Applied Information Technology And Computer Science Vol. 4 No. 1 (2023) 916-934
© Universiti Tun Hussein Onn Malaysia Publisher's Office

AITCS

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/aitcs
e-ISSN :2773-5141

Development of Lasto Virtual Queue Management System

Priya Sri[1], **Noraini Ibrahim**[1]*,

[1]Faculty of Computer Science and Information Technology,
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author Designation

DOI: https://doi.org/10.30880/aitcs.2023.04.01.052
Received 14 June 2022; Accepted 26 October 2022; Available online 30 June 2023

Abstract: Lasto Virtual Queue Management System (VQMS) is a web-based system that will assist the Lasto Computer Service Centre. This system is designed to regulate the queues of customers efficiently and effectively. This approach is required because controlling queues has been a big challenge at Lasto Computer Service Centre as the organization tends to become quite busy, particularly during peak periods. Activities in the prototyping model are used to develop Lasto VQMS. Apart from that, the software process model chosen to build Lasto VQMS is Prototyping Software Process Model and written using PHP, HTML, CSS, JavaScript, the Bootstrap framework, and XAMPP server software. Whereas the database of the system will be developed using MySQL. This system is expected to efficiently manage customers and make their journey as pleasant and stress-free as possible by enhancing queueing processes by enabling customers to get their token virtually by scanning the QR code or by entering their email address. Additionally, the system will send an email notification when they are next in the line. The system passed 23 out of 23 test cases, equivalent to 100% of test cases. To conclude, this system is helpful in improving customer satisfaction and operational efficiency of the Lasto Computer Service Centre. Lasto VQMS can be improved in the future by implementing a number of recommendations, such as the option to cancel tokens and notify customers through SMS or WhatsApp when they arrive in place of email notifications as well as the expected waiting time after token booking.

Keywords: Virtual Queue Management System, web-based, QR code, Queue

1. Introduction

In an ordinary routine, queues are something that all individuals must deal with on a daily basis, as they must wait for long periods of time to be served in many public places, such as banks, hospitals, shops, pharmacies, and so on. As a necessary consequence, a Queue Management System (QMS) should be developed to manage and organize queue formations of people waiting in line in the most efficient and effective manner. This approach is required because controlling queues has been a major concern in recent years as organizations tend to get extremely crowded, particularly during peak periods

**Corresponding author: noraini@uthm.edu.my*

[1]. As customer satisfaction is one of the causes for concern during service provision, an effective way of managing queues is essential since long waiting times may lead to customer dissatisfaction [2]. As a part of this project, the Lasto Virtual Queue Management System (VQMS), which is a web-based system, is being developed for a computer service centre called the Lasto Computer Service Centre which is located in Sungai Petani, Kedah.

In brief, the Lasto Computer Service Centre provides various types of services such as hardware maintenance, data recovery, software installations, refresh, reformatting, or resetting windows, and purchasing items such as laptops, printers, and chargers. Generally, five to six tables serve as serving counters in this shop, with the staff on one side and the customers on the other. Each table will be allocated two chairs, which are one for the customer and one for the staff. The purpose of these counters is to serve customers who come to the shop with a specific problem, and so each counter will be in charge of one specific task. Customers that require more than a service are not served in a systematic manner. In short, it is crucial to establish Lasto VQMS for this shop because the number of customers visiting this shop will always be high, and there will always be a long line of customers waiting for their service because of a lack of space for customers to wait inside the shop due to its small size, sometimes causing them to queue outside the shop with their laptops and printers. After all, there is a lack of manpower, making it difficult for workers to manage the crowds, especially during peak hours. As a result, customers are likely to become dissatisfied, forcing them to seek service from other computer service providers.

Thus, there are three main objectives of this project. They are to design the Lasto VQMS using an object- oriented approach, develop a web based Lasto VQMS and perform functionality and user acceptance testing on the developed system. The users can be categorized into three groups, which are: the administrator who is the owner of Lasto Computer Service Centre, staff, and customers. The administrator can login into the system, manage the staff, services, counters and customers, and manage queue. The staff can also log into the system to manage the queue. Lastly, customers can book their token virtually to get services. Therefore, the development of this system is important since it provides customers with a variety of benefits in terms of ease while also assisting organizations in boosting customer satisfaction. Additionally, reducing the physical queue can make customers happier, and virtual queuing allows them to do anything they want with their time. The following papers will be organized as follows: Section 2 discusses the related work of this project. Section 3 focuses on methodology. Section 4 discusses the outcome and discussion of the system results. Finally, Section 5 discusses the project's conclusion.

2. Related Work

This section discusses all the related work collected to develop the system.

2.1 E-Queue System

E-Queue system is meant to organize queues in service sectors such as banks and post offices, where a high number of customers are expected on a daily basis [3]. E-Queue system has been developed and is being used in a variety of places to manage waiting customers, increase efficiency, and provide a significantly better system for them. This is because, as living standards rise, customers have started to seek services that will allow them to spend less time on them [4]. An electronic queue management system is a web-based system that ensures customers do not have to wait in the queue for long periods of time at various customer service areas. Basically, the E-Queue system will allow customers to check the queueing status at the premise and book their token without physically attending the premise. In summary, organisations can control their current capacity and only host customers whose turn has arrived.

2.2 Web-based Technology

Web-based technology is a technology that consists of client-server computing systems. These are comprised of two logical parts, which are a web server that provides services and a client that requests services from the server [5]. Systems using a client-server architecture are so popular in today's computing world because they are used almost every day for various applications [6]. Lasto VQMS employs this web technology because the demand for client-server-based applications has increased. Additionally, client server architectures are modular and adaptable, meaning they can be altered, expanded, and evolved in a variety of ways [7]. Besides that, all data is stored in a single location known as a server, which aids with data updates that are both effective and simple to implement.

2.3 Study of Existing Related Systems

The benefits and drawbacks of the proposed system compared to the present system are assessed using a variety of criteria. The comparison of the current system and the system that will be constructed is summarized in Table 1.

Table 1: Comparison between the existing systems and Lasto VQMS

Features/ System	QLess [8]	Wavetec Virtual Queue Management System [9]	Qminder [10]	Lasto VQMS
System Type	Web-Based, iOS, Android Application	iOS, Android Application	iOS Application	Web-based System
Registration	Available (Name, Phone number and password)	Not Available	Available (Name, email, password)	Available (Name, email, password)
Login	Available (Mobile number)	Not Available	Available (Email, Password)	Available (Email, Password)
Manage Services	Not Available	Available (WhatsApp)	Not Available	Available (System)
Manage Queue	Available (Call, Recall and No show	Available (Call, Recall and No show)	Available (Transfer, Release and No show)	Available (Call, Recall, Pending and Absent)
Generate Report	Available	Available	Available	Available
Notification feature	Available (SMS and voice call)	Available (Application)	Available(SMS)	Available(Email)
Technology used to book a queue	QR code	QR code	QR code	QR code

According to Table 1, all queue management systems have the basic functions of sending notification, manage queue, technology used and generate report. Furthermore, only QLess and

Qminder contains login and registration module like Lasto VQMS, whereas Wavetec will not. In short, the Lasto VQMS outperforms existing systems in terms of functionality.

3. Methodology

The software process model chosen to build Lasto VQMS is Prototyping Software Process Model. This project implements seven phases. Table 2 shows seven phases of prototyping model.

Table 2: The Phases of Prototyping Model

Phase	Task	Output
Planning	▪ Propose project ▪ Task scheduling. ▪ Identify problem, scope and objectives.	▪ Proposal ▪ Gantt Chart
Analysis	▪ Collect and analyze the information.	▪ Swimlane diagram (To-be-model) ▪ Use case diagram ▪ Activity diagram ▪ Sequence diagram ▪ Class diagram ▪ Requirement Definition
Design	▪ Design user interface of the system by using PHP ▪ Design database	▪ System architecture ▪ database schema and data dictionaries ▪ User interface
Prototype Implementation	▪ Programming ▪ Test the system and recorrect the bugs	▪ Program code ▪ Test case
Prototype 1	▪ Detect errors on system and repair the prototype 1	▪ Prototype 1
Prototype 2	▪ Detect errors on system and repair the prototype 2	▪ Prototype 2
System Implementation	▪ Develop a complete system.	▪ System ▪ Test cases

3.2 Analysis

System analysis describes the overall structure of a system where the requirements gathered, analysed and specified in form of modelling and text-based. The results of the analysis for this system are shown in the swimlane diagram, the use case diagram, the use case specification, the associated sequence diagram, the activity diagram, and the class diagram. In addition, to ensure that the tools used are appropriate for developing a system, all the hardware and software used should be thoroughly evaluated. The software and hardware requirements are the software and hardware that will be needed to develop the proposed system. The hardware utilizes is this project is an HP 15-bs0xx personal laptop with an Intel Core i5 processor. This laptop comes with 8GB RAM and a 1TB hard drive. In terms of software, Windows 10 is the operating system. Besides that, XAMPP is the server used by the new system's host and MySQL is used to control the database. The Visual Studio Code is used to manage web-based platforms including HTML, CSS, JavaScript and PHP. It's also necessary to have pre-installed web browsers like Mozilla Firefox, Google Chrome, or Microsoft Edge. The Gantts Project is a planning tool that is used to create a Gantt chart which helps to schedule the project timeline. In addition, Microsoft Word 2010 was used to handle all documentation, and Visual Paradigm was used to construct UML diagrams during the design process. Tables 3 and Table 4 shows the summarizations of hardware and software requirements.

Table 3: Hardware Requirement

Aspect		Requirement
Personal Laptop	-	HP laptop 15-bs0xx
Processor	-	Intel Core i5
Memory	-	8 GB
Storage of Hard Disk	-	1TB

Table 4: Software Requirement

Aspect		Requirement
Operating System	-	Windows 10
Server	-	XAMPP
Browser	-	Pre-installed web browser (Google Chrome,Microsoft Edge)
Code Editor	-	Visual Studio Code
Additional Software	-	Microsoft Word 2010, Gantts Project, VisualParadigm
Server Scripting Language	-	PHP
Additional Languages	-	HTML, CSS, JavaScript
Databases	-	MYSQL

3.2.1 Swimlane diagram (to be)

The workflow and interactions of the to-be Lasto VQMS were illustrated in the swimlane diagram depicted in Figure 1. This diagram primarily depicts how the process of obtaining service from the Lasto Computer Service Centre operates as a computerized process when Lasto VQSM is utilized. Aside from that, the diagram depicts the sequential activities of customers, system, staff, and administrator, as well as how the database eventually stores the information.

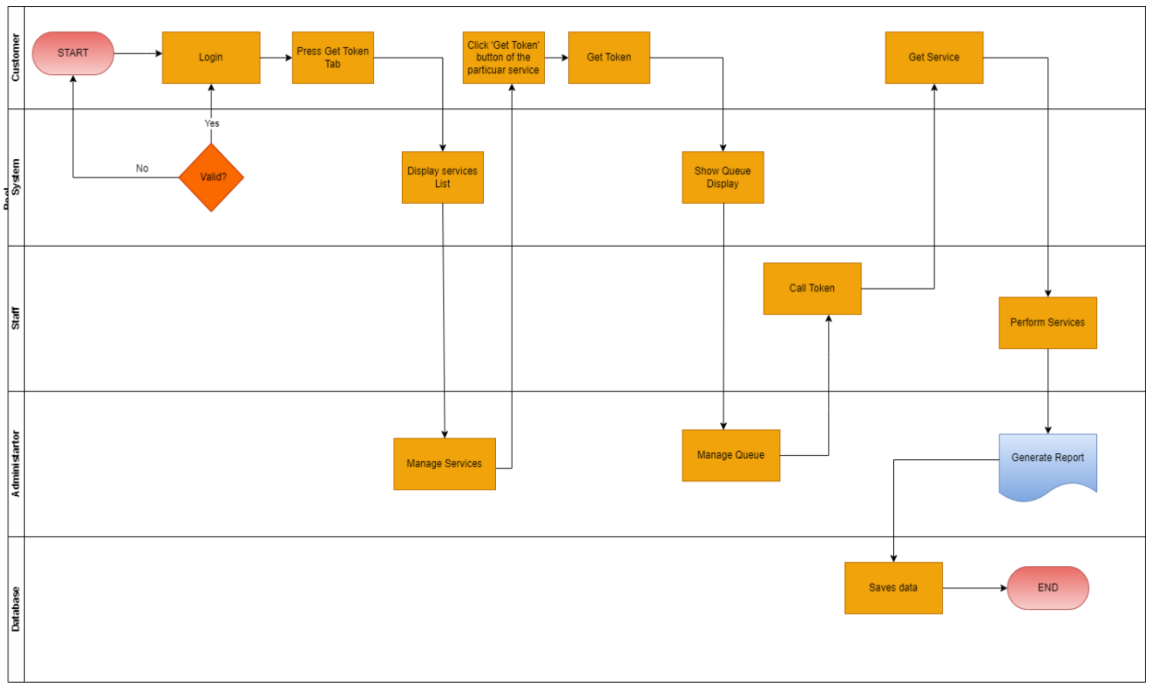


Figure 1: Swimlane Diagram (to-be) of Lasto VQMS

3.2.2 Use case diagram

The use case diagram is primarily used to describe the system user as well as the relationship between the system user and the use cases. It is used to illustrate the system's features. The system user, also known as an actor, will interact with the system while remaining outside the system boundary. Lasto VQMS is used by three actors which are administrators, staff, and customers. There are five main use cases in VQMS which are Register, Login, Manage Service, Manage Queue and Generate Report as shown in Figure 2.

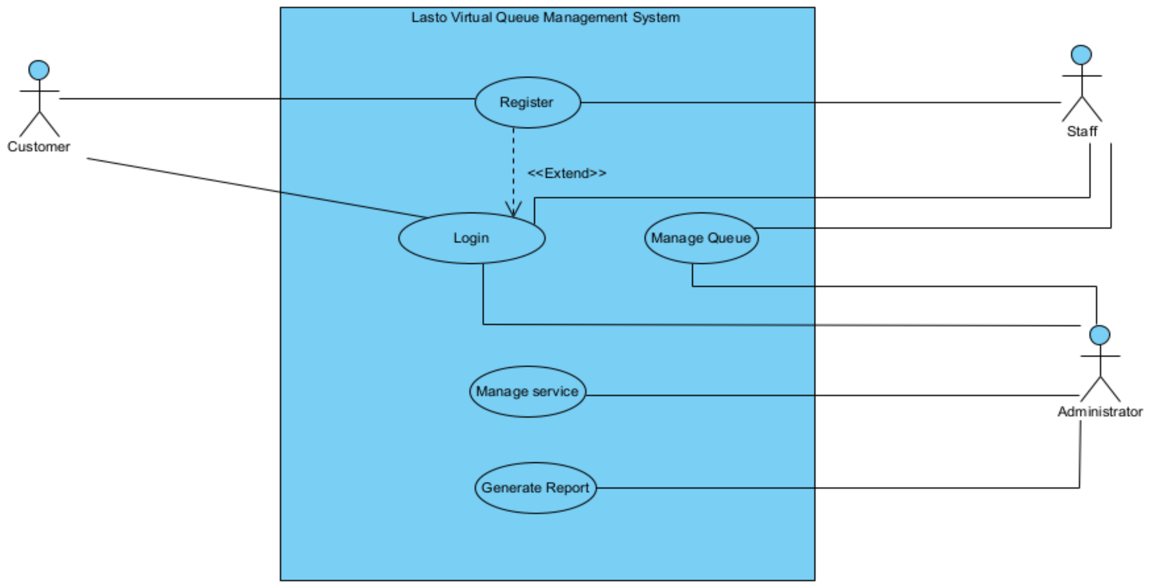


Figure 2: Use Case Diagram of Lasto VQMS

The register use case will basically manage the registration process of staffs and customers. Besides, the login use case will be used by all the users in order to login into the system. The third use which is manage service is to manage all the actions related to services and handled by the administrator. In addition, the manage queue use case is actually for staffs and administrator to handle the queue related actions and finally generate report use case will be related to report generating activities by administrator.

3.2.4 Class Diagram

The Lasto VQMS class diagram is shown in Figure 3. This diagram is a static structure that depicts the classes, attributes, methods, and relationships of the system. The classes identified include user, user_login, user_address, user_econtact, profile_picture, counter, service, booking, booking_history, feedback, qrcode, queue, and date_today.

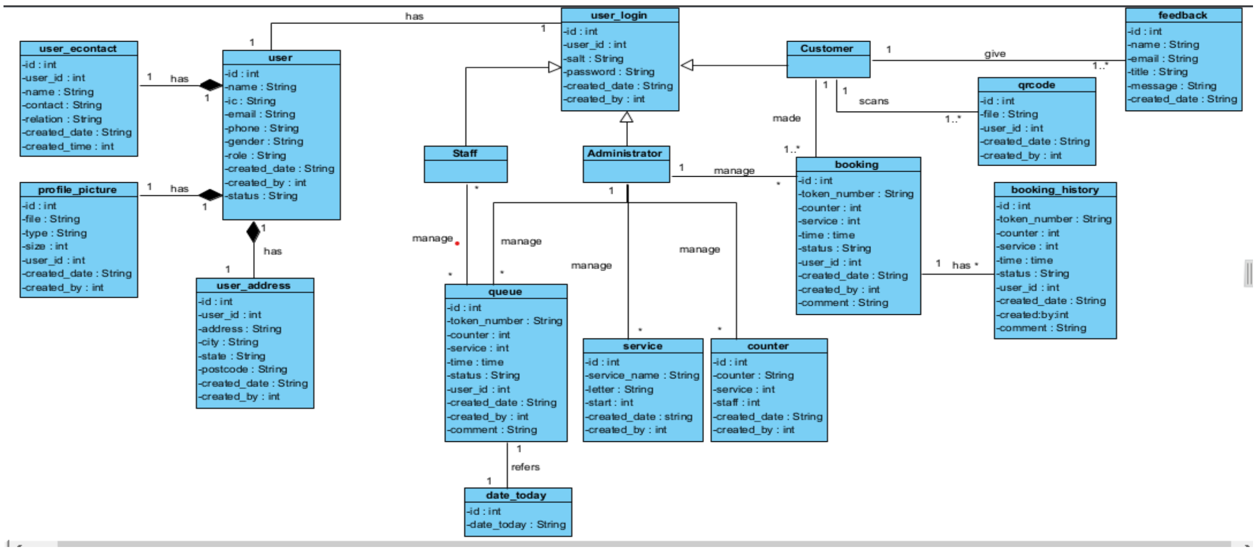


Figure 3: Class Diagram of Lasto VQMS

3.2.5 Requirement definition

The Requirement Definition contains all of the functional, non-functional, user, and system requirements for each module in Lasto VQMS. The system includes the registration module, login module, manage services module, manage queue module, and generate report module. The functional and non-functional requirements of Lasto VQMS are shown in Table 5 and Table 6.

Table 5: Functional Requirements

Use cases	Functional Requirements
Registration	- The system shall allow staffs and customers to register into the system using valid credentials such as name, email, password and confirm password. - The system shall be able to display registration success message for a valid registration. - The system shall be able to display error message if the email or password is already exist. - The system shall be able to display error message if invalid email or password is entered. - The system shall be able to display error message if any field is left blank. - The system shall be able to display error message if the password entered does not matches the confirm password.

Table 5: (cont)

Login	- The system should allow users to login into the system using registered email and password. - The system shall be able to display error message if invalid email or password is entered. - The system shall be able to display error message if any field is left blank. - The system shall not allow users to login without an active internet connection. - The system should redirect user to that respective main menu upon successful login.
Manage Services	- The system shall be able to display the list of services that has been added - The system shall allow the user to add mew services - The system shall allow the user to update the services - The system shall allow the user to delete any particular services - The system shall be able to display the search result - The system shall be able to display “No matching records found” if no services match the result - The system shall allow the user to use functions like PDF, Print, Excel, Copy and CSV to export the service records.
Manage Queue	- The system shall be able to allow the user to view the real-time queue - The system shall allow the user to call the token number. - The system shall allow the user to mark the token number as served. - The system shall allow the user to recall a token number. - The system shall allow the user to mark the token number as absent. - The system shall allow the user to mark the token number as pending if the service is not completely done. - The system shall allow the user to use functions like PDF, Print, Excel, Copy and CSV to export queue details. - The system shall be able to display the search result. - The system shall be able to display "No matching records found" message if none of the queue details matches the search result.
Generate Report	- The system shall allow the user to use functions like PDF, Print, Excel, Copy and CSV to export report. - The system shall be able to display "No matching records found" message if no records match the search result. - The system shall be able to display the search result.

Table 6: Non-Functional Requirements

Non-Functional Requirement	Description
Performance	- The database should be updated in real time.
Security	- Users can only access their own account with email id and password registered
Operational	- The system should be able to work on any web browser. - The system should be easily maintained and updated.
Cultural and Political	- The language of the system should be in English as it is easily understood by all kind of users

3.3 Design and Implementation

The design is represented in two subsections. They are the outcome of designing the system database and designing the interface with its implementation.

3.3.1 Database Design

For login, the customer, staff, and administrator are involved. However, for registration, only the customer and staff are involved. In order to obtain a token, the service and counter tables are used. The token booking information of the consumer is saved in the token table. Meanwhile, the information from the customer's feedback is stored in the feedback table. The relational schema of the Lasto VQMS is as follows:

i. `user(id,name,ic,email,phone,gender,role,created_date,created_by,status)`
ii. `user_login(id,user_id,salt,password,created_date,created_by)`
iii. `user_address(id,user_id,address,city,state,postcode,created_date, created_by)`
iv. `counter (id,counter,service,staff,created_date,created_by)`
v. `service (id,service_name, start, letter,created_date,created_by)`
vi. `booking(id,token_number,counter,service,time,status,user_id,created_date,created_by,comment)`
vii. `booking_history(id,token_number,counter,service,time,status, user_id,created_date,created_by,comment)`
viii. `feedback (id,name,email,title,message,created_date)`
ix. `qrcode(id,file,user_id,created_date,created_by)`
x. `queue(id_token_number,counter,service,time,status,user_id,created_date,created_by,comment)`
xi. `date_today(id,date_today)`
xii. `user_econtact(id,user_id,name,contact,realtion,created_date, created_by)`
xiii. `profile_picture(id,file,type,size,created_date,  created_by)`

3.3.2 Interface Design

This subsection will explain the interfaces design and implementation for Lasto VMQS. The customer and staff can register an account in Lasto VQMS by entering valid credentials such as name, email address and password as shown in Figure 4(a) and Figure 4(b).

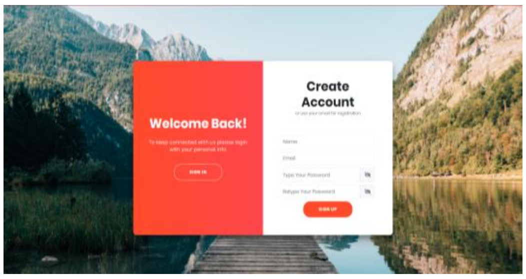


Figure 4(a): Customer Registration Interface

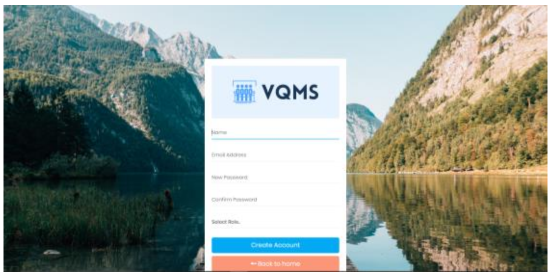


Figure 4(b): Staff Registration Interface

All users which is customer, staff and administrator can login into the system with registered email address and password as shown is Figure 5.

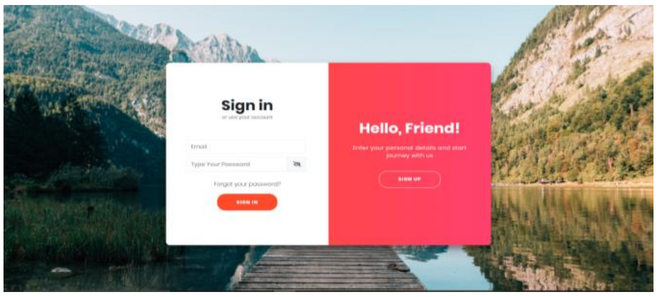


Figure 5: User Login Interface

The administrator can add, delete, update and view the services as shown in Figure 6(a) and Figure 6(b).

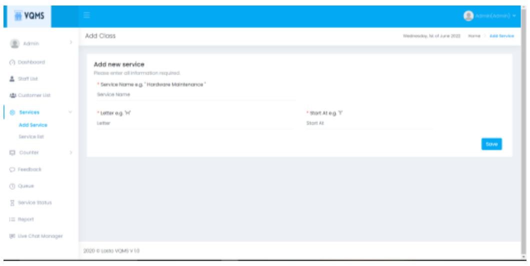
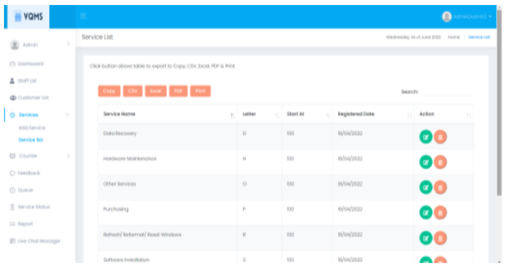


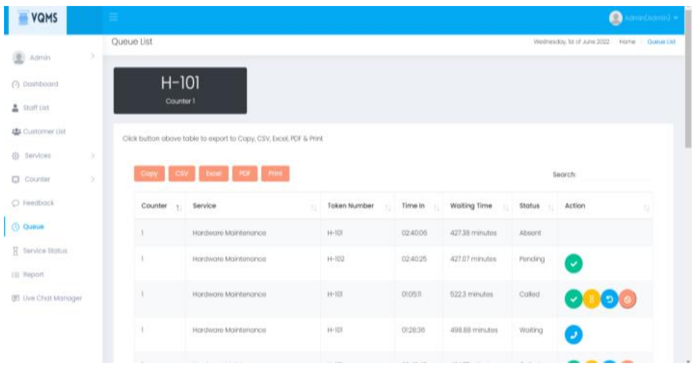
Figure 6(a): Add Services Interface



Service Name	Letter	Start At	Registered Date	Action
Data Recovery	D	100	10/04/2022	
Hardware Maintenance	H	100	10/04/2022	
Other Services	O	100	10/04/2022	
Purchasing	P	100	10/04/2022	
Refresh/ Reformat/ Reset Windows	R	100	10/04/2022	
Software Installation	S	100	10/04/2022	

Figure 6(b): Display Services List Interface

The system displays the real-time queue list as shown in Figure 7(a). Users can call the token number. Besides, users can also mark the token number as served, absent or can also recall if needed.



Queue List

Wednesday, 1st of June 2022 Home Queue List

H-101
Counter 1

Click button above table to export to Copy, CSV, Excel, PDF & Print

Counter	Service	Token Number	Time In	Waiting Time	Status	Action
1	Hardware Maintenance	H-101	02:40:06	427.38 minutes	Absent	
1	Hardware Maintenance	H-102	02:40:25	427.07 minutes	Pending	✓
1	Hardware Maintenance	H-101	01:05:11	522.3 minutes	Called	✓ ⏳ ↺ ⊘
1	Hardware Maintenance	H-101	01:28:36	498.88 minutes	Waiting	☎

Figure 7(a): Queue List Interface

The main function of this module is to enable the administrator and staffs to click the call icon to call the token number when customer turns arrive as shown in Figure 7(b). Then, the status will be updated from 'Waiting' to 'Called'.

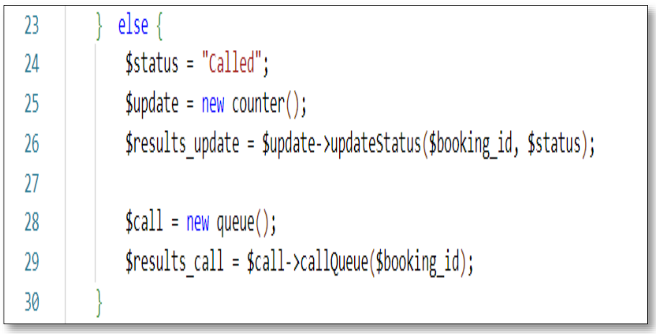
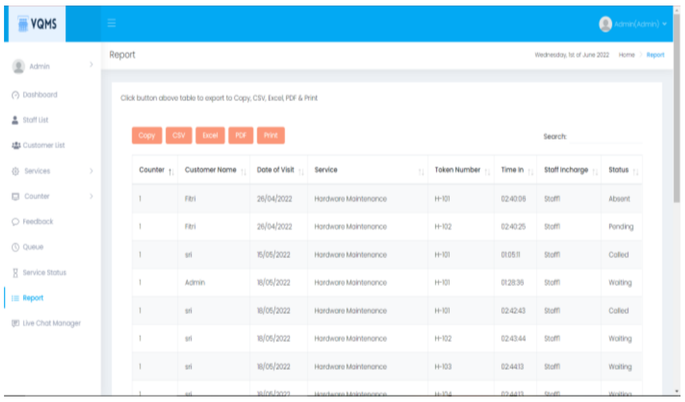

```
} else {
    $status = "Called";
    $update = new counter();
    $results_update = $update->updateStatus($booking_id, $status);

    $call = new queue();
    $results_call = $call->callQueue($booking_id);
}
```

Figure 7(b): Code Segment for Call Function

The administrator can generate overall report which consists of certain details of the customers who booked token as shown in Figure 8(a).



Report

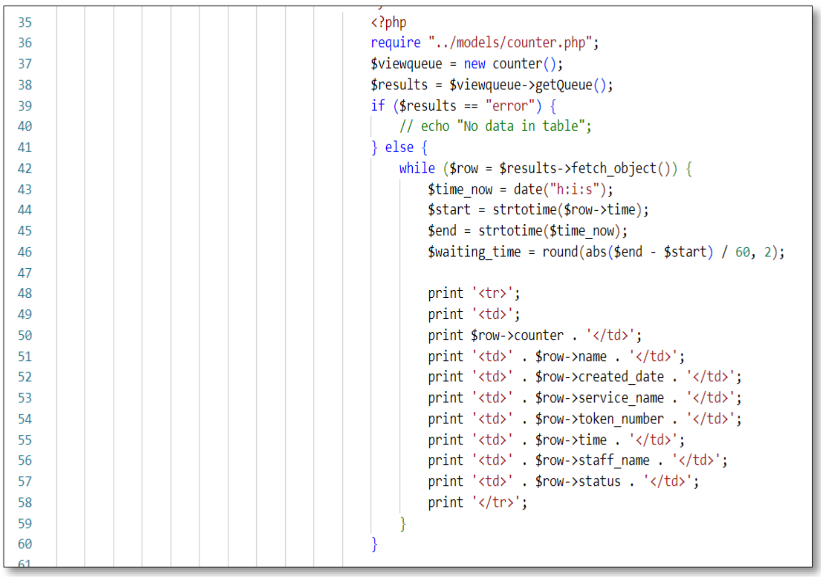
Wednesday, 1st of June 2022 Home Report

Click button above table to export to Copy, CSV, Excel, PDF & Print

Counter	Customer Name	Date of Visit	Service	Token Number	Time In	Staff Incharge	Status
1	Fitri	26/04/2022	Hardware Maintenance	H-101	02:40:06	Staff1	Absent
1	Fitri	26/04/2022	Hardware Maintenance	H-102	02:40:25	Staff1	Pending
1	sri	15/05/2022	Hardware Maintenance	H-101	01:05:11	Staff1	Called
1	Admin	18/05/2022	Hardware Maintenance	H-101	01:28:36	Staff1	Waiting
1	sri	18/05/2022	Hardware Maintenance	H-101	02:42:43	Staff1	Called
1	sri	18/05/2022	Hardware Maintenance	H-102	02:43:44	Staff1	Waiting
1	sri	18/05/2022	Hardware Maintenance	H-103	02:44:13	Staff1	Waiting

Figure 8(a): Generate Report Interface

The searched data will be fetched and will display in the report. If there is no such, data error message will be displayed. These function works as shown in the code segment in Figure 8(b).



```php
<?php
require "../models/counter.php";
$viewqueue = new counter();
$results = $viewqueue->getQueue();
if ($results == "error") {
    // echo "No data in table";
} else {
    while ($row = $results->fetch_object()) {
        $time_now = date("h:i:s");
        $start = strtotime($row->time);
        $end = strtotime($time_now);
        $waiting_time = round(abs($end - $start) / 60, 2);

        print '<tr>';
        print '<td>';
        print $row->counter . '</td>';
        print '<td>' . $row->name . '</td>';
        print '<td>' . $row->created_date . '</td>';
        print '<td>' . $row->service_name . '</td>';
        print '<td>' . $row->token_number . '</td>';
        print '<td>' . $row->time . '</td>';
        print '<td>' . $row->staff_name . '</td>';
        print '<td>' . $row->status . '</td>';
        print '</tr>';
    }
}
```

Figure 8(b): Generate Report Interface

The detail description of the system implementation is attached in **Appendix**.

4. Result and Discussion

The section will discuss the testing results of functional and user acceptance testing and observations made from the results.

4.1 Functional Testing

Functional testing is a type of software testing that is used to verify the functionality of a software system, as well as whether the function is working as expected. In functional testing, each function will be tested by providing an appropriate value, determining the output, and comparing the actual output to the expected value. Functional testing was performed manually by preparing test cases to ensure that the user interface, database, security, and functionality of Lasto VQMS were fully functional. There are 23 test cases in total used to test the Lasto VQMS. The test case ID for each use case is listed with the description in Table 7.

Table 7: Test Cases

No.	Test Case ID	Description
		TC_100 Login
1.	TC_100_01	Verify whether customers, staffs and administrator are able to login with a valid email and password.
2.	TC_100_02	Verify whether customers, staffs and administrator are able to login with any blank fields.
3.	TC_100_03	Verify whether customers, staffs and administrator are able to login with an invalid email or password.

Table 7: (cont)

No.	Test Case ID	Description
		TC_200 Register
1.	TC_200_01	Verify whether customer and staffs are able to register with a name, valid email, password, and confirm password
2.	TC_200_02	Verify whether customer and staffs are able to register with any blank fields.
3.	TC_200_03	Verify whether customer and staffs are able to register with invalid credentials.
4.	TC_200_04	Verify whether customer and staffs are able to register into the system with a registered email and password.
		TC_300 Manage Service
1.	TC_300_01	Verify whether administrator is able to add a new service to the system.
2.	TC_300_02	Verify whether administrator is able to add a new service with any blank fields.
3.	TC_300_03	Verify whether administrator is able to add a service that already existed.
4.	TC_300_04	Verify whether administrator is able to view the list of services.
5.	TC_300_05	Verify whether administrator is able to delete an existing service from the system.
6.	TC_300_06	Verify whether administrator is able to edit the added services from list.
7.	TC_300_07	Verify whether administrator is able to search service by any of its detail.
		TC_400 Manage Queue
1.	TC_400_01	Verify whether administrator and staffs are able to view the real-time queue.
2.	TC_400_02	Verify whether administrator and staffs are able to search queue details.
3.	TC_400_03	Verify whether administrator and staffs are able to call a token number in the queue.
4.	TC_400_04	Verify whether administrator and staffs are able to recall a token number in the queue.
5.	TC_400_05	Verify whether administrator and staffs can mark a customer as absent if they are not present when the token number is called.
6.	TC_400_06	Verify whether administrator and staffs are able to update a customer status as pending if their service has not been completed.
7.	TC_400_07	Verify whether administrator and staffs can mark a customer as served after serving the customer.
		TC_500 Generate Report
1.	TC_500_01	Verify whether administrator is able to search for a report.
2.	TC_500_02	Verify whether administrator is able to export the report using various functions such as Copy, CSV, Excel, PDF and Print.

The overall results of the testing are depicted in Table 8 and summarized in a pie chart as shown in Figure 9. To conclude, all of the function in the test case modules are functioning as intended, where there are no failed test cases for each modules. Therefore, the outcome of each test case modules is 100%, which gives the overall result of 100%.

Table 8: Overall Result Test Cases

Test Case Modules	Number of Test Cases	Total Passed Test Cases	Total Failed Test Cases
TC_100 Login	3	3 (100%)	0
TC_200 Register	4	4 (100%)	0
TC_300 Manage Service	7	7 (100%)	0
TC_400 Manage Queue	7	7 (100%)	0
TC_500 Report	2	2 (100%)	0

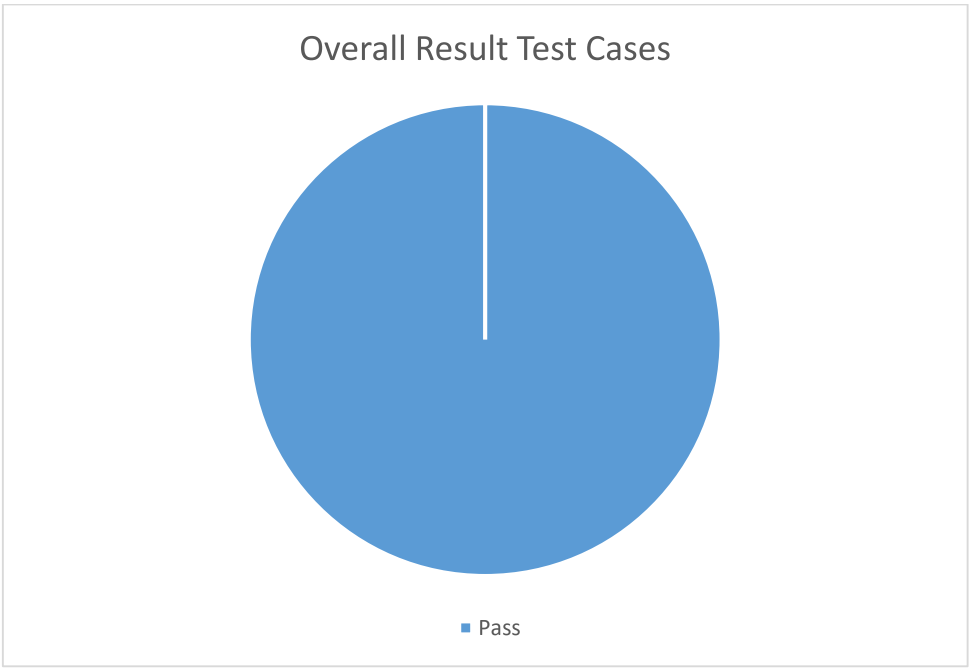


Figure 9: Pie chart for overall test cases result

4.2 User Acceptance Testing

Total of 10 respondents were to indicate their acceptance level for the statements made on five point rating scale. The questionnaire statements and the evaluation from respondent for each statement are presented in Table 9. In conclusion, the majority of respondents rate the system's functionality and user interface as 4 or 5. This demonstrates that the system is more likely to be accepted by users.

Table 9: User Acceptance Evaluation

No	Statement	Scale					Total
		1	2	3	4	5	
1	The homepage of Lasto VQMS website is very attractive.	0	0	1	2	7	10
2	The color scheme used throughout the website is appealing.	0	0	2	3	5	10
3	The interface design of the website is pleasant and user friendly.	0	0	0	5	5	10
4	The icons used in the website is easy to understand	0	0	0	4	6	10
5	The website allows the users to navigate to other pages easily.	0	0	0	4	6	10
6	The navigation is clear and concise.	0	0	1	3	6	10
7	The information provided in the website is sufficient.	0	0	0	4	6	10
8	The content in the website is relevant.	0	0	0	4	6	10
9	The services information provided in the Get Token online page is organized and clear.	0	0	0	3	7	10
10	The token booking process is efficient.	0	0	0	4	6	10
11	The Qr-Code method was adequate to perform the token booking task.	0	0	0	3	7	10
12	The website makes the queueing process easier.	0	0	0	2	8	10
13	It was easy to learn to use this website.	0	0	0	5	5	10
14	Overall, I'm very satisfied are with the website.	0	0	0	4	6	10

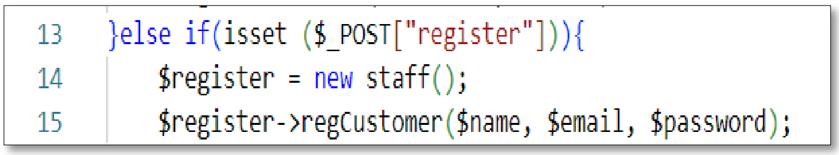
5. Conclusions

As a conclusion, developing Lasto VQMS will be one of the most effective ways to address the challenges that arise for employees and customers. Lasto VQMS will also be able to handle large crowds, which will improve customer service and employee efficiency. This method is also meant to help keep their company's reputation in good shape by making customers happier. This system was developed with many advantages, such as the ability for customers to access it at any time and from any location. Customers will have multiple options for receiving their tokens, and users will be able to monitor the real-time queue. Lasto VQMS has been developed to meet the objectives. However, the system still has some limitations, such as no voice features for calling the call token, as there are in real life, and no way to cancel a token. Aside from that, the token reserved does not indicate the expected serving time. Some of the recommendations that can be implemented are to use SMS or WhatsApp notifications instead of email to notify customers when they arrive. Furthermore, token cancellation features should be implemented. Moreover, once a customer books a token, the system should notify them of the estimated serving time. These recommendations are expected to improve Lasto VQMS in the future.

Acknowledgment

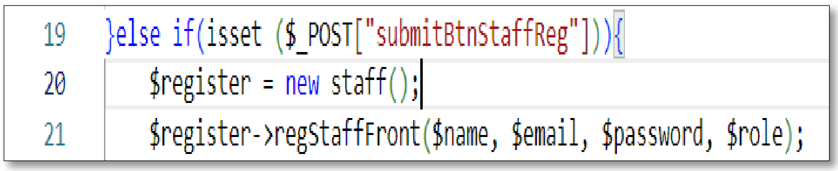
The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support.

Appendix



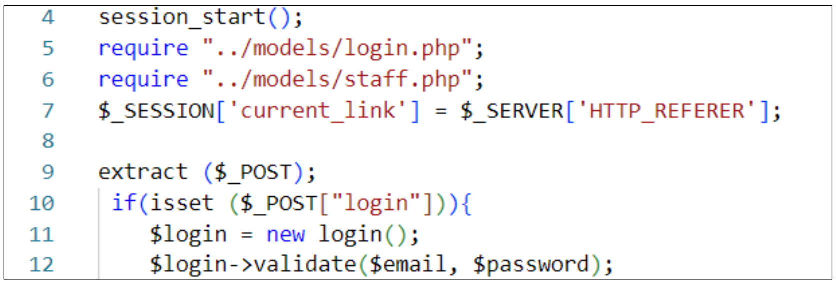
```
}else if(isset ($_POST["register"])){
    $register = new staff();
    $register->regCustomer($name, $email, $password);
```

Code Segment of Customer Registration



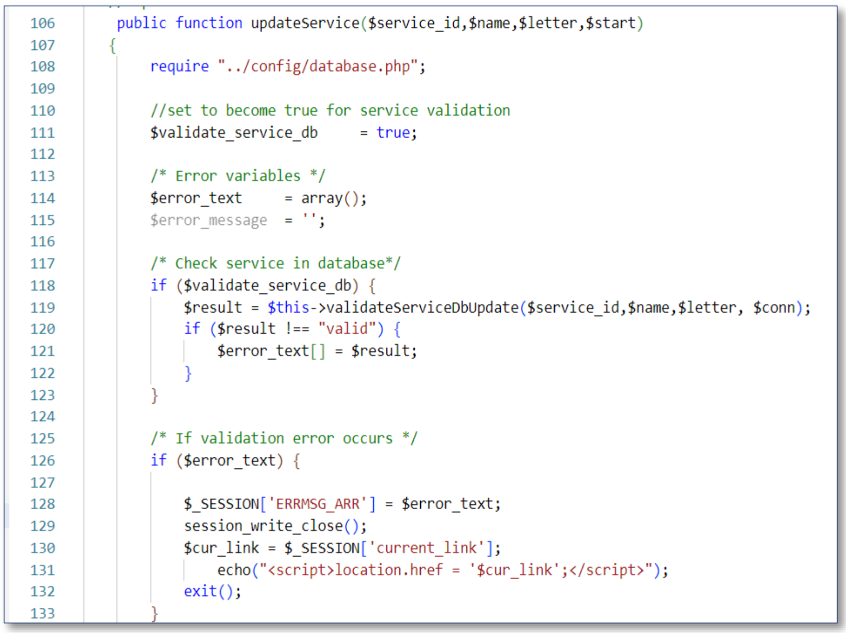
```
}else if(isset ($_POST["submitBtnStaffReg"])){
    $register = new staff();
    $register->regStaffFront($name, $email, $password, $role);
```

Code Segment of Staff Registration



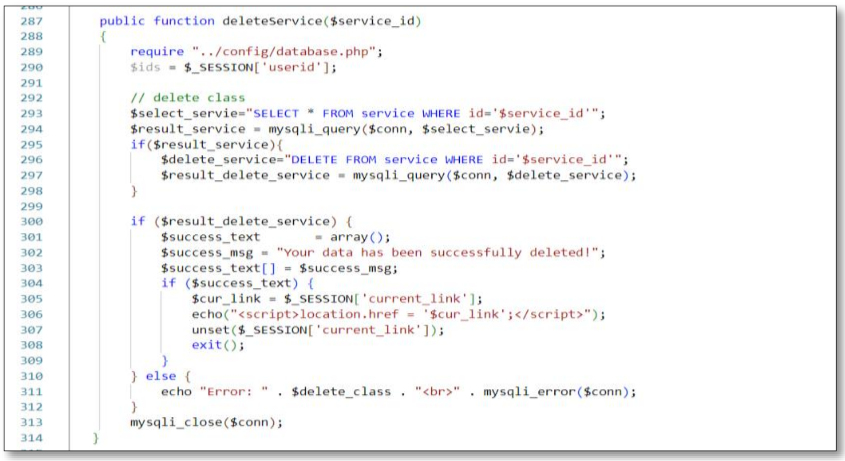
```
session_start();
require "../models/login.php";
require "../models/staff.php";
$_SESSION['current_link'] = $_SERVER['HTTP_REFERER'];

extract ($_POST);
 if(isset ($_POST["login"])){
    $login = new login();
    $login->validate($email, $password);
```

Code Segment to Validate User Login Details



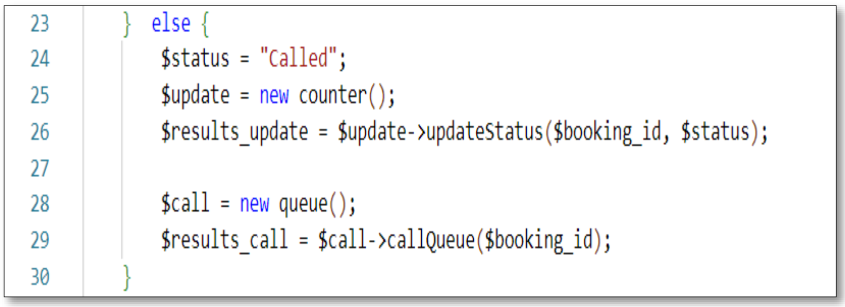
```
 public function updateService($service_id,$name,$letter,$start)
{
     require "../config/database.php";

     //set to become true for service validation
     $validate_service_db      = true;

     /* Error variables */
     $error_text      = array();
     $error_message  = '';

     /* Check service in database*/
     if ($validate_service_db) {
         $result = $this->validateServiceDbUpdate($service_id,$name,$letter, $conn);
         if ($result !== "valid") {
             $error_text[] = $result;
         }
     }

     /* If validation error occurs */
     if ($error_text) {

         $_SESSION['ERRMSG_ARR'] = $error_text;
         session_write_close();
         $cur_link = $_SESSION['current_link'];
             echo("<script>location.href = '$cur_link';</script>");
         exit();
     }
```

Code Segment for update service function



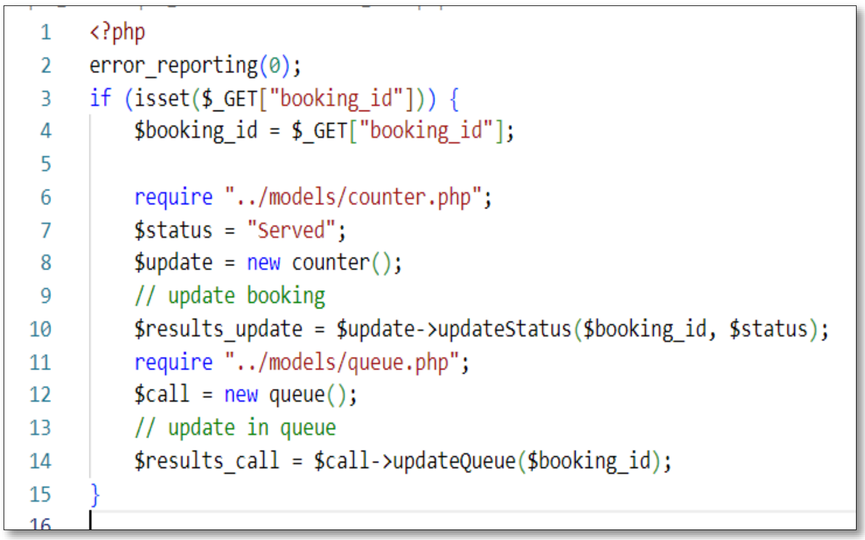
```
    public function deleteService($service_id)
    {
        require "../config/database.php";
        $ids = $_SESSION['userid'];

        // delete class
        $select_servie="SELECT * FROM service WHERE id='$service_id'";
        $result_service = mysqli_query($conn, $select_servie);
        if($result_service){
            $delete_service="DELETE FROM service WHERE id='$service_id'";
            $result_delete_service = mysqli_query($conn, $delete_service);
        }

        if ($result_delete_service) {
            $success_text         = array();
            $success_msg = "Your data has been successfully deleted!";
            $success_text[] = $success_msg;
            if ($success_text) {
                $cur_link = $_SESSION['current_link'];
                echo("<script>location.href = '$cur_link';</script>");
                unset($_SESSION['current_link']);
                exit();
            }
        } else {
            echo "Error: " . $delete_class . "<br>" . mysqli_error($conn);
        }
        mysqli_close($conn);
    }
```

Code Segment for delete service function



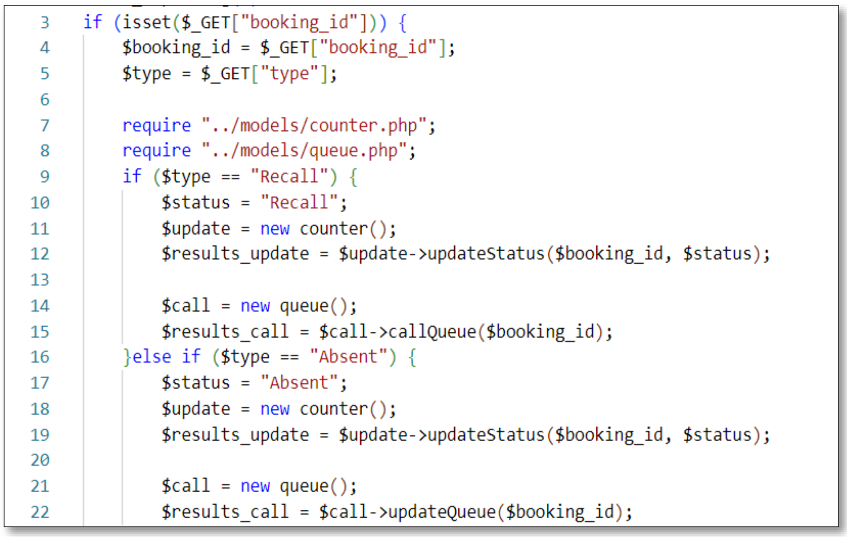
```
    } else {
        $status = "Called";
        $update = new counter();
        $results_update = $update->updateStatus($booking_id, $status);

        $call = new queue();
        $results_call = $call->callQueue($booking_id);
    }
```

Code Segment for Call Function



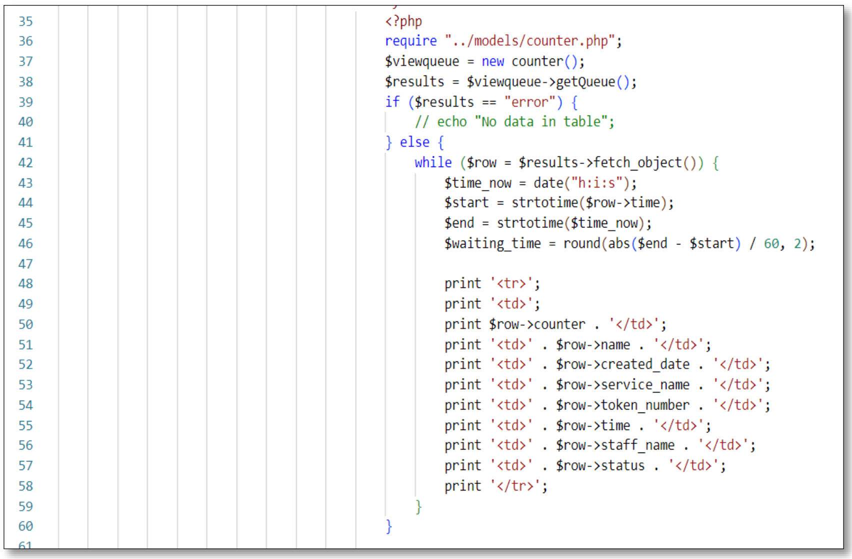
```
<?php
error_reporting(0);
if (isset($_GET["booking_id"])) {
    $booking_id = $_GET["booking_id"];

    require "../models/counter.php";
    $status = "Served";
    $update = new counter();
    // update booking
    $results_update = $update->updateStatus($booking_id, $status);
    require "../models/queue.php";
    $call = new queue();
    // update in queue
    $results_call = $call->updateQueue($booking_id);
}
```

Code Segment for Serve Function



```php
if (isset($_GET["booking_id"])) {
    $booking_id = $_GET["booking_id"];
    $type = $_GET["type"];

    require "../models/counter.php";
    require "../models/queue.php";
    if ($type == "Recall") {
        $status = "Recall";
        $update = new counter();
        $results_update = $update->updateStatus($booking_id, $status);

        $call = new queue();
        $results_call = $call->callQueue($booking_id);
    }else if ($type == "Absent") {
        $status = "Absent";
        $update = new counter();
        $results_update = $update->updateStatus($booking_id, $status);

        $call = new queue();
        $results_call = $call->updateQueue($booking_id);
```

Code Segment for Recall and Absent Function



```php
<?php
require "../models/counter.php";
$viewqueue = new counter();
$results = $viewqueue->getQueue();
if ($results == "error") {
    // echo "No data in table";
} else {
    while ($row = $results->fetch_object()) {
        $time_now = date("h:i:s");
        $start = strtotime($row->time);
        $end = strtotime($time_now);
        $waiting_time = round(abs($end - $start) / 60, 2);

        print '<tr>';
        print '<td>';
        print $row->counter . '</td>';
        print '<td>' . $row->name . '</td>';
        print '<td>' . $row->created_date . '</td>';
        print '<td>' . $row->service_name . '</td>';
        print '<td>' . $row->token_number . '</td>';
        print '<td>' . $row->time . '</td>';
        print '<td>' . $row->staff_name . '</td>';
        print '<td>' . $row->status . '</td>';
        print '</tr>';
    }
}
```

Code Segment for Reporting Module

References

[1] K. Ramasamy and F.F Chua, "Queue management optimization with short message system (SMS) notification". International Conference on Economics, Business Innovation IPEDR. Singapore, 2012.

[2] X. X. Zhao, "Queueing theory with the bank management innovation", Modern finance, vol. 3, pp.9-10, 2007. [Online]. Available: https://www.researchgate.net/publication/326328214 [Accessed Dec. 18, 2021].

[3] A. Z. Jidin, N. Mohd Yusof, and T. Sutikno, "Arduino based Paperless Queue Management System," TELKOMNIKA (Telecommunication Computing Electronics and Control), vol. 14, no. 3, 2016. [Online]. Available: http://telkomnika.uad.ac.id/index.php/TELKOMNIKA/article/view/3114 [Accessed Dec. 20, 2021].

[4] M. M. Davis and J. Heineke, "Understanding the roles of the customer and the operation for Better Queue management," International Journal of Operations & Production Management, vol. 14, no. 5, pp. 21–34, 1994, doi: 10.1108/01443579410056777.

[5] S. M. Lewandowski, "Frameworks for component-based client/server computing," ACM Computing Surveys, vol. 30, no. 1, pp. 3–27, 1998, doi: 10.1145/274440.274441.

[6] H.S. Oluwatosin, "Client-server model," IOSR Journal of Computer Engineering (IOSR-JCE) , vol. 16, January 2014. [Online]. Available: https://www.researchgate.net/profile/Shakirat-Sulyman/publication/271295146. [Accessed Dec. 29 2021].

[7] G. Schussel, "Client/server: Past, present and future." 1995. [Online]. Available: http://ciains.info/elearning/Solutions/Architecture/ClientServer/CS-past,presentFurure.pdf. [Accessed Dec 29 2021].