key features and implementation of Queue management systems in organizations across the country and in other parts of the world. The implications of the gaps identified during research was to greatly aid in the design of more effective queue management.

III. SYSTEM METHODOLOGY

INTRODUCTION

System Methodology involves dividing software development work distinct stages and coming up with tasks or activities aimed at achieving better planning and time management. It is considered a trivial part of the systems development life cycle.

THE WATERFALL MODEL

The Waterfall approach is one of the most popular model in Software engineering, it was in fact the very first SDLC model to be used. The approach taken was to treat the whole process of modelling software in a sequential order, the outcome or output of the previous step would serve as the input for the next step. This would enable software developers to separate these steps into distinct stages, with each phase having different requirements and activities. The illustration bellows shows a representation of the stages found in the Waterfall Model.

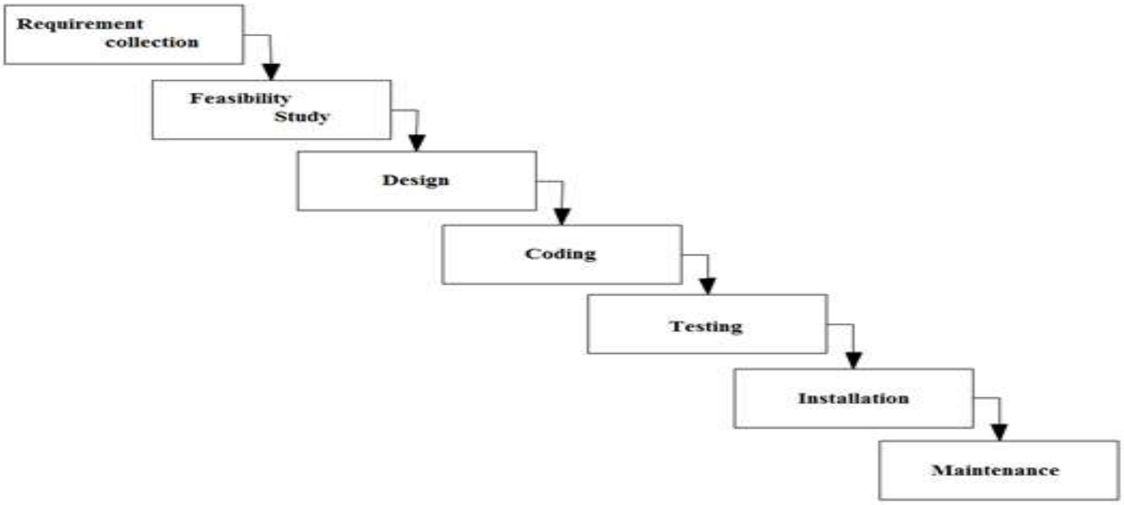


Figure 1 System methodology

The sequential phases in Waterfall model are:

i. **Requirement Gathering and analysis:** This phase involved performing a feasibility study and ascertaining all possible requirement of the system.

ii. **System Design:** This phase involves getting specifications, functional and non-functional requirements of both software and hardware required to develop the system. This step also involved defining the system architecture.

iii. **Implementation:** With outcome from system design of the developed system prototype, the system was developed in small programs called units, which were pushed to the next stage to be tested.

iv. **Integration and Testing:** All the units developed in the implementation were integrated into the system, isolated and fully tested. Once verified to be working, the entire system was tested to detect any undetected failures or faults.

v. **Deployment of system:** Once testing was done, the developed system was deployed and tested with live customer data.

vi. **Maintenance.** Maintenance of the developed system was to be done in continuum to ascertain that all functional and nonfunctional requirements were reached and any changes to the software and hardware would not affect the customer environment.

JUSTIFICATIONS

Waterfall model is preferable as it has provision for changes and the changes can be implemented in the maintenance phase. This is because the waterfall model is simple and easy to understand and use for the developer and the other users. This model also allows for early design changes and places emphasis on requirement and design before writing any single line of code which ensures minimal time wastage and effort in design changes.

IV. SYSTEM DESIGN

INTRODUCTION

System design is an important part of any system that is worth implementing. In this section we shall look into the hardware, functional and non-functional system requirements, data aspect and process design for the Virtual Queue Management System (VQMS).

HARDWARE REQUIREMENTS

Consist of hardware requirements to be met in order to successfully run the system:

Desktop Computer

To be used for design and development of the system. The computer had these specifications:

i. Intel® Pentium® processor (or equivalent) with a speed of 2.50GHz or greater

ii. At least 2GB RAM

iii. 30 GB (or larger) Hard Disk Drive

Tablet or Smart Phone

It was to be used for registering and login in users on mobile, joining nearby queues and sending text or push notifications in real time.

SOFTWARE REQUIREMENTS

To successfully run the system, there are a number of software requirements had to be met which were:

i. **Operating System:** Windows 7 or higher versions of OS (either x86 or x64) or Alternative Unix based systems like Ubuntu or Linux Mint.

ii. **Database Management System:** A database to store the details of various patients, specialists and appointments. The following databased was to be used for the web server and the mobile-based application.

iii. **Firebase Cloud storage:** This was to be for data exchange between the application and the web server.

iv. **MySQL Database:** For the Web server running the application.

v. **WampServer 2.1:** Has been used as a web server.

vi. **Android Studio:** At least API 7 or above.

vii. **Programming Languages:** PHP, HTML, JavaScript, CSS

FUNCTIONAL REQUIREMENTS

Functional requirements capture the intended behavior of a system, and thus, a way of providing a structured functional blueprint, useful for both developers and users. Having the functional requirements in mind, we define use-cases for better guiding the interfaces development from a user perspective and proceed with a general use-case diagram for the overall picture.

High-Level Use Case Diagram

In modelling literature, use cases are described as “interactions with a specific goal between actors and the system under consideration”. Actors are external parties to the system that interact with it, possibly being classes of users or roles, a user can play. This system was to be used mainly by two parties: the staff, and the customers. System actors and their goals, that is, the actions they can perform in the system, are now listed.

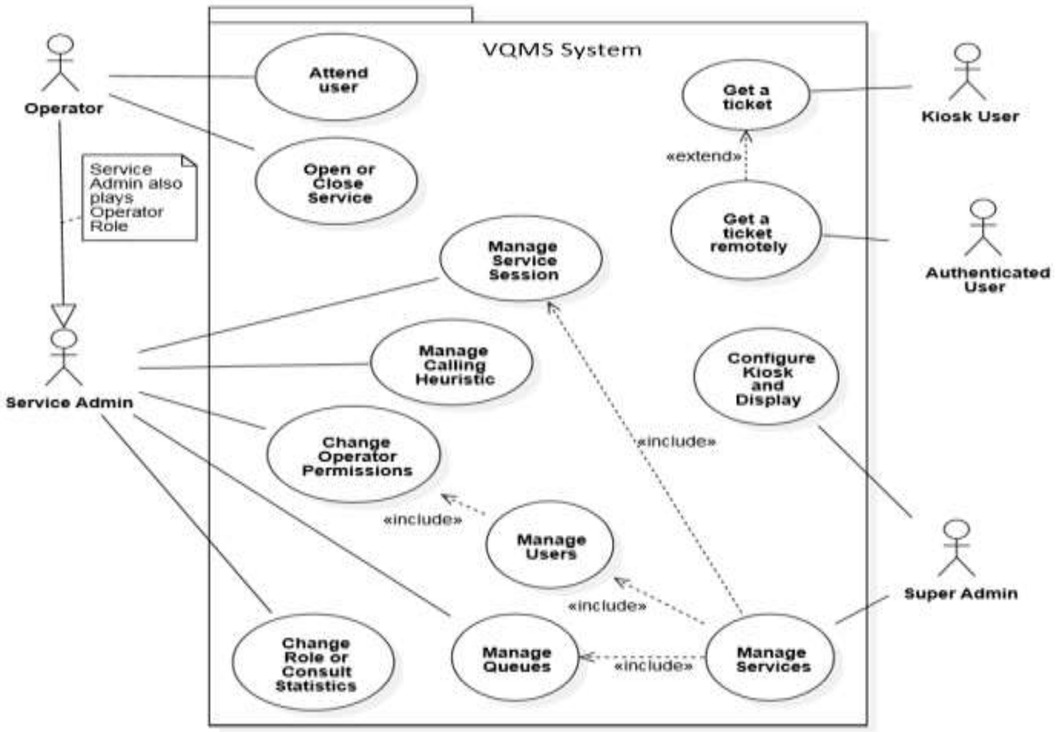


Figure 2 High Level Use case Diagram

System actors and what they should be able to do with the system are now listed.

1) Staff

Super admin

Login and logout to and from the super admin interface. Create/Modify/Delete Services for a generic service provider entity. The admin can define service open-hours (service management) and maximum number of queues for certain service (service management). The Super Admin doubles as the technical administrator and maintainer of the whole system. This means he was to have access to all the created data and used technologies.

Service Admin

He or she can login and logout to and from the service admin interface, this allows him or her to give and remove service admin privileges to and from operators, View and edit Service Settings, use Operation Mode and visualize Statistics and access other setting.

Operator

Upon login, the operator is prompted to select a desk number, call a customer to his desk, Open or Close the service, that is, stops tickets creation.

2) Customers

Kiosk Customer: Get a ticket for a queue that categorizes this customer’s issue. The kiosk customer is identified by the ticket and gets called by the service to solve an issue.

Authenticated Customer: Log in the appropriate service application, mobile or web, and get a virtual ticket for a queue categorizing this customer's issue. Get notifications about that queue's progress, until called by the service to solve the issue, given that the customer used the given info to approach the service in time.

Additional Functional Requirements

Security Requirements

Also, the kiosk interface shall not have a direct internet connection, to prevent tampering from the customer's side. The kiosk must also prevent clients that may request many tickets for the ill-purpose of wasting resources (e.g. paper), by having an acceptable (0.5-1s) time-wait cool down between prints, besides blocking ticket printing request while printing the ticket. The printing of tickets may only be authorized to the physical ticket dispenser or to a user that is authenticated.

Integration Requirements

Authorization backend customization: staff should be able to login into back-office operation with already existing login back-end system.

Mobile App customers should be able to request remote tickets, with the app and respective notification service also using the previously integrated authentication backend.

NON-FUNCTIONAL REQUIREMENTS

This system should be easy to work with for both customers and staff (user friendly interfaces), customizable and deployable for different service provider entities (e.g. other universities, hospitals, etc.) and achieve cost-effectiveness. It should be server-based, easy to configure and scalable, ideally enabling the remote deployment of client units that was to self-configure upon server connection, making it easier to deploy in large organizations and extensible to provide interfaces to devices external to this system.

PROCESS DESIGN: CONTEXT LEVEL DFD

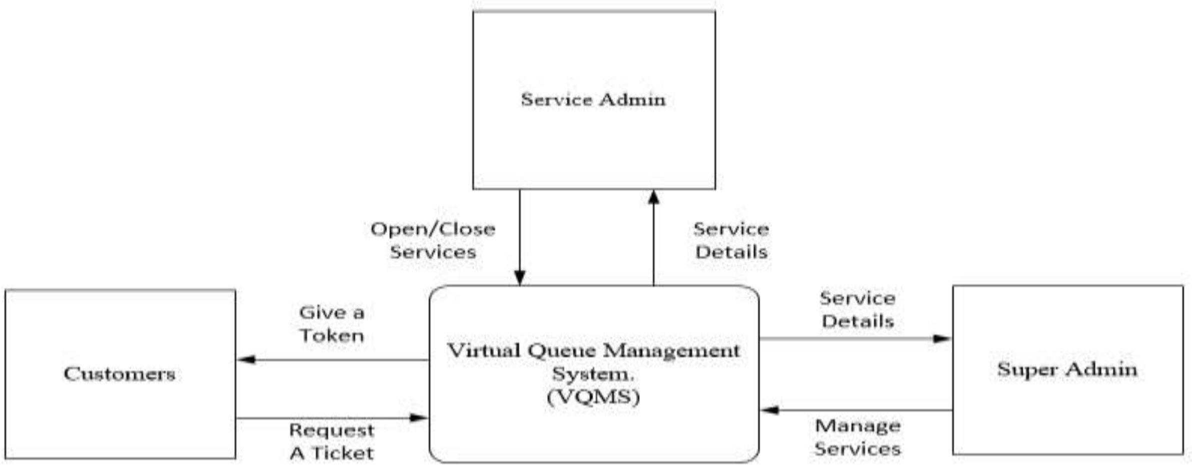


Figure 3 Context Level DFD

Flow chart

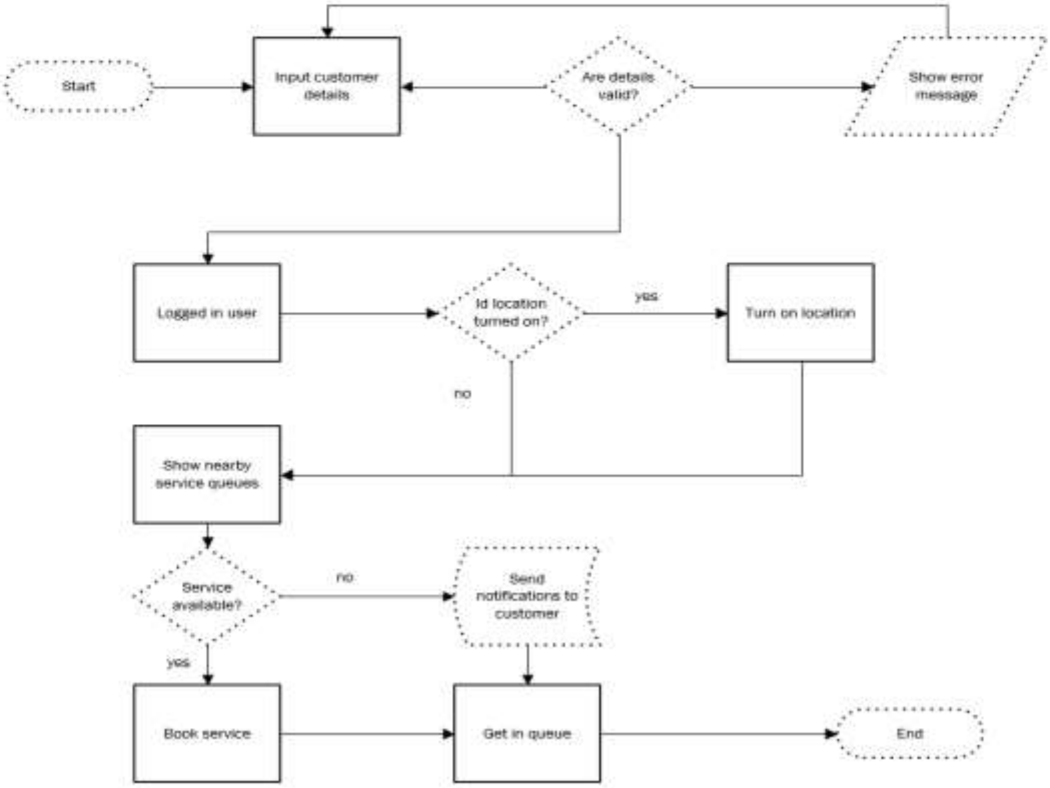


Figure 4 Flow chart Diagram

Data Flow Diagram

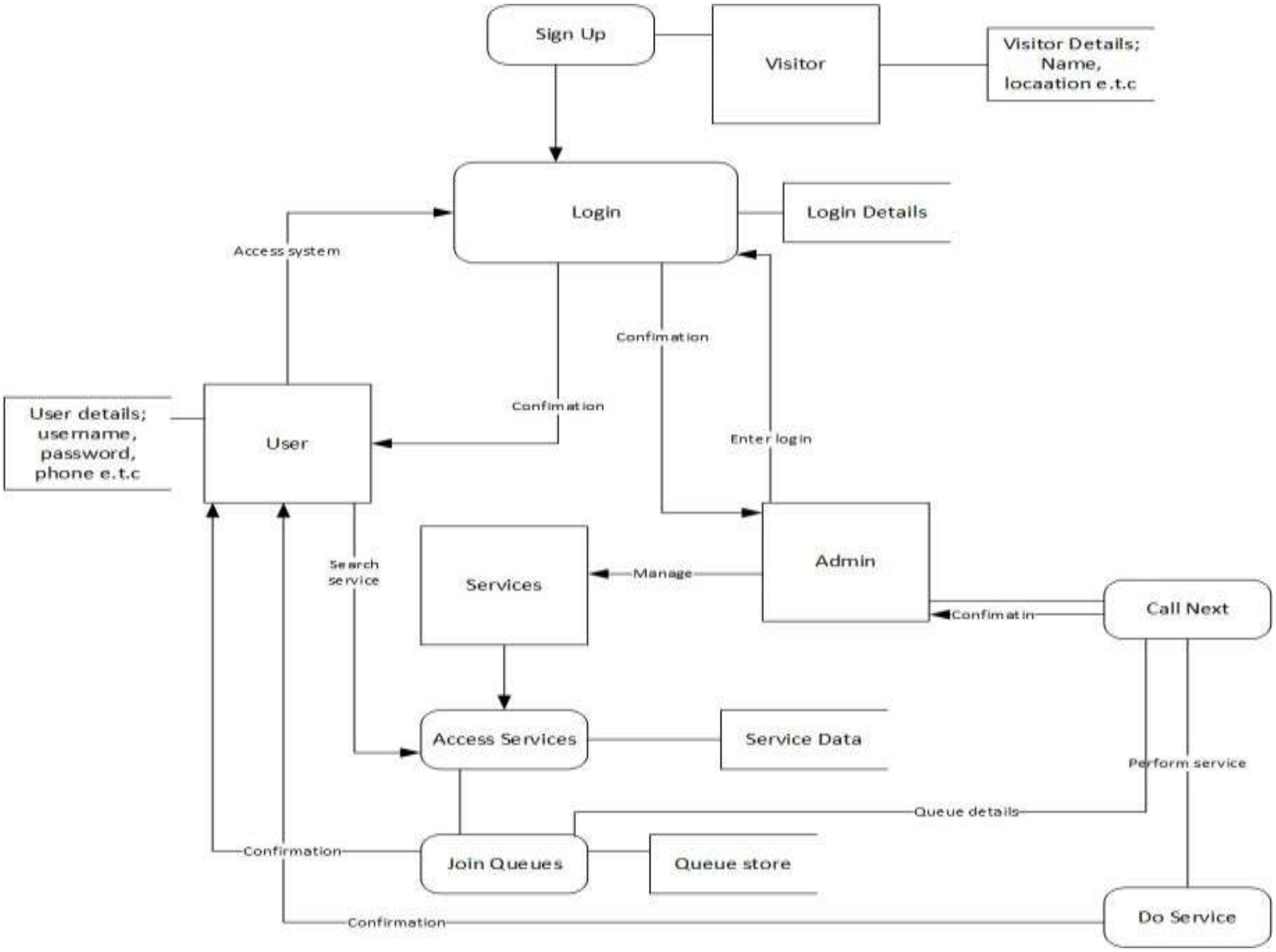
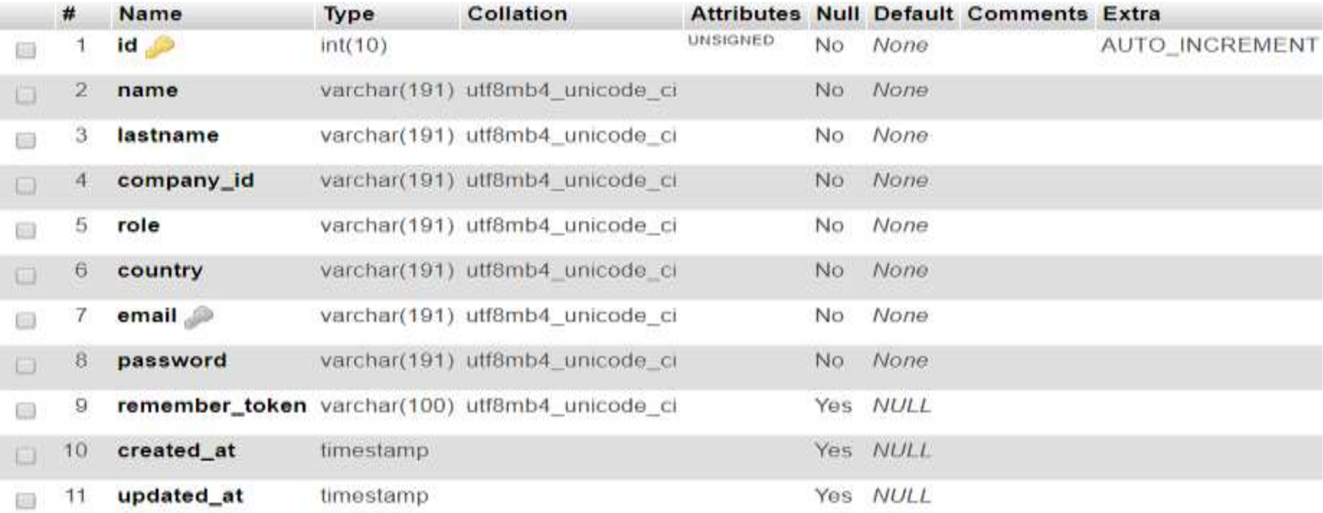


Figure 5 Data Flow Diagram

DATABASE DESIGN

Tables

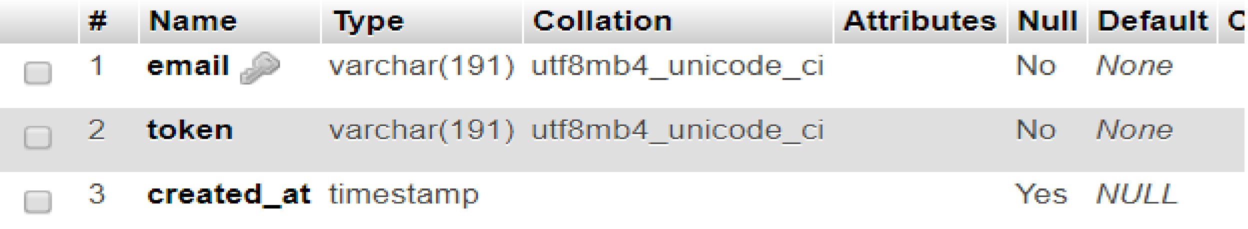
Users Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**id** 🔑	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
☐	2	**name**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	3	**lastname**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	4	**company_id**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	5	**role**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	6	**country**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	7	**email** 🔑	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	8	**password**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	9	**remember_token**	varchar(100)	utf8mb4_unicode_ci		Yes	*NULL*		
☐	10	**created_at**	timestamp			Yes	*NULL*		
☐	11	**updated_at**	timestamp			Yes	*NULL*		

Figure 6 Users Table

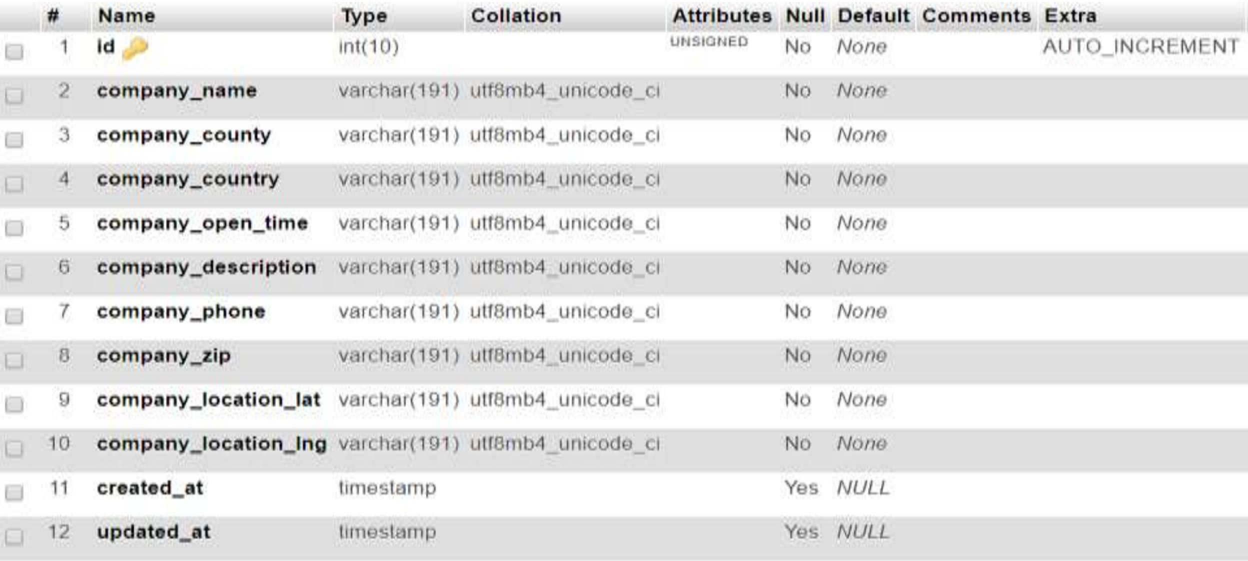
Passwords Reset



	#	Name	Type	Collation	Attributes	Null	Default	C
□	1	**email** 🔑	varchar(191)	utf8mb4_unicode_ci		No	*None*	
□	2	**token**	varchar(191)	utf8mb4_unicode_ci		No	*None*	
□	3	**created_at**	timestamp			Yes	*NULL*	

Figure 7 Passwords Reset Table

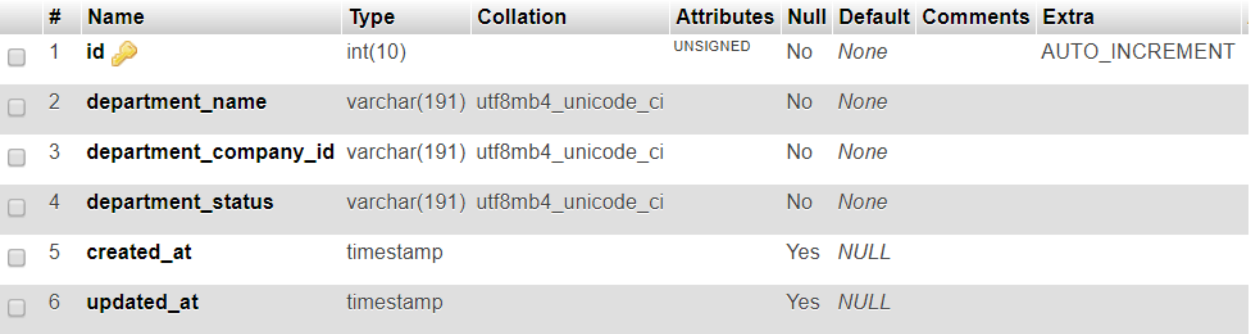
Company Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
□	1	**id** 🔑	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
□	2	**company_name**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	3	**company_county**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	4	**company_country**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	5	**company_open_time**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	6	**company_description**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	7	**company_phone**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	8	**company_zip**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	9	**company_location_lat**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	10	**company_location_lng**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	11	**created_at**	timestamp			Yes	*NULL*		
□	12	**updated_at**	timestamp			Yes	*NULL*		

Figure 8 Company Table

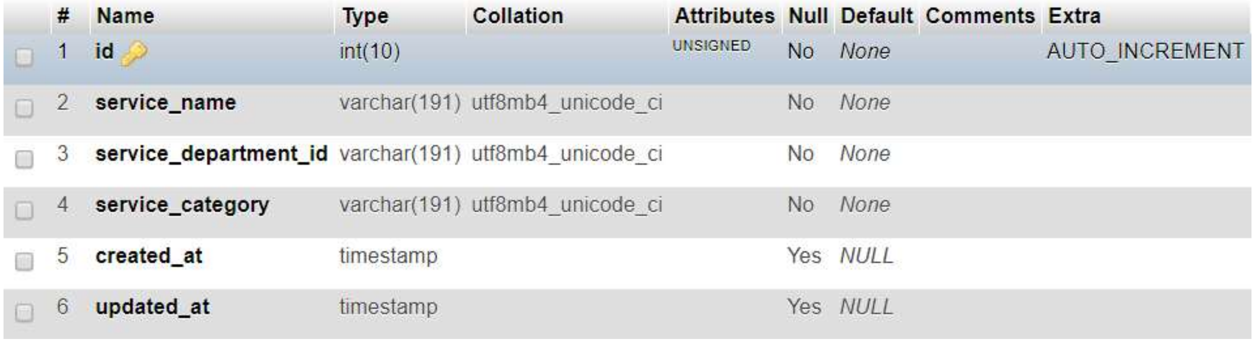
Departments Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
□	1	**id** 🔑	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
□	2	**department_name**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	3	**department_company_id**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	4	**department_status**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	5	**created_at**	timestamp			Yes	*NULL*		
□	6	**updated_at**	timestamp			Yes	*NULL*		

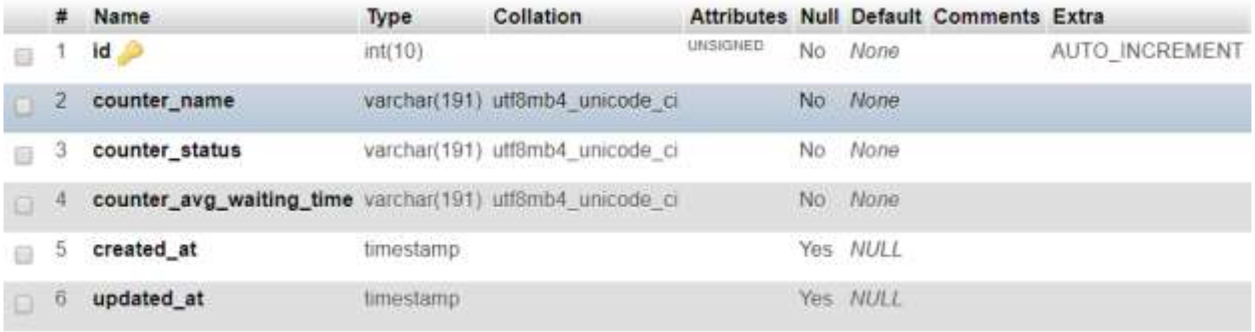
Figure 9 Department Table

Services Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
□	1	**id** 🔑	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
□	2	**service_name**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	3	**service_department_id**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	4	**service_category**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	5	**created_at**	timestamp			Yes	*NULL*		
□	6	**updated_at**	timestamp			Yes	*NULL*		

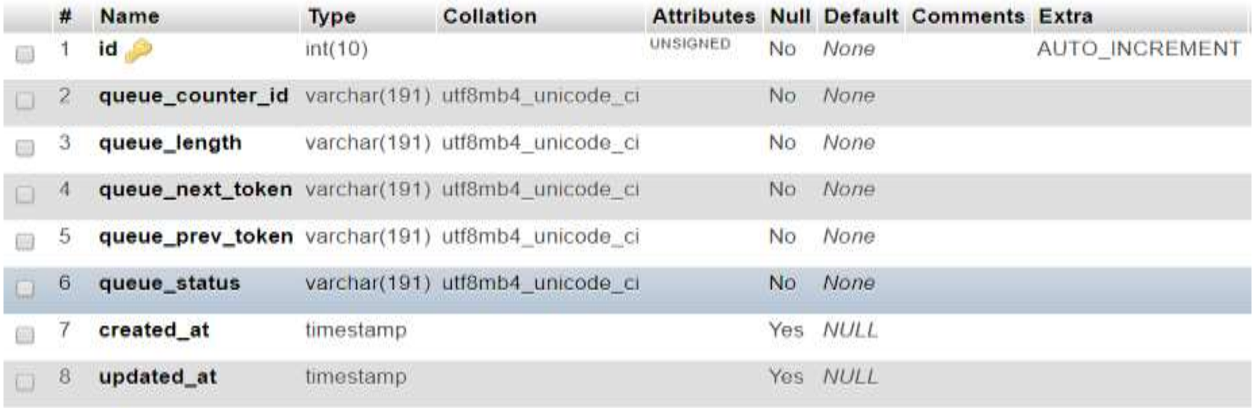
Counters Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
□	1	**id** 🔑	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
□	2	**counter_name**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	3	**counter_status**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	4	**counter_avg_waiting_time**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	5	**created_at**	timestamp			Yes	*NULL*		
□	6	**updated_at**	timestamp			Yes	*NULL*		

Figure 10 Counter Table

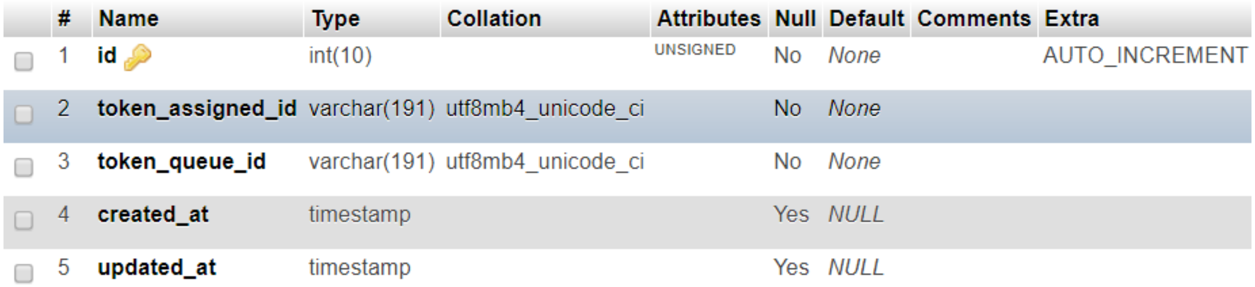
Queue Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
□	1	**id** 🔑	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
□	2	**queue_counter_id**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	3	**queue_length**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	4	**queue_next_token**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	5	**queue_prev_token**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	6	**queue_status**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
□	7	**created_at**	timestamp			Yes	*NULL*		
□	8	**updated_at**	timestamp			Yes	*NULL*		

Figure 11 Queue Table

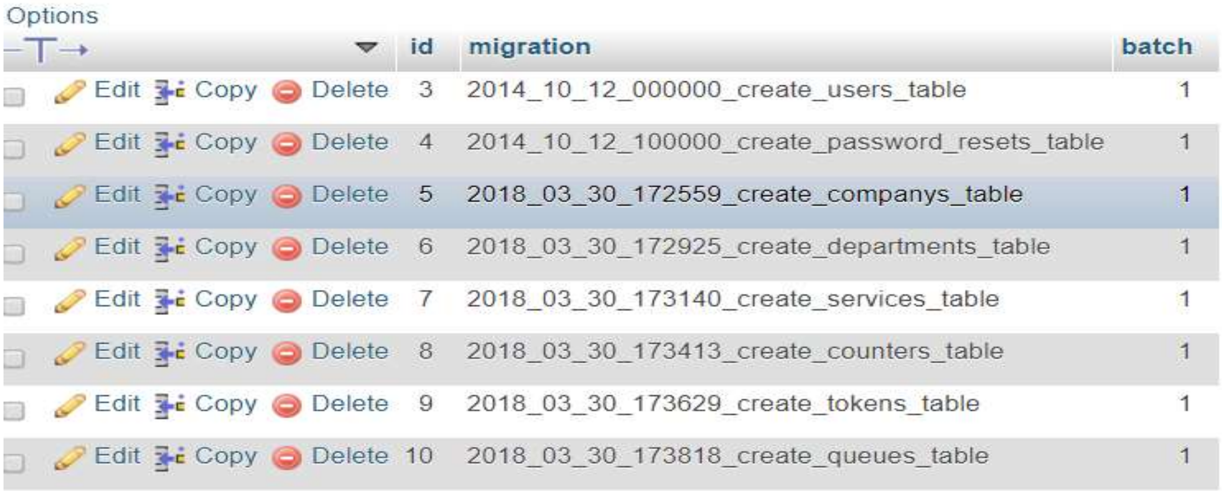
Tokens Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**id**	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
☐	2	**token_assigned_id**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	3	**token_queue_id**	varchar(191)	utf8mb4_unicode_ci		No	*None*		
☐	4	**created_at**	timestamp			Yes	*NULL*		
☐	5	**updated_at**	timestamp			Yes	*NULL*		

Figure 12 Tokens Table

Migrations Table



Options

		id	migration	batch
☐	Edit Copy Delete	3	2014_10_12_000000_create_users_table	1
☐	Edit Copy Delete	4	2014_10_12_100000_create_password_resets_table	1
☐	Edit Copy Delete	5	2018_03_30_172559_create_companys_table	1
☐	Edit Copy Delete	6	2018_03_30_172925_create_departments_table	1
☐	Edit Copy Delete	7	2018_03_30_173140_create_services_table	1
☐	Edit Copy Delete	8	2018_03_30_173413_create_counters_table	1
☐	Edit Copy Delete	9	2018_03_30_173629_create_tokens_table	1
☐	Edit Copy Delete	10	2018_03_30_173818_create_queues_table	1

Relationships

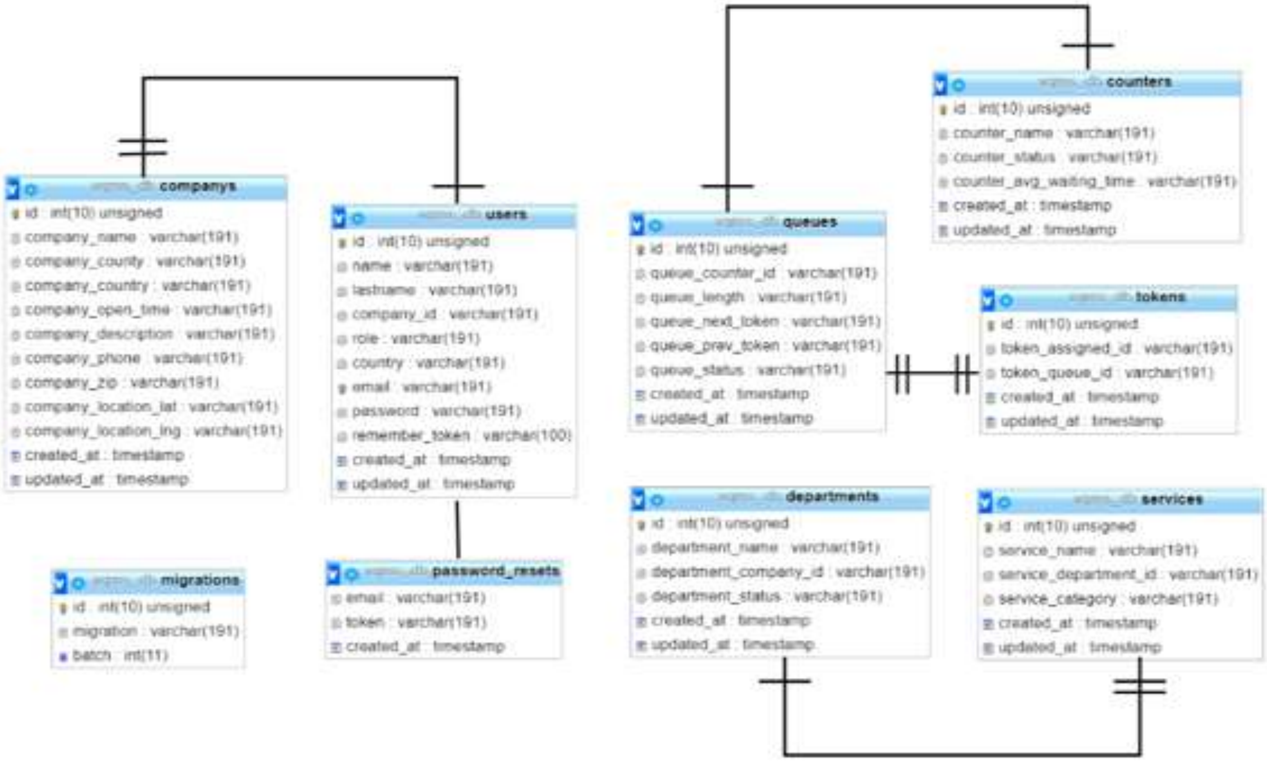

companys
id : int(10) unsigned
company_name : varchar(191)
company_county : varchar(191)
company_country : varchar(191)
company_open_time : varchar(191)
company_description : varchar(191)
company_phone : varchar(191)
company_zip : varchar(191)
company_location_lat : varchar(191)
company_location_lng : varchar(191)
created_at : timestamp
updated_at : timestamp
users
id : int(10) unsigned
name : varchar(191)
lastname : varchar(191)
company_id : varchar(191)
role : varchar(191)
country : varchar(191)
email : varchar(191)
password : varchar(191)
remember_token : varchar(100)
created_at : timestamp
updated_at : timestamp
migrations
id : int(10) unsigned
migration : varchar(191)
batch : int(11)
password_resets
email : varchar(191)
token : varchar(191)
created_at : timestamp
counters
id : int(10) unsigned
counter_name : varchar(191)
counter_status : varchar(191)
counter_avg_waiting_time : varchar(191)
created_at : timestamp
updated_at : timestamp
queues
id : int(10) unsigned
queue_counter_id : varchar(191)
queue_length : varchar(191)
queue_next_token : varchar(191)
queue_prev_token : varchar(191)
queue_status : varchar(191)
created_at : timestamp
updated_at : timestamp
tokens
id : int(10) unsigned
token_assigned_id : varchar(191)
token_queue_id : varchar(191)
created_at : timestamp
updated_at : timestamp
departments
id : int(10) unsigned
department_name : varchar(191)
department_company_id : varchar(191)
department_status : varchar(191)
created_at : timestamp
updated_at : timestamp
services
id : int(10) unsigned
service_name : varchar(191)
service_department_id : varchar(191)
service_category : varchar(191)
created_at : timestamp
updated_at : timestamp

Figure 13 Relationship Diagram

INPUT DESIGN

Login Page

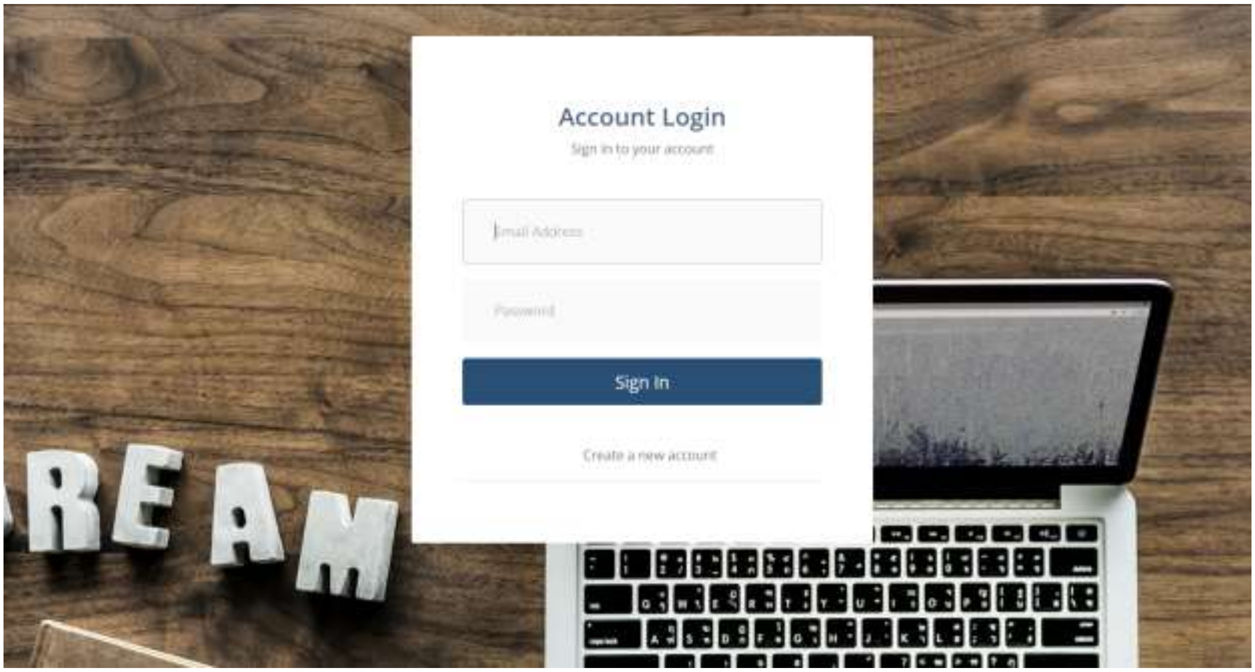
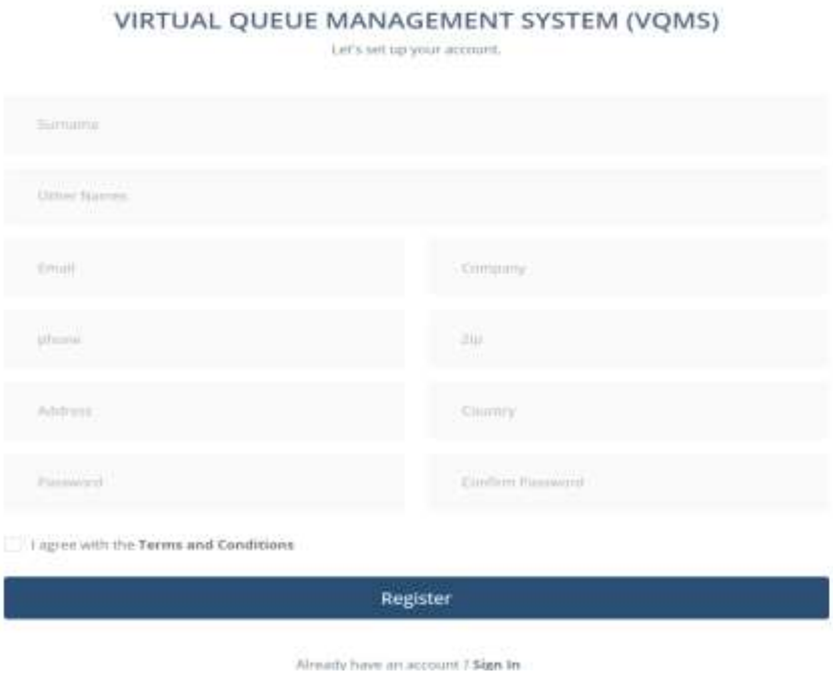


Figure 14 Login page

Register account page

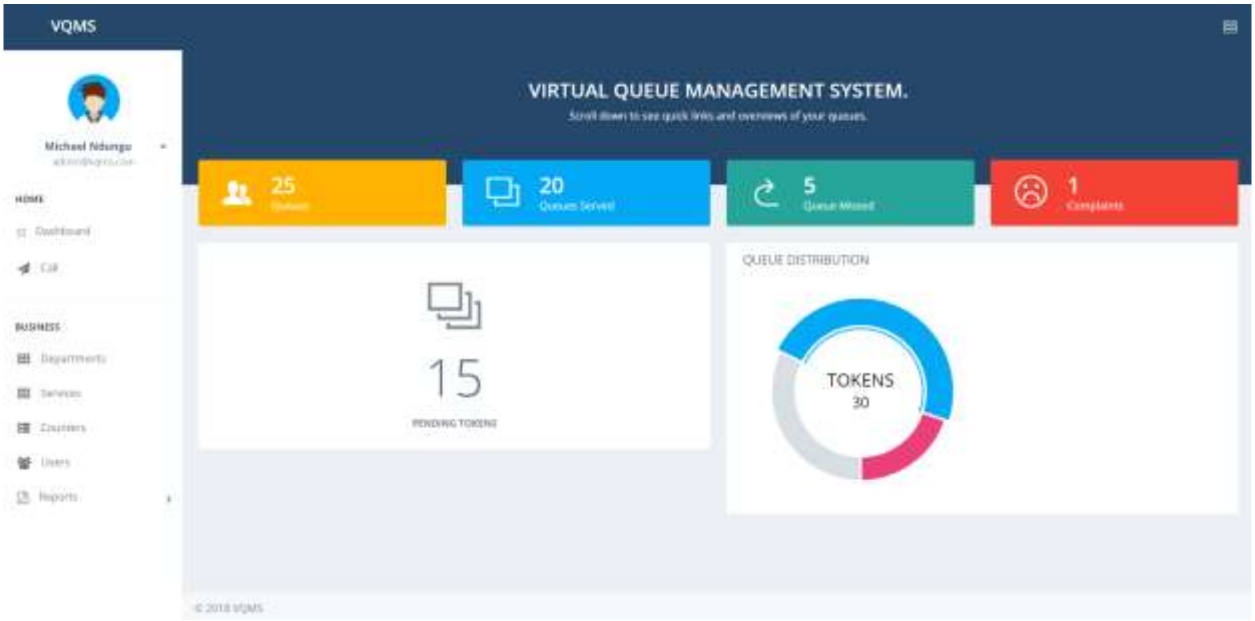


VIRTUAL QUEUE MANAGEMENT SYSTEM (VQMS)

Let's set up your account.

Figure 15 Register Page

Dashboard

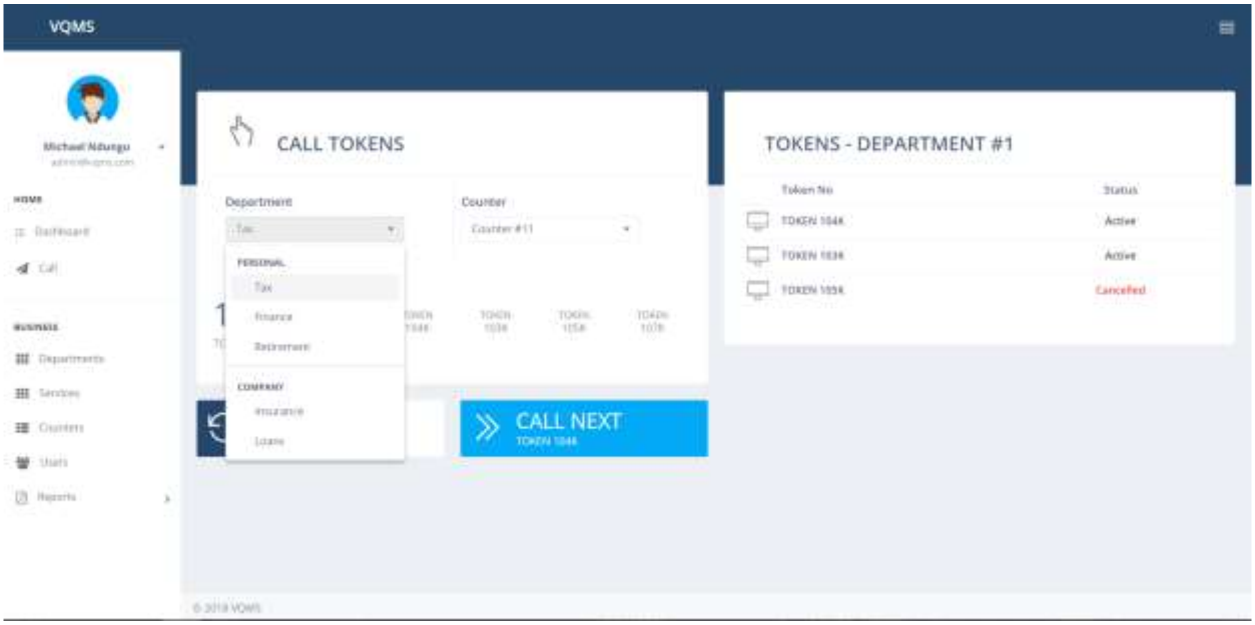


VIRTUAL QUEUE MANAGEMENT SYSTEM.

Scroll down to see quick links and overviews of your queues.

Figure 16 Dashboard

Call next page



CALL TOKENS

Department: Tax

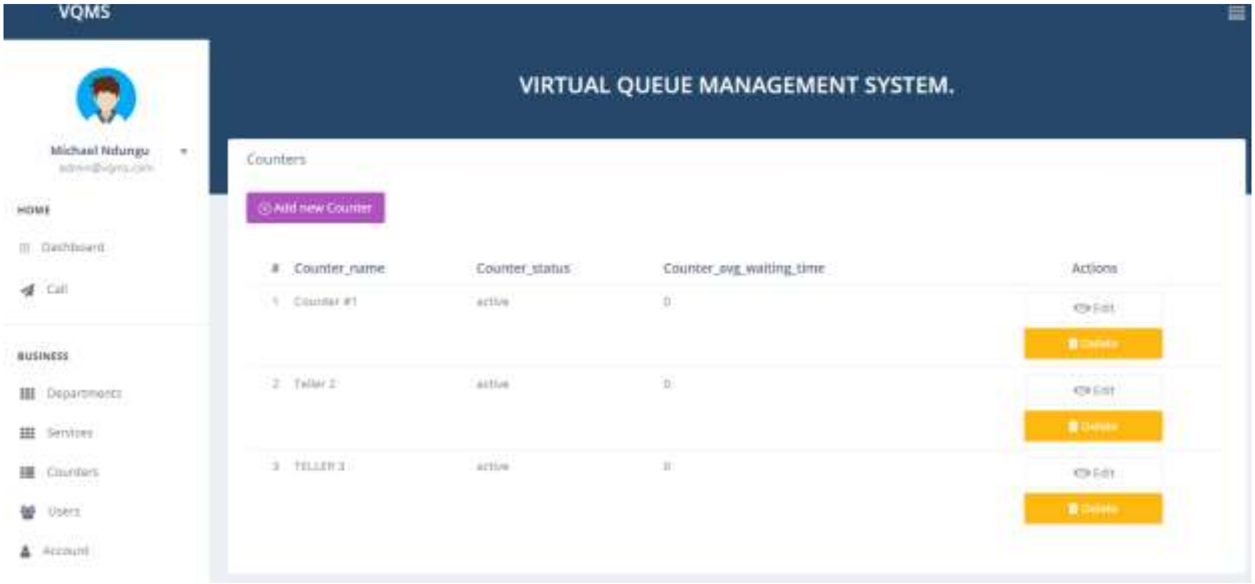
Counter: Counter #11

TOKENS - DEPARTMENT #1

Token No	Status
TOKEN 104K	Active
TOKEN 103K	Active
TOKEN 105K	Cancelled

Figure 17 Agent Call Center Page

Manage counters Page



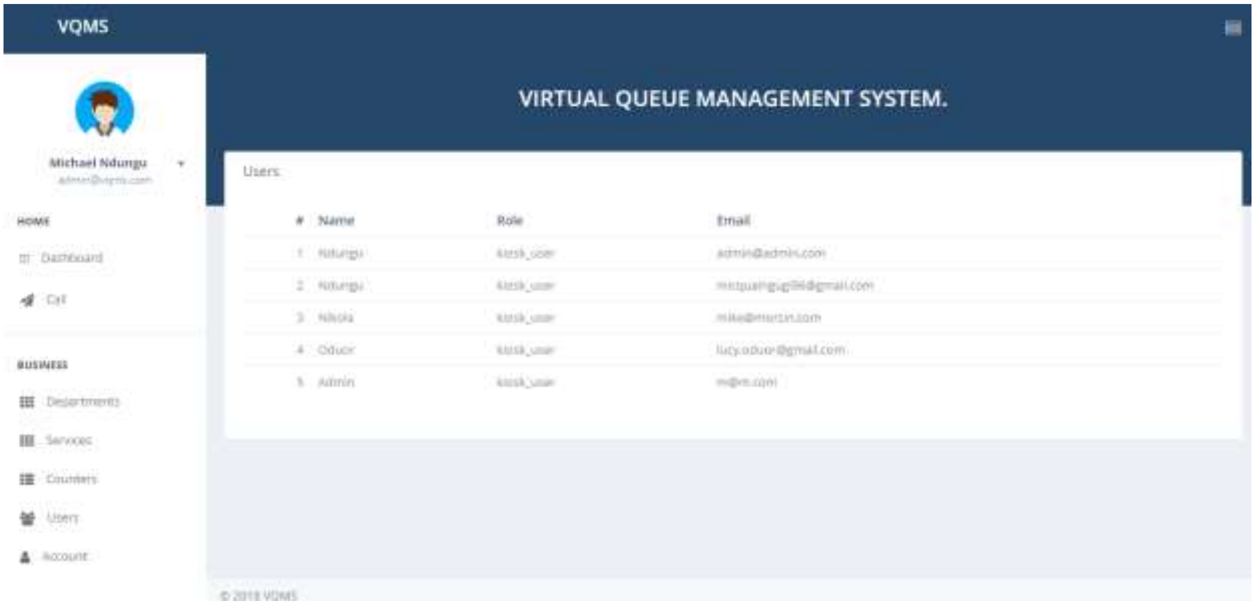
VIRTUAL QUEUE MANAGEMENT SYSTEM.

Counters

#	Counter_name	Counter_status	Counter_avg_waiting_time	Actions
1	Counter #1	active	0	Edit Delete
2	Teller 2	active	0	Edit Delete
3	TELLER 3	active	0	Edit Delete

Figure 18 Manage Counters

Manage Users Page



VIRTUAL QUEUE MANAGEMENT SYSTEM.

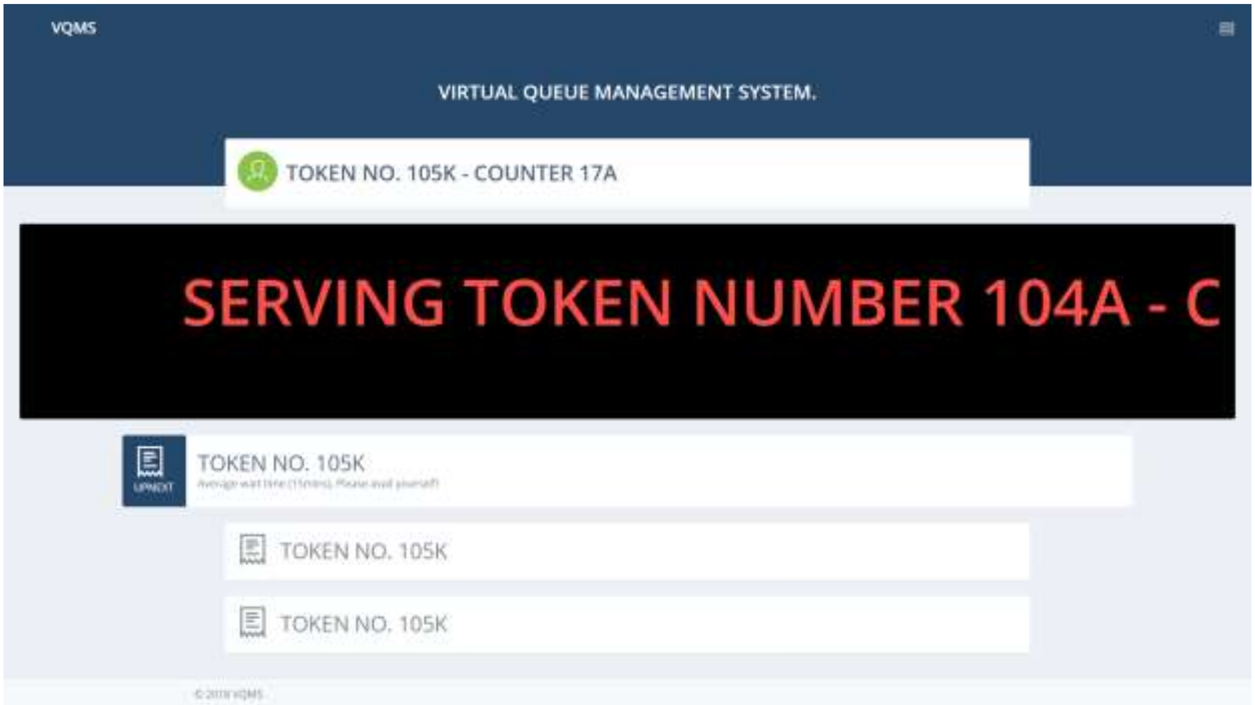
Users

#	Name	Role	Email
1	Ndungu	kiosk_user	admin@admin.com
2	Ndungu	kiosk_user	[illegible]@gmail.com
3	Nikola	kiosk_user	mike@martin.com
4	Oduor	kiosk_user	lucy.oduor@gmail.com
5	Admin	kiosk_user	m@m.com

© 2018 VQMS

Figure 19 Manage Users

TV screen or Monitor display



VIRTUAL QUEUE MANAGEMENT SYSTEM.

TOKEN NO. 105K - COUNTER 17A

SERVING TOKEN NUMBER 104A - C

TOKEN NO. 105K
Average wait time (15mins). Please avail yourself

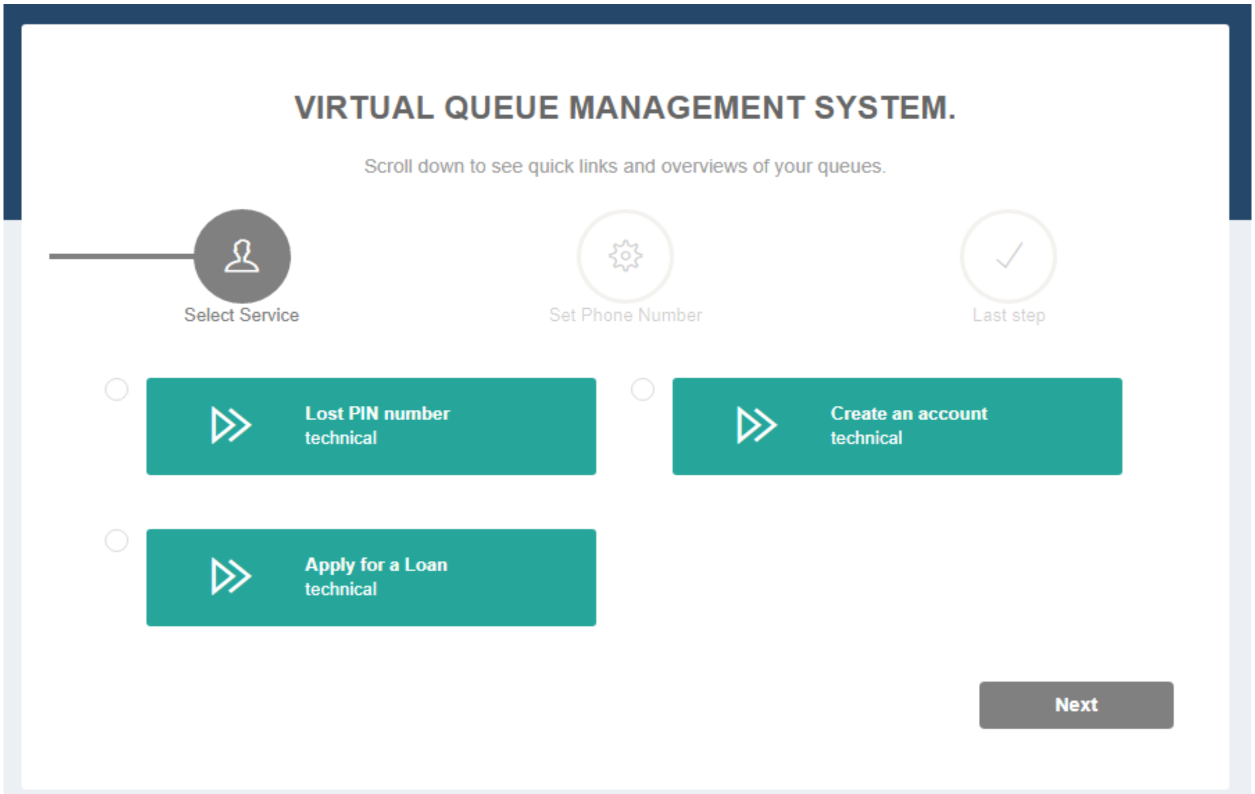
TOKEN NO. 105K

TOKEN NO. 105K

© 2018 VQMS

Figure 20 TV monitor page

Tablet mode display page



VIRTUAL QUEUE MANAGEMENT SYSTEM.

Scroll down to see quick links and overviews of your queues.

○ Lost PIN number
technical

○ Create an account
technical

○ Apply for a Loan
technical

Figure 21 Services page

Android Application Screens

Booking a token

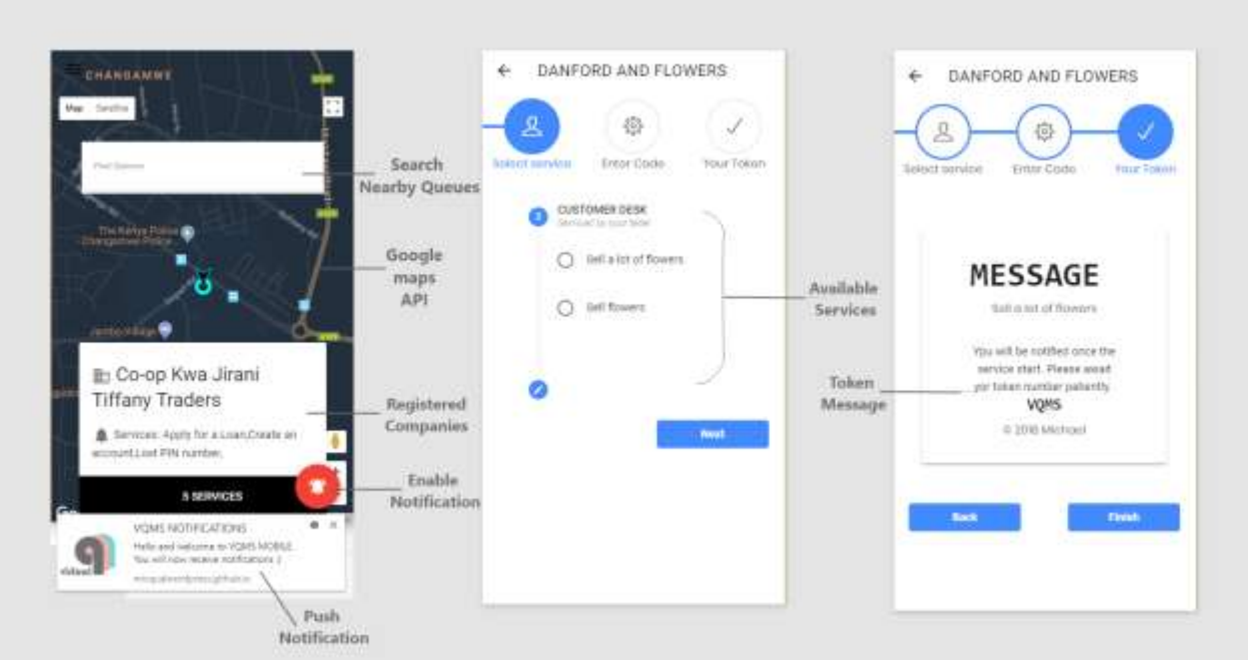


Figure 22 Mobile: booking a token part 1

Viewing Tokens

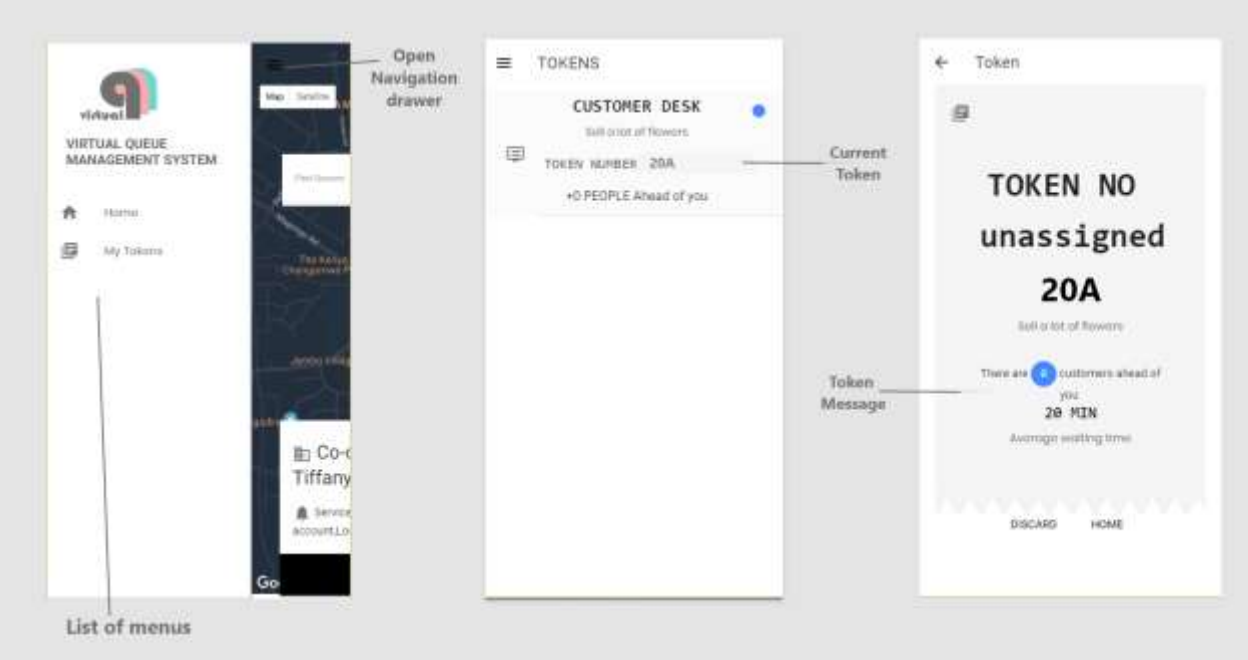
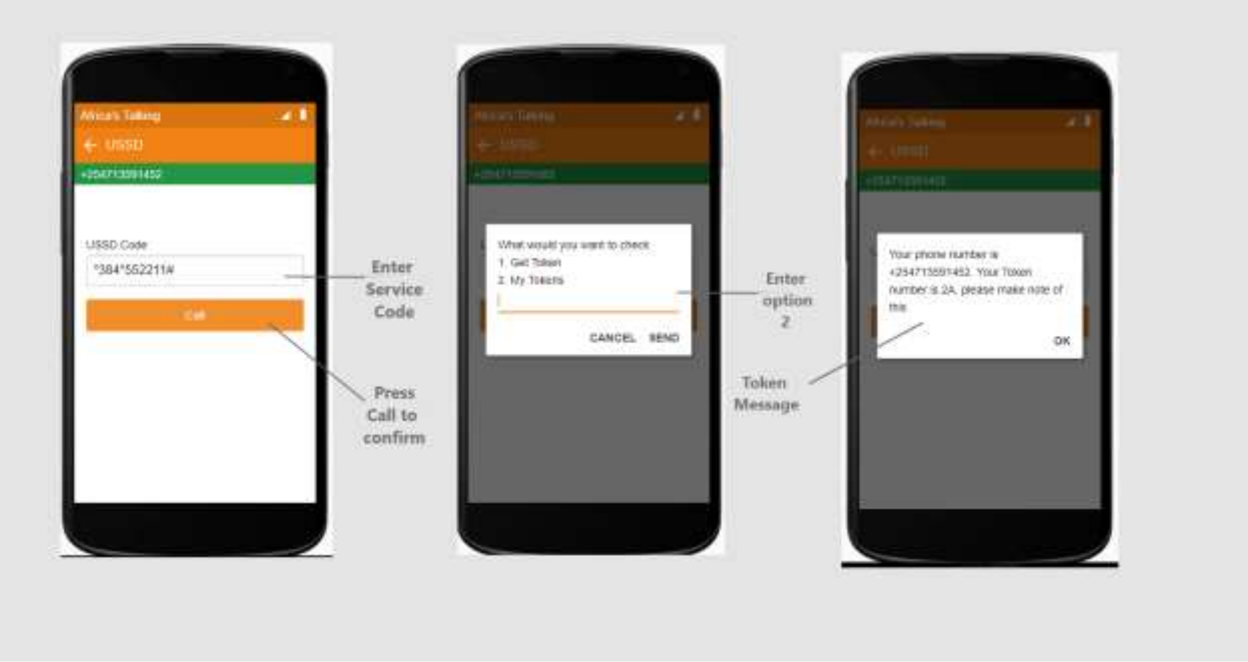


Figure 23 Mobile: booking a token part 2

Booking using USSD



What would you want to check
1. Get Token
2. My Tokens

Your phone number is +254713591452. Your Token number is 2A, please make note of this

Figure 24 Mobile: Booking using USSD

V. SYSTEM IMPLEMENTATION AND TESTING

INTRODUCTION

System testing involves checking each of the system modules to make sure that they are functioning properly. System implementation is done after successful system testing to incorporate the new system into an organization using any of the various changeover methods.

DATABASE TESTING

The database systems used in this project are MySQL by Oracle Technologies and firebase, a Realtime NoSQL database by Google. Database testing was done after system development to check whether the database was able to store the desired data as well as testing its integration with other components of the system. The developer used various categories of data to test the integrity of the database. First normal range data was used and the results were valid after processing. Extreme data was used to test whether the system could accommodate the extreme ranges. Further, exceptional data was used to test how the system would respond when subjected to invalid data.
Consider the following test data performed on the Users table. The two fields under test are Email and password.

FIELD NAME	DATA TYPE	TEST DATA	RESPONSE	COMMENT
Email	varchar	admin@admin.com	Accepted	Valid
Email	varchar	Admin.com	Rejected	Invalid
Email	Varchar	admin@admin.com (email already exists)	Rejected	Valid
Password	Varchar	Password01	Accepted	Valid
Password	Varchar	#pAssword (Confirmation Error)	Rejected	Invalid

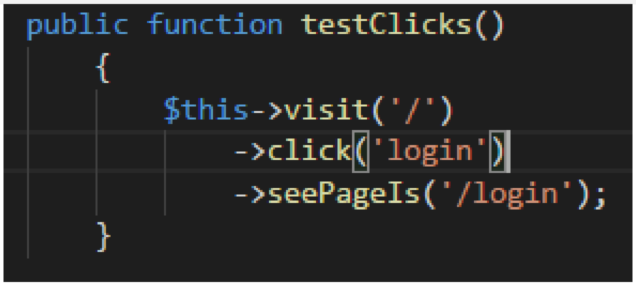
Table 1 Database testing

UNIT TESTING

Unit involves testing software with a small piece of source code [5]. VQMS is built on top of the Laravel php framework, thus has access to a custom TDD library. When performing tests, some assertions would be made, and the testing function would then assert if true or false.
Source code for unit testing were created by the developer as a part of software development. The following unit tests were performed to ascertain functionality.

TEST CASES (TC#)	TEST NAME	TEST DESCRIPTION	SOFTWARE	TEST ENVIRONMENT
TC1	Navigation Tests	This test verifies if the user is able to navigate the site and access all	VQMS	Windows Home Edition, 500GB HDD, 6GB RAM, Apache

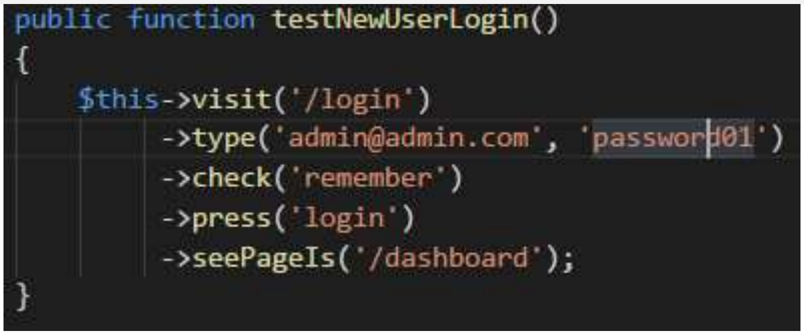
		URLs. Testing a login scenario		Server, PHP Engine, MySQL Server



```
public function testClicks()
    {
        $this->visit('/')
            ->click('login')
            ->seePageIs('/login');
    }
```

Figure 25 unit test navigation

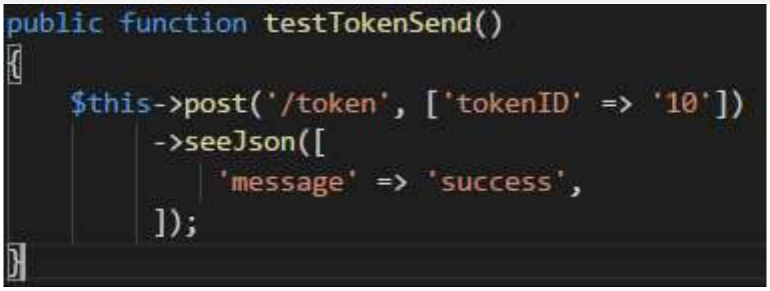
TC2	Authentication Tests	This test verifies the username and password to access VQMS	VQMS	Windows Home Edition, 500GB HDD, 6GB RAM, Apache Server, PHP Engine, MySQL Server



```
public function testNewUserLogin()
{
    $this->visit('/login')
        ->type('admin@admin.com', 'password01')
        ->check('remember')
        ->press('login')
        ->seePageIs('/dashboard');
}
```

Figure 26unit test authentication

TC3	Booking tests	This test verifies sending tokens to the VQMS API and receiving feedback	VQMS	Windows Home Edition, 500GB HDD, 6GB RAM, Apache Server, PHP Engine, MySQL Server



```
public function testTokenSend()
{
    $this->post('/token', ['tokenID' => '10'])
        ->seeJson([
            'message' => 'success',
        ]);
}
```

Figure 27 Unit test sending token

INTERFACE TESTING

Interface testing was performed to evaluate if all units of the interface transferred control to each other. To perform the interface tests, the developer created a checklist that outlined all the functional requirements of the system and the various test cases to assess them.

Functional requirement	Description
FR01	Authentication for users of the application
FR02	Selecting services for kiosk users
FR03	Message and notification to notify user of token number
FR04	Second Level authentication for admin users and agents
FR05	Call in the next token

Table 2Functional requirements

TEST CASES

The table below shows how each of the functional requirements were assessed using Test Cases.

TEST CASE 1 (TC1)

Table 3Test Case 1

TEST CASE (TC#)	FUNCTIONAL REQUIREMENT	TEST NAME	TEST DESCRIPTION	SOFTWARE	TEST ENVIRONMENT
TC1	FR1	Authentication	Verify and authenticate user using email and password	VQMS	Windows Home Edition, 500GB HDD, 6GB RAM, Apache Server, PHP Engine, MySQL Server

TEST CASE 2

TEST CASE (TC#)	FUNCTIONAL REQUIREMENT	TEST NAME	TEST DESCRIPTION	SOFTWARE	TEST ENVIRONMENT
TC3	FR3	Notification or SMS	Enter Phone number and receive a token using SMS or notification	VQMS	Windows Home Edition, 500GB HDD, 6GB RAM, Apache Server, PHP Engine, MySQL Server

Action Performed	Action's output	Valid action	Invalid action	Result
Press the Get Token Button in tablet mode display	Token received via SMS. On mobile, received via notification	Enter valid Phone or sign in using mobile device	Incorrect phone	Pass Test

TEST CASE 3

TEST CASE (TC#)	FUNCTIONAL REQUIREMENT	TEST NAME	TEST DESCRIPTION	SOFTWARE	TEST ENVIRONMENT
TC3	FR3	Notification or SMS	Enter Phone number and receive a token using SMS or notification	VQMS	Windows Home Edition, 500GB HDD, 6GB RAM, Apache Server, PHP Engine, MySQL Server

Action Performed	Action's output	Valid action	Invalid action	Result
Press the Get Token Button in tablet mode display	Token received via SMS. On mobile, received via notification	Enter valid Phone or sign in using mobile device	Incorrect phone	Pass Test

TEST CASE 4 (TC4)

TEST CASE (TC#)	FUNCTIONAL REQUIREMENT	TEST NAME	TEST DESCRIPTION	SOFTWARE	TEST ENVIRONMENT
TC4	FR4	High Level Authentication	Enter login details of agent or admin to access the system dashboard	VQMS	Windows Home Edition, 500GB HDD, 6GB RAM, Apache Server, PHP Engine, MySQL Server

Action Performed	Action's output	Valid action	Invalid action	Result
Click the Hamburger icon on the top right corner. Select login with rights	User is logged out, then redirected to sign in page	Enter login details of user with rights	Incorrect login details	Pass Test

TEST CASE 5 (TC5)

TEST CASE (TC#)	FUNCTIONAL REQUIREMENT	TEST NAME	TEST DESCRIPTION	SOFTWARE	TEST ENVIRONMENT
TC5	FR5	Queue Calling	Call in the next token	VQMS	Windows Home Edition, 500GB HDD, 6GB RAM,

	Apache Server, PHP Engine, MySQL Server

Action Performed	Action's output	Valid action	Invalid action	Result
Select department and counter, then call in next token Number.	Token received via SMS for user.	Select active counter and department	Failure to change status of counter	Pass Test

USABILITY TESTING

The table below summarized tests that were performed to ascertain the usability and experience of users while interacting with the system.

Table 4 Usability Testing

Element	**Output**
Flow from start to finish	Yes
Feedback from Actions performed	Instant Feedback
Tokens Received	Received
Seamless Navigation	Yes
Performance	Optimal
Failure or crashes	None
Runtime error messages	None
Slow or delayed loading	Acceptable

INTEGRATION TESTING

The purpose of this testing was to check whether the various modules of the system are well integrated and working harmoniously. All the form modules were well connected with the database and processing of data was successfully done. The reports generated outputs successfully as expected from the database and in the correct formats.

CONCLUSION

The need for faster and more efficient services in organization has increased exponentially over the last decade. More and more organizations are taking to the cloud to make their services easily accessible to their clients. An innovation to the current service model in organizations was greatly overdue. Armed with various developments tools and a working model at hand, the virtual queue management system was achieved. The system brings about quick and easy management of queues with very little cost and can be successfully implemented in medium crowd environment and thus help in the elimination of physical lines and waiting time all over the country in service-based institutions and organizations.

VI. PROJECT SUCCESS

A novel intelligent system for quick and effective management of queues using virtual tickets has been developed. The system allows customers to access virtual tokens online or via kiosks at service centers. They are then notified via SMS or notifications when it is their turn. All this is done in a centralized system that handles requests and dispatches responses to various distributed entities. The system lays out the base foundations for the existing ticket management systems, aiming that one day, after its continuous improvement, it becomes a full fledged product, battle tested, and ready to be deployed in schools or institutions looking for a better management of their services.

VII. FUTURE WORKS

In future, it would be interesting to implement the system with additional features that would make it work more efficiently. The following are some of the noted improvements that can be made on the system in future.

i. Integrate the system with physical hardware and token management Systems.

ii. Back-office usability testing.

iii. Possible inclusion of Web Sockets.

iv. Extend next-ticket metadata (based on acquired data, for data mining and trends matching).

v. Test in a production or simulated production environment.

vi. Deploy to production.

REFERENCES

[1] Kostecki M., "Waiting Lines as a Marketing Issue", European Management Journal, Vol. 14,

No.3, pp. 295-303, 1996

[2] "The importance of management information systems | Library Review | Vol 46, No 5", Emeraldinsight.com, 2017. [Online]. Available: http://www.emeraldinsight.com/doi/pdf/10.1108/00242539710178452. [Accessed: 14- Oct- 2017].

[3] "Explore Your Queuing System Options", LineLogic, 2017. [Online]. Available: http://blog.linelogic.com/blog/2011/11/17/explore-your-options/. [Accessed: 14- Oct- 2017].

[4] Dube L., Schmitt B.H., Leclerc F. "Intrusions into waiting lines: does the queue constitute a

social system?", Journal of Personality and Social Psychology, Vol. 63, Issue 5, pp. 806-815,199.

[5] "Testing", tutorialpoint.com, 2018. Available: www.tutorialspoint.com [Accessed: 14 April 2018]

[6] "Features if QLess", Qless.com, 2015. [online]. Available: www.qless.com [Accessed: 14 April 2018]

[7] "Wavetec Products Features", wavetec.com, 2015. [online]. Available: www.wacvetec.com [Accessed: 14 April 2018]