APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 6 No. 1 (2025) 1782-1799
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Web-Based UTHM Event Management System

Muhammad Ali Uzair Marzele[1], Rozita Abdul Jalil*

[1] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: rozita@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2025.06.01.099

Article Info

Received: 13 June 2024
Accepted: 19 June 2025
Available online: 30 June 2025

Keywords

UTHM Event Management System, UTHM, EMS, Web-Based, Prototype model

Abstract

Currently, there are more than 60 student organizations and clubs at Universiti Tun Hussein Onn Malaysia (UTHM) with formal registrations. Students find that managing event applications by hand requires a significant amount of time and effort. Therefore, the objective of this project is to develop a system that effectively manages all of the data related to the different events the organization hosts. The UTHM Event Management System (EMS) will be developed using the object-oriented methodology and prototype model. The goal is to keep all event-related data in one central MySQL database. Numerous tasks and procedures required for effective event data handling are made easier by the system. This system has been designed using three iterations of the evolving software prototyping approach. Event administration is streamlined and event applications are systematized with a web-based UTHM EMS developed using PHP. The UTHM EMS makes it possible to retrieve data quickly and with little input effort. It helps minimize errors caused by human error. The system is accessible from anywhere and provides easy-to-use interfaces for rapid data collection and information searches. In conclusion, the UTHM EMS is helpful for effectively and successfully managing information about UTHM events.

1. Introduction

The Universiti Tun Hussein Onn Malaysia has around 60 organizations and clubs as of 2024, aimed at providing students with opportunities to participate in various programs aligned with their interests. Though the existing event management in UTHM is largely manual, there is no software for independent solutions yet other than Google and Microsoft Form. These challenges threaten the integrity of data and present barriers to effective coordination arising from issues such as overlapping participation, non-streamlined event application, and non-centralized event data. The overlapping events can lead to attendance conflicts due to a lack of information about the event time. Attendance conflicts can lower the participation of the event and event success. Poor communication between participants and the organizer will lead to confusion among them. Real-time access to event data is restricted by the lack of a centralized infrastructure, which makes it challenging to monitor participant activity and make educated judgements.

The objectives of this system are to design a web-based management system for UTHM events management using an object-oriented approach, develop a web-based management system for UTHM events management, and test the developed system. Three user roles are involved in the UTHM EMS system which are event planners (organizers), students (participants), and administrators. The system covers important modules such as user registration, login, account management, event creation, event joining, and attendance management with a focus on user-friendliness.

This article consists of six parts, which are introduction of the project, literature review, methodology, system development analysis and design, results and discussion of the project, conclusions of the system that has been developed.

2. Literature Review

This section discusses the type of existing system to be compared with the system that will be developed. Each system has its own advantages and disadvantages depending on the type of system that suits the user perfectly.

2.1 Introduction

This chapter will discuss the literacy studies that have been analyzed and studied based on previous research topics. The purpose of this literacy study is to provide a clear picture of some of the questions to be studied through various methods to find out the requirements that need to be met. In achieving analysis and meeting the requirement needed, reading methods, evaluation, and analysis are mainly related to this event management system topic needed in the development process of this project.

2.2 Case Study

Event planners frequently use Google Forms or Microsoft Form, which simplifies participant registration but has drawbacks with centralized infrastructure and real-time data access [1]. Although it makes registration for events easier and reduces issues with attendance, it has drawbacks such as slow participant activity tracking and poor decision-making. Due to the basic structure of the platform, attendees' preferences and potential areas for event enhancement may be overlooked. Furthermore, there are not many effective communication tools on the site between organizers and participants, which might lead to miscommunication and information being delayed [2]. Google Forms does a good job of handling attendance conflicts and basic event communication, but it falls short when it comes to real-time data, therefore event organizers will need to think about sustainability standards outside of the digital sphere [3].

2.3 Method or Technology

Event management is a system that encompasses the planning, organization, and execution of live events [4]. Events such as a conference, exhibition, concert, or even the debut of a new product or brand may all benefit from event management services [4]. This means planning a private or professional gathering. The current manual system will be replaced with the web-based UTHM Event Management System. Computerizing every step of the application, registration, participation, and payment processes for events can result in a better organized management system.

There are three languages most frequently used for web-based programming. Python, PHP, and JavaScript are the languages used. PHP stands for Hypertext Preprocessor. A popular general-purpose programming language that is free and open source that may be used with HyperText Markup Language (HTML) for web development [5]. Because it is deep enough to manage Facebook, the largest social network, and strong enough to form the basis of the largest blogging system on the internet, this language is employed. PHP can therefore handle data better and is easier to adapt to the needs of the project, making it a better choice for this one. Among the greatest languages for quickly and safely establishing a database connection is PHP.

For the database connection, Structured Query Language has been abbreviated to SQL. Databases are communicated using the SQL language [6]. In addition to creating databases, the author may read, edit, and work with data in them using SQL. MySQL is an open-source database server system. MySQL is compatible with Windows, Linux, and Mac OS in addition to supporting regular SQL. MySQL's excellent performance, great reliability, and user-friendliness are its main draws for authors. Because MySQL is open-source software, MySQL will be used in the development of the system.

2.4 System Comparison

When systems are compared, each has unique characteristics and features. Finding the best solution for a given set of requirements requires an understanding of these variations. Making educated decisions is made possible by a system comparison, whether assessing user interfaces, data management capabilities, or overall efficiency. With optimal performance and user satisfaction guaranteed, the study helps choose the system that best fits the needs and objectives. The existing systems that has been choose is Eventbrite, Cvent and Bizzabo. Table 1 below regarding the comparison between the existing systems and the UTHM EMS system.

Table 1 *Comparison of existing systems*

Features/System	Eventbrite [7]	Cvent [8]	Bizzabo [9]	UTHM EMS
User Login and Registration	Yes	Yes	Yes	Yes
Manage Account	Yes	Yes	Yes	Yes
Event Creation	Yes	Yes	Yes	Yes
Manage Attendance	Yes	No	No	Yes
Join Event	Yes	Yes	Yes	Yes
Payment	Yes	Yes	Yes	Yes

Even though these three systems have different intended users and features, taken as a whole, they provide insightful information about the state of event management systems today. Considering their advantages and disadvantages offers a chance to create more inventive systems that can adapt to changing consumer needs and preferences.

3. Methodology/Framework

Model that will be used to develop the web based UTHM Event management System is Prototype Model. The prototype model is a systems development method in which a prototype is built, tested, and then reworked as necessary until an acceptable outcome is achieved from which the system or product can be developed [10]. The prototype model in this project will follow phase, which is initial requirements, analysis, design, prototype, user evaluation, review and update, implementation, and testing. Fig. 1 shows Prototype Model. All activities that are involved in the phase prototype model are in Table 2.

The Prototype Model has a major influence on the project's development because it enables early system feature visualization and facilitates feedback from stakeholders, including students and event planners, at every stage of the process. This lowers the possibility of significant changes occurring late in the project and improves comprehension of user requirements. The system can be continuously enhanced to meet user expectations by incorporating users in the user evaluation, review, and update phases. Additionally, the model's iterative approach enhances system usability, facilitates flexibility in adapting to new or changing requirements, and aids in the early identification of problems. Moreover, this system contributes to the sustainability and scalability of event management at UTHM by promoting higher participation rates, fostering better organization, and supporting data-driven decision-making. It also lays the foundation for long-term digital transformation within the institution.

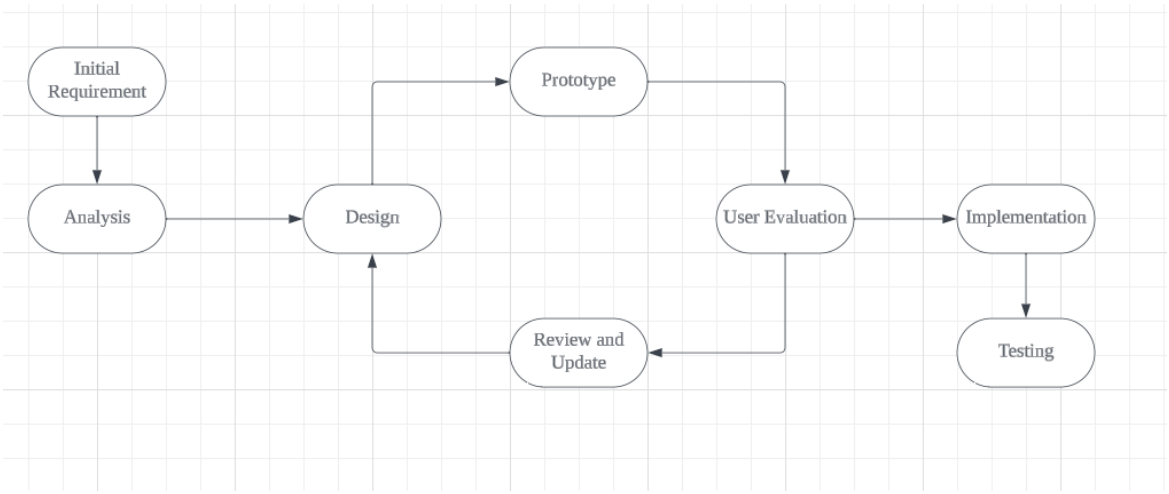

Fig. 1 *Prototype model [10]*

Table 2 *Software development activities and tasks*

Phase	Task	Output
Initial requirement	Determining the problem and project timeline. Analyze the requirements, interviews, and observations. Visualize the design of the system.	Project Proposal Project Requirements
Analysis	Identifying problems through interviews and observation Determine requirements for new system that will used	Use Case Diagram UML Diagram ERD
Design	Designing the system architecture, user interface, database schema, and website structure based on the requirements.	Database design Web user interface design
Prototype	Develop the prototype according to the gathered requirements and design	Prototype
User Evaluation	Interact with the prototype toevaluate its functionality, usability, and overall user experience	User's review
Review and update	Review the feedback and make an update according tothe feedback	Updated prototype
Implementation	Develop the final product based on all user's requirements, feedback, and prototype.	Full report of theproject Complete thewebsite
Testing	Test the validity of thewebsite	Test the validity of thewebsite

4. Analysis and Design

After completing the planning stage, the next step is analysis and design, as discussed in Chapter 4. This chapter will elaborate on the system database flow and showcase the system architecture. The design and development of data in the system will be supported by models such as Unified Modelling Language (UML) and Class Diagram, which are essential for database production [11]. Following this chapter is crucial to ensure the implementation of a system that not only meets user demands but also aligns with the previously outlined goals.

4.1 System Requirement Analysis

Functional requirements are system functions that are implemented to enable users to complete the tasks performed. This process also explains to the development team and stakeholders the nature of the inner workings of the system. Non-functional requirements are related to how integrity, security, availability, compatibility, and usability of functions in the system are performed. Table 3 and Table 4 show the functional and non-functional requirements for UTHM Event Management System.

Table 3 *Functional Requirement*

Module	Functionality
1. User Registration and Login	- Allows users to register for the system. - Registered users can log in to access the web-based event management.
2. Manage Account Module	- Allows users to edit and update their personal information. - Ability to update their information anytime that they want.
3. Event Creation Module	- Allows users to create new events by providing important information such as event names, date time location and a brief description. - Ability to determine registration deadlines, set up ongoing events, and set ticket pricing
4. Join Event Module	- Can join the event that they meet their interest. - Can join any event anytime that they want if the event is not expired.
5. Manage Attendance Module	- Allows organizer to update the attendance of the participants. - Ability to keep tracking the participant like how many participants will join, change the capacity of the participant, rejecting the participant who failed to pay the fee after due payment and generate the QR code for attendance.
6. Payment Management Module	- Can make the payment after joining the event.

Table 4 *Non-functional Requirement*

No	Requirement	Description
1	Performance	- The system should always be usable. - The system should allow users to place orders in a short period of time.
2	Operational	- The loading time required for a website is no more than 1 minute. - The system can be updated and maintained with easy.
3	Security	- The system should be safe from SQLI attack. - The URL cannot be manipulated.
4	Usable	- The user interface should be responsive and user-friendly. - The system should be easy to understand

4.2 Use Case Diagram

There are three actors in the system which are Participant, Organizer, and Administrator. There are 6 main functionalities in this system which are Register and Login, Manage Personal Information, Manage Event Creation, Join Event, Make Payment and Manage Attendance. The main functionalities and overall activity in this system as shown as use case in the Fig. 2

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 1782-1799

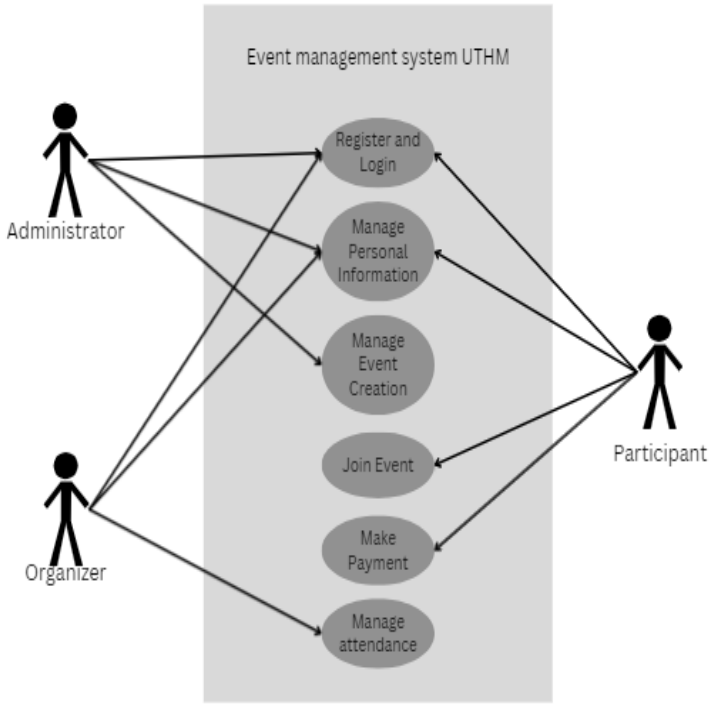

Fig. 2 *Use Case Diagram of UTHM EMS System*

4.3 Use Case Specification

Use case specification is used to describe and explain the details of use case diagrams of event management system UTHM event. It is an important section that offers a thorough explanation of the system's functionality. It describes how users, including Participants, Organizers, and Administrator will communicate with the system and how certain events will cause the system to react. The use case specification will show the details of the process of each functionality which are Register and Login, Manage Personal Information, Manage Event Creation, Join Event, Manage Attendance, Make Payment. Writing a use case specification is important for provides a systematic, structured, and detailed overview to understand the system's functionalities and process.

4.4 Class Diagram

UML Class Diagram is a visual notation for the design and representation of object-oriented systems. Such a diagram represents the structure of the system by showing the classes, attributes, functions, and relationships between its objects. The UTHM EMS system's class diagram is as shown in Fig. 3

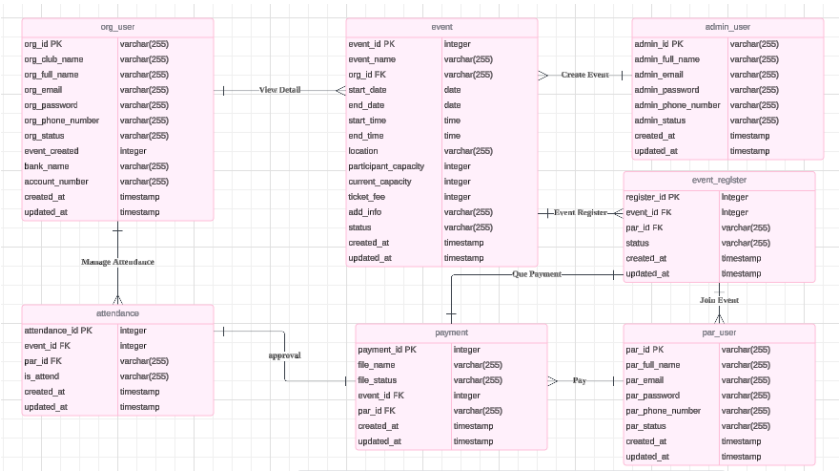

Fig. 3 *Class Diagram*

4.5 Flowchart

A flowchart shows each stage in a logical sequence of the process. It is a general tool that may be applied to various tasks used to describe several processes including administrative. The flowchart process of the participants is shown in Fig. 4 in the system where participants will be able to login into the system and redirect towards the home page. Fig. 5 describes the flowchart process of the organizer in the system where the organizer will be able to login into the system and redirect to their home page. Fig. 6 displays the respective flowchart process of the administrator in the system where the administrator will be able to login into the system and redirect to their home page.

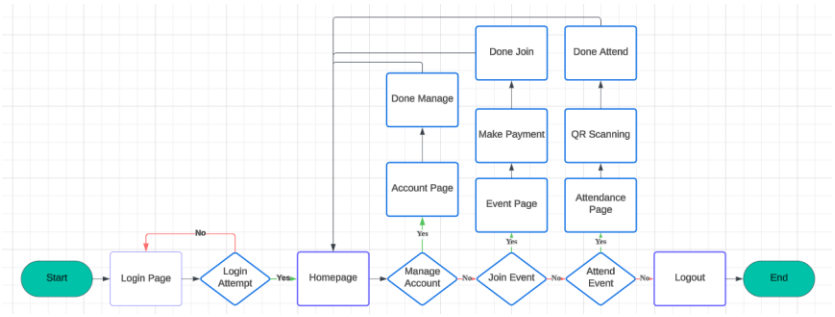

Fig. 4 *Flowchart of Participant Process*

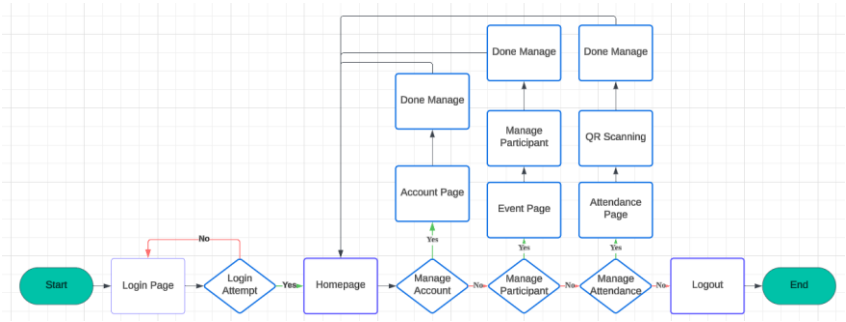

Fig. 5 *Flowchart of Organizer Process*

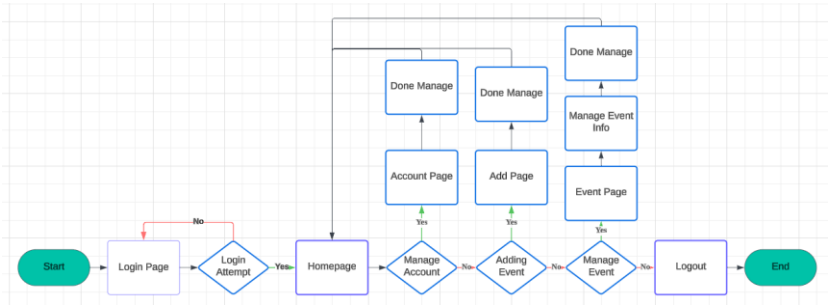

Fig. 6 *Flowchart of Administrator Process*

4.6 Interface Design

The design of the system interface is one of the important and necessary things for every system development that is made. It aims to give an idea of how the interface of the system is to be developed. A good system interface design will help especially in addressing system requirements. The following are the interfaces that have been designed based on each module.

4.6.1 Login And Registration Interface

The user interface of login as shown in Fig. 7 and user interface of registration as shown in Fig. 8. They need to fill in the Matric Number or ID and Password. The information will be validated. If the information is correct, it will be redirected to the homepage. If the information is not correct, the error message will pop up. If a user needs to register, they need to click the sign-up link.

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 1782-1799

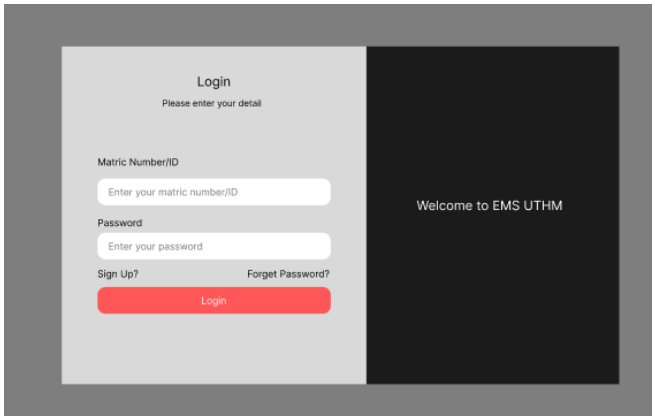

Fig. 7 *Login*

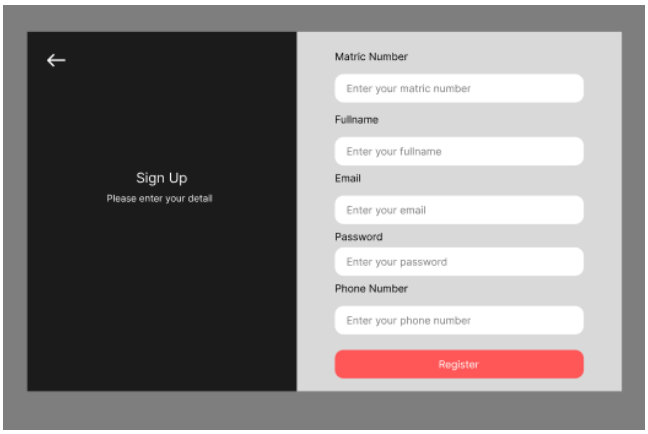

Fig. 8 *Registration*

4.6.2 Manage Personal Information Interface

User can manage personal information, update account information and contact number on account in Fig. 9.

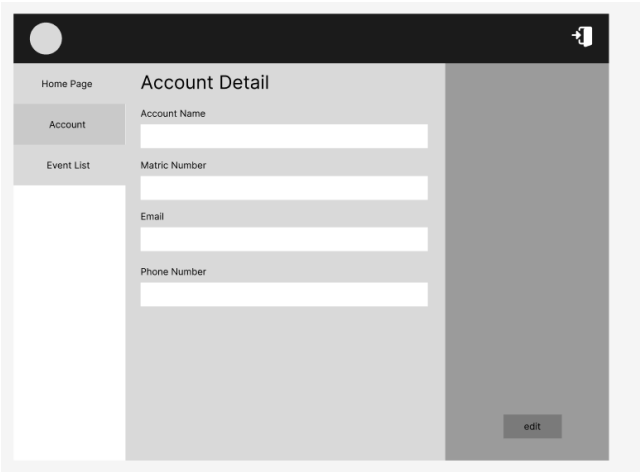

Fig. 9 *Manage Personal Information*

4.6.3 Event Creation Interface

Event Creation Interface will be in role administrator only because the administrator only can add or create the event. The administrator will fill in the details of the event and click the button create to add it. The user interface is shown in Fig. 10.

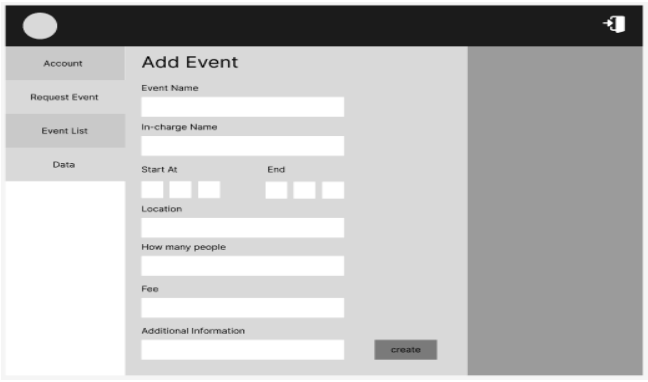

Fig. 10 *Event Creation*

4.6.4 Join Event Interface

Join Event Interface will be in role participant only because participants are the only user that can join the event. Participants will join the event that they are interested in and want to join. They need to click on the join button. If successful, it will redirect to the payment page. The user interface is shown in Fig. 11.

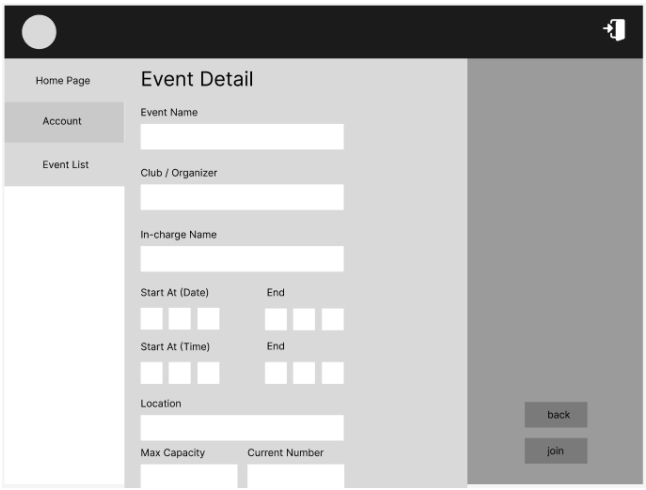

Fig. 11 *Join Event*

4.6.5 Payment Interface

After participants join the event, they need to make a payment. They will make the payment to the organizer account. The details of the bank will be displayed in the interface. Then click upload to continue. The user interface is shown in Fig. 12.

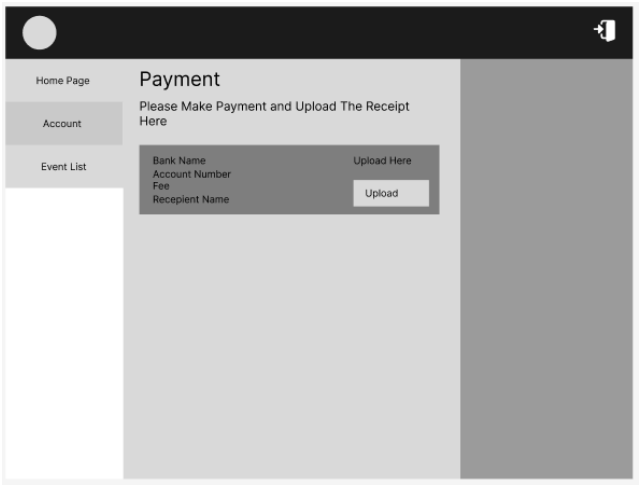

Fig. 12 *Payment*

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 1782-1799

4.6.6 Manage Attendance Interface

Manage attendance will be conducted by organizer where organizer needs to scan the QR code of the participant. Every event the participant joined will be given a QR code. The QR code needs to be shown to the person in charge of scanning. The user interface of the QR code is shown in Fig. 13.

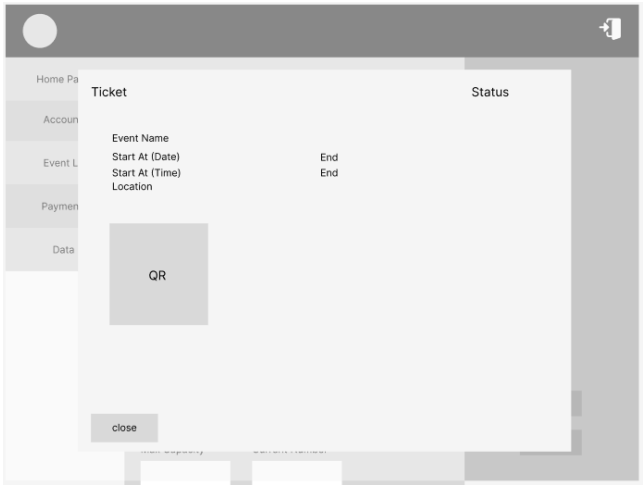

Fig. 13 *user interface of the QR code*

5. Result and Discussion

This section focuses on system implementation and testing for the web-based UTHM Event Management System. It covers the implementation of Login and Registration, Manage Personal Information, Event Creation, Join Event, Payment, and Manage Attendance module functionalities. Additionally, it discusses the comprehensive functional testing conducted to the system's performance.

5.1 Implementation

This section describes the development of functional modules in a system. Program code is provided to aid clarification.

5.1.1 Login And Registration Interface

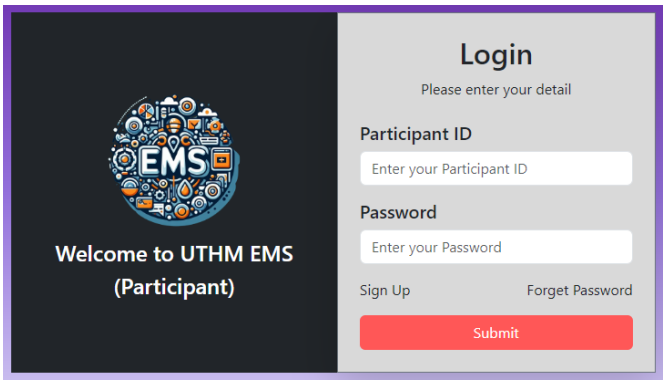

Fig. 14 *User Interface Login*

```php
public function verifyuser()
{
    $par_id = $_POST["par_id"];

    $user = DB::table('par_user')->where('par_id', $par_id)->first(); // Fetch user data

    if ($user) {

        // If user exists, set the properties of $ParUser object

        if ($_POST["par_password"] == DB::table('par_user')->where('par_id', $par_id)->value('par_passwor

            session(['par_id' => $user->par_id]);
            session(['par_full_name' => $user->par_full_name]);
            session(['par_email' => $user->par_email]);
            session(['par_password' => $user->par_password]);
            session(['par_phone_number' => $user->par_phone_number]);
            session(['par_status' => $user->par_status]);
            session(['created_at' => $user->created_at]);
            session(['updated_at' => $user->updated_at]);

            $alert = "Welcome to EMS UTHM, " . $user->par_full_name . " - (PARTICIPANT)";

            session(['alert' => $alert]);

            $today = Carbon::now()->toDateString();

            $events = DB::table('event')->whereDate('end_date', '<=', $today)->update(['status'=> "Not Av

            $getevent = DB::table('event_register')->where('par_id', $par_id)->orWhere(function (Builder
                $query->where('status', 'success')
```

Fig. 15 *Source code Login*

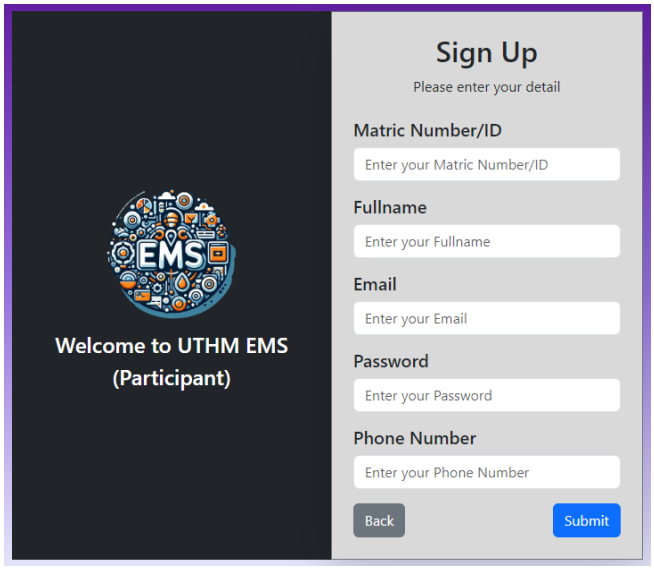

Fig. 16 *User Interface Registration*

```php
$check_user = DB::table('par_user')->where('par_id', $par_id)->first();

if (empty($check_user)) {

    DB::table('par_user')->insert([
        'par_id' => $par_id,
        'par_full_name' => $par_full_name,
        'par_email' => $par_email,
        'par_password' => $par_password,
        'par_phone_number' => $par_phone_number,
        'par_status' => $par_status,
        'created_at' => $created_at,
        'updated_at' => $updated_at
    ]);

    unset($_POST["par_id"]);
    unset($_POST["par_password"]);
    unset($_POST["par_fullname"]);
    unset($_POST["par_email"]);
    unset($_POST["par_phone_number"]);

    $alert = "Registeration Successful";
```

Fig. 17 *Source code Registration*

Fig. 14 show the user interface of the Login. Fig. 15 show the source code for Login where have the verification of user. The verification will use SQL code to connect with the database. It also use the method that already Laravel prepared to use the SQL. The successful verification will redirect to home page. The failed verification will return to login again with error pop up. Fig. 16 show the user interface of the Registration. Fig. 17 show the source code for Registration. The registration start with checking the user if it already exist or not. If the user not exist, then it will proceed to insert the data into database using SQL Laravel method. If the user exist, it will redirect login page with user exist message.

5.1.2 Manage Personal Information Interface

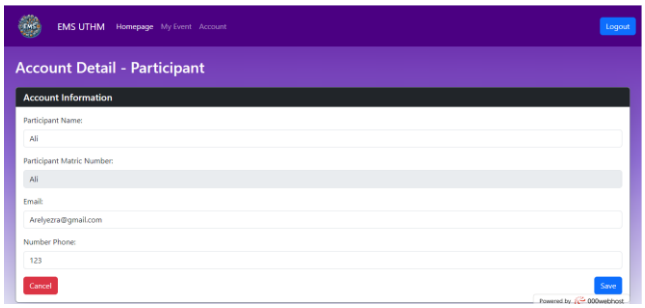

Fig. 18 *User Interface Manage Personal Information*

```php
public function updateacc()
{

    date_default_timezone_set('UTC');
    $updated_at = date("Y-m-d H:i:s", strtotime("-16 hours"));

    DB::table('par_user')
        ->where('par_id', session('par_id'))
        ->update([
            'par_full_name' => $_POST['par_full_name'],
            'par_email' => $_POST['par_email'],
            'par_phone_number' => $_POST['par_phone_number'],
            'updated_at' => $updated_at
        ]);

    session(['par_full_name' => $_POST['par_full_name']]);
    session(['par_email' => $_POST['par_email']]);
    session(['par_phone_number' => $_POST['par_phone_number']]);

    $alert = "The account information has been update";

    session(['alert' => $alert]);

    return redirect('/par/acc');
}
```

Fig. 19 *Source Code Manage Personal Information*

Fig. 18 show the user interface of Manage Personal Information. Fig. 19 shows the source code of Manage Personal Information. The user needs to fill the input to update the information. After fill and check the input, click the save button to save the information. It will execute the SQL update query through the Laravel method to update at the database.

5.1.3 Event Creation Interface

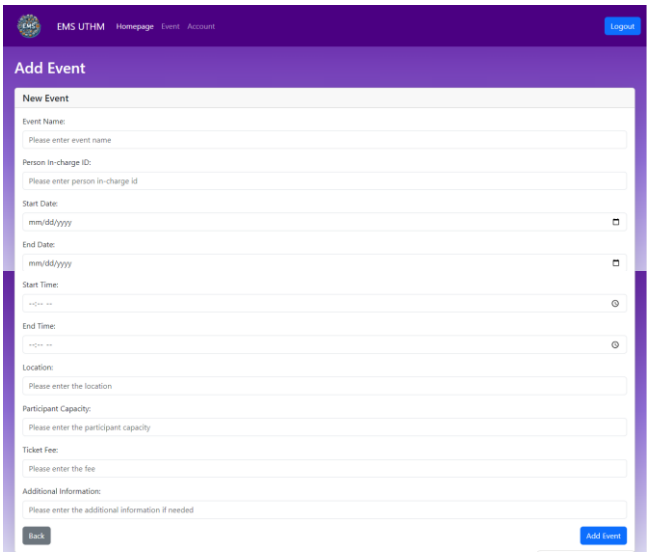

Fig. 20 *User Interface Event Creation*

```php
if ($checkorgid) {

    //kalau org exist

    $event_name = $_POST["event_name"];
    $in_charge_id = $_POST["in_charge_id"];
    $start_date = $_POST["start_date"];
    $end_date = $_POST["end_date"];
    $start_time = $_POST["start_time"];
    $end_time = $_POST["end_time"];
    $location = $_POST["location"];
    $participant_capacity = $_POST["participant_capacity"];
    $current_capacity = 0;
    $ticket_fee = $_POST["ticket_fee"];
    $add_info = $_POST["add_info"];
    $created_at = date("Y-m-d H:i:s");
    $updated_at = date("Y-m-d H:i:s");
    $status = "Available";

    $start_time = $start_time . ":00";
    $end_time = $end_time . ":00";

    DB::table('event')->insert([
        'event_name' => $event_name,
        'org_id' => $in_charge_id,
        'start_date' => $start_date,
        'end_date' => $end_date,
        'start_time' => $start_time,
        'end_time' => $end_time,
        'location' => $location,
        'participant_capacity' => $participant_capacity,
        'current_capacity' => $current_capacity,
        'ticket_fee' => $ticket_fee,
        'add_info' => $add_info,
        'status' => $status,
        'created_at' => $created_at,
```

Fig. 21 *Source Code Event Creation*

Fig. 20 shows the user interface of Event Creation. Fig. 21 shows the source code of the Event Creation. The administrator needs to fill in the form according to the event details. After finishing filling in the form, Admin need to click the add button to add the data to the database. It will execute the SQL insert query through the Laravel method to update at the database.

5.1.4 Join Event Interface

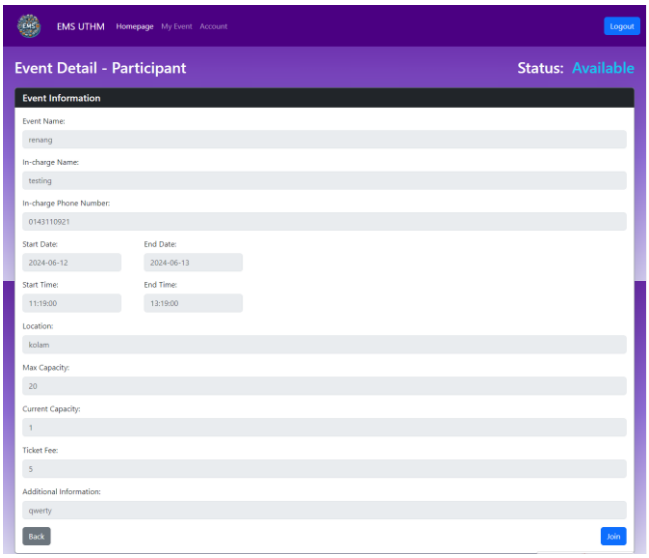

Fig. 22 *User Interface Join Event*

```php
$event_id = $request->query('event_id');

$event_date_range = array();

$eventdetail = DB::table('event')->where('event_id', $event_id)->first();

$date1 = new \DateTime($eventdetail->start_date);
$date2 = new \DateTime($eventdetail->end_date);

$interval = new \DateInterval('P1D'); // 1 day interval
$date_range = new \DatePeriod($date1, $interval, $date2->modify('+1 day'));

foreach ($date_range as $date) {
    array_push($event_date_range, $date->format('Y-m-d'));
}

$busy_day = session('busy_day');

$date1_timestamps = array_map('strtotime', $busy_day);
$date2_timestamps = array_map('strtotime', $event_date_range);

$result_timestamps = array_intersect($date1_timestamps, $date2_timestamps);

//

$result = DB::table('event_register')->where('par_id', session('par_id'))->where('event_id', $event_id)->get();

if ($result->count() == 0) {

    //tak join lagi
```

Fig. 23 *Source Code Join Event*

Fig. 22 shows the user interface of Join Event. Fig. 23 shows the source code of Join Event. The participant can join the event that they are interested in by clicking the join button. The system will get the information and

use it to find it in the database. It will search first whether the participant already joined the event. If the participant did not join the event yet, it will redirect to the payment page.

5.1.5 Payment Interface

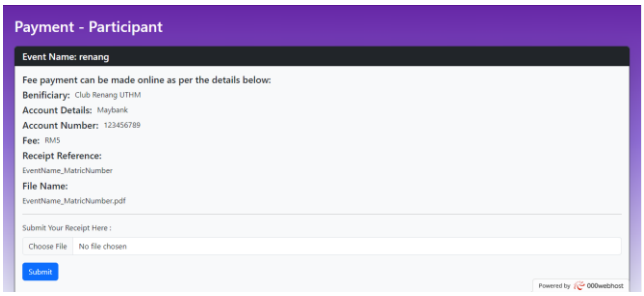

Fig. 24 *User Interface Payment*

```php
if (isset($_FILES['payment_file'])) {

    //kalau file ade

    if ($_FILES['payment_file']['type'] == 'application/pdf') {

        //file pdf only

        $file = $request->file('payment_file');

        $filedestination = "uploads";

        $file_name = $_FILES['payment_file']['name'];

        $file_status = "pending";

        $created_at = date("Y-m-d H:i:s");
        $updated_at = date("Y-m-d H:i:s");

        if ($file->move($filedestination, $file->getClientOriginalName())) {

            //done move file

            DB::table('payment_file')->insert([
                'file_name' => $file_name,
                'file_status' => $file_status,
                'event_id' => $event_id,
                'par_id' => session('par_id'),
                'created_at' => $created_at,
                'updated_at' => $updated_at,
            ]);

            DB::table('event_register')->insert([
                'event_id' => $event_id,
                'par_id' => session('par_id'),
                'status' => $file_status,
```

Fig. 25 *Source Code Payment*

Fig. 24 shows the user interface of Payment. Fig. 25 shows the source code of Payment. Participant needs to make the transaction to the bank of organizer. The details of the bank information will be shown on the payment page. Participant needs to upload the receipt of transaction on the input field and click the submit button. The system will be checking the file type whether the file type is pdf or not. If successful, it will insert and update the database for the organizer to check it later.

5.1.6 Manage Attendance Interface

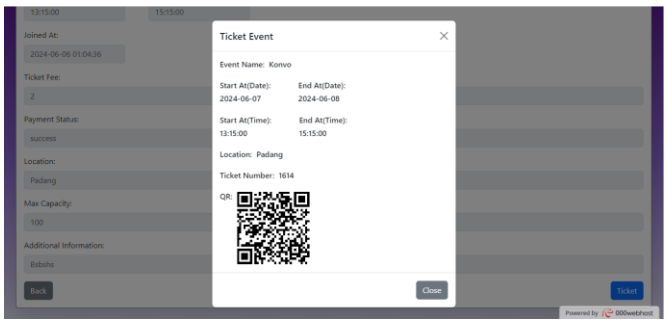

Fig. 26 *User Interface Manage Attendance (QR Code)*

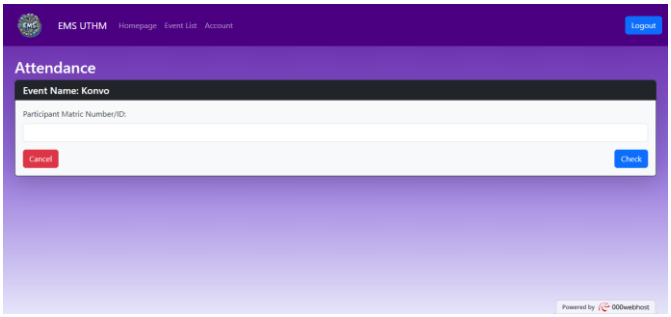

Fig. 27 *User Interface Manage Attendance (Matric Number)*

```php
public function approved(Request $request)
{

    $par_id = $request->query('par_id');
    $event_id = $request->query('event_id');

    date_default_timezone_set('UTC');
    $created_at = date("Y-m-d H:i:s", strtotime("-16 hours"));
    $updated_at = date("Y-m-d H:i:s", strtotime("-16 hours"));

    DB::table('event_register')
        ->where('par_id', $par_id)
        ->where('event_id', session('temp_event_id'))
        ->update(['status' => "success", "updated_at" => $updated_at]);

    DB::table('payment_file')
        ->where('par_id', $par_id)
        ->where('event_id', session('temp_event_id'))
        ->update(['file_status' => "success", "updated_at" => $updated_at]);

    DB::table('event')
        ->where('event_id', session('temp_event_id'))
        ->increment('current_capacity', 1);

    DB::table('event')
        ->where('event_id', session('temp_event_id'))
        ->update(["updated_at" => $updated_at]);

    DB::table('attendance')->insert([
        'event_id' => $event_id,
        'par_id' => $par_id,
        'is_attend' => "Not Yet",
        'created_at' => $created_at,
        'updated_at' => $updated_at,
    ]);

    return view("org.parlist", ["event_id" => session('temp_event_id')]);
}
```

Fig. 28 *Source Code Manage Attendance*

Fig. 26 shows the user interface of Manage Attendance by QR code. The organizer can scan the QR code of participant to approve and update the database. Fig. 27 shows the user interface of Manage Attendance by searching for the matric number. The organizer will search for the matric number and update the attendance. Fig. 28 shows the source code of the Manage Attendance where the system will search and update the participant attendance through the SQL Laravel method.

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 1782-1799

5.2 Testing Result

Table 5 *Testing Results*

Module	Description	Expected Result	Actual Results	Pass/Fail
Login and Registration	To check whether participant, organizer and administrator can register an account	Participant, organizer and administrator able create account	Participant, organizer and administrator successfully create account	Pass
	To check whether participant, organizer and administrator can log in to system	Participant, organizer and administrator able log in to system	Participant, organizer and administrator successfully log in	Pass
Manage Personal Information	To check whether participant, organizer and administrator can edit their profile	Participant, organizer and administrator able to edit their profile	Participant, organizer and administrator successfully edit their profile	Pass
Event Creation	To check whether administrator can add event	Administrator able to add the event	Administrator successfully add the event	Pass
Join Event	To check whether participant can join the event	Participant able to join the event	Participant successfully join the event	Pass
Payment	To check whether participant can submit the payment receipt	Participant able to submit the payment receipt	Participant successfully submit the payment receipt	Pass
Manage Attendance	To check whether organizer can manage attendance through QR code scanning	Organizer able to manage attendance through QR code scanning	Organizer successfully manage attendance through QR code scanning	Pass
	To check whether organizer can manage attendance through search matric number	Organizer able to manage attendance through search matric number	Organizer successfully manage attendance through search matric number	Pass

5.3 User Acceptance Testing

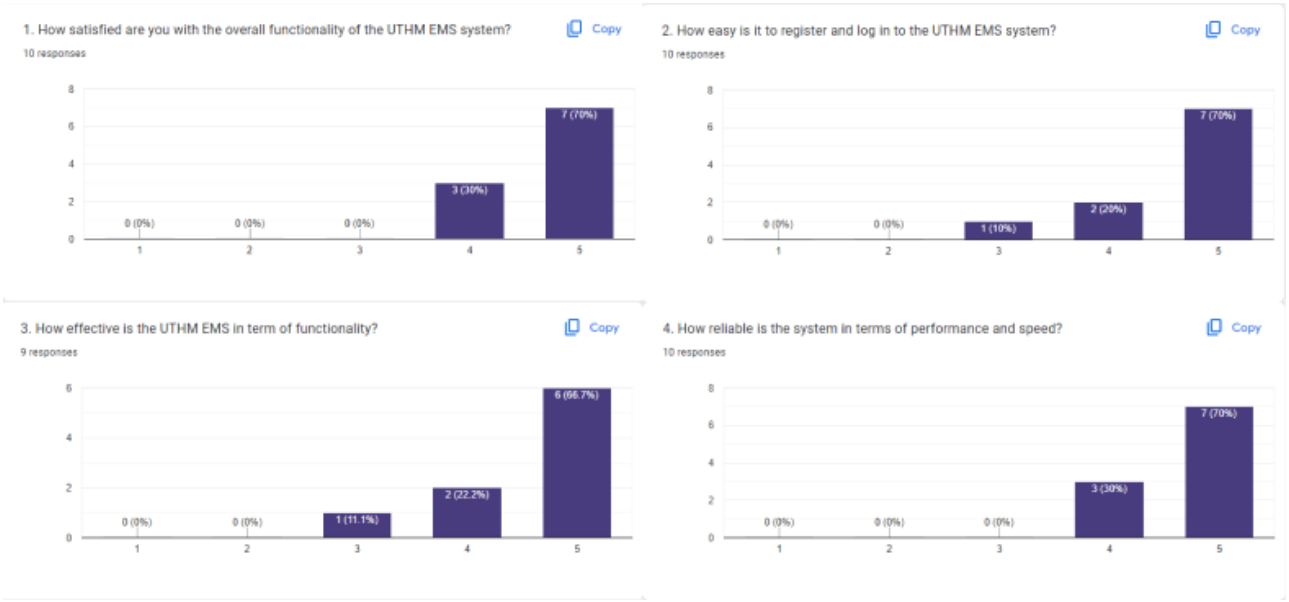

Fig. 29 *Graphs of Testing Result for System Funtionality*

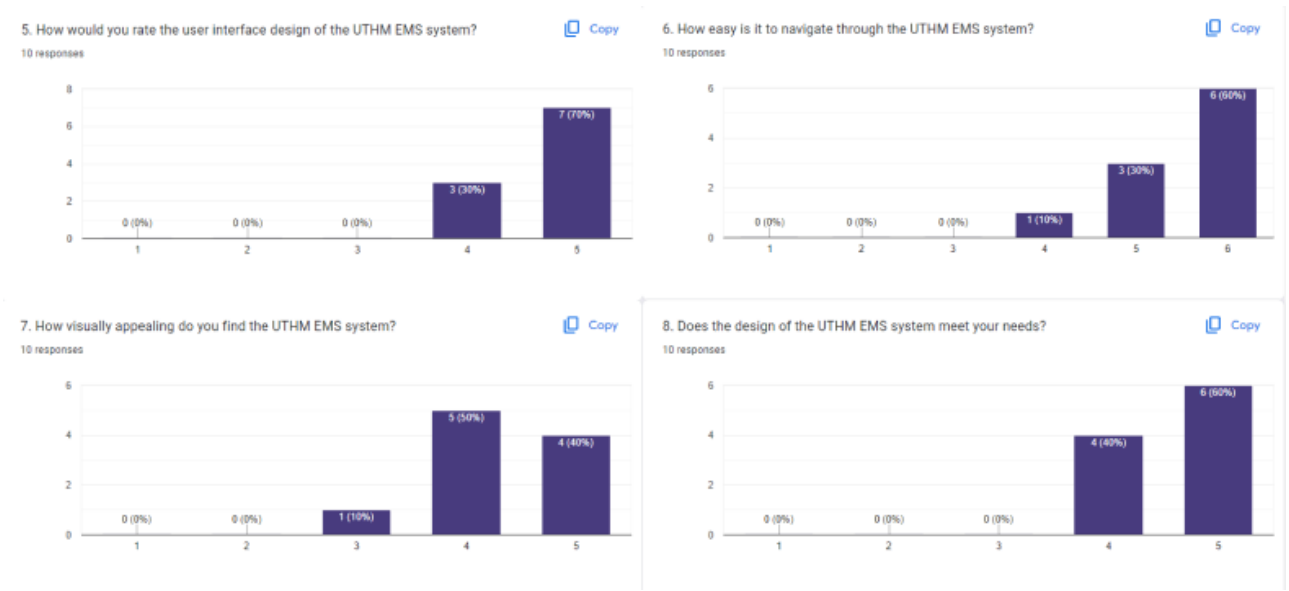

Fig. 30 *Graphs of Testing Result for System Funtionality*

Fig. 29 and Fig. 30 present the results from user acceptance testing for the UTHM Event Management System. It yielded positive results across all evaluated areas, with 10 respondents participating in the survey. In terms of overall functionality, 70% of respondents rated the system as 5 (highly satisfied), and 30% rated it a 4. The ease of registration and login was also highly rated, with 70% giving it a 5, 20% a 4, and 10% a 3. Effectiveness saw 66.7% of respondents rating it a 5, 22.2% a 4, and 11.1% a 3. Reliability in performance and speed was rated a 5 by 70% and a 4 by 30%. No respondents rated the system below a 3 in any category, indicating a strong overall approval. Based on this feedback, it is recommended to maintain and enhance the current system functionality, address any minor concerns with the registration and login process, and ensure ongoing system reliability and performance to sustain user confidence. The user interface design received a rating of 5 (highly satisfied) from 70% of respondents and 4 from 30%. Navigation ease was rated a 5 by 60%, a 4 by 30%, and a 3 by 10%. Visual appeal was rated as 5 by 40%, a 4 by 50%, and a 3 by 10%. Lastly, the design's ability to meet user needs was rated at 5 by 50% and a 4 by 50%. No respondents rated any aspect below a 3, indicating a positive reception. Recommendations include maintaining the current design standards, improving navigation ease, and ensuring the system continues to meet user needs effectively.

6. Conclusion

In conclusion, a number of significant issues with event management at Universiti Tun Hussein Onn Malaysia (UTHM) were resolved with the creation of the web-based UTHM Events Management System (EMS). Conflicts over event scheduling, dispersed and dispersed data management, misunderstandings between stakeholders, and the lack of a streamlined and effective system are a few of these difficulties. For both organizers and attendees, the EMS streamlines and improves the event management process by providing a centralized, user-friendly platform.

User registration and login, event creation and updating, personal information management, event participation, attendance tracking, and integrated payment processing are some of the system's key features. A well-organized and secure database that guarantees strong data management for user profiles, event specifics, registration records, and transactional data underpins these features. By decreasing manual labor, increasing accuracy, and simplifying communication, the EMS improves operational efficiency and the user experience for all users.

Additionally, by encouraging greater participation rates, better organization, and data-driven decision-making, this system helps UTHM's event management be sustainable and scalable. Additionally, it establishes the framework for the institution's long-term digital transformation.

Acknowledgement

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

Author Contribution

This paper requires that all authors take public responsibility for the content of the work submitted for review. The contributions of all authors must be described in the following manner:

The authors confirm contribution to the paper as follows: ***study conception and design****: Muhammad Ali Uzair Marzele, Rozita Abdul Jalil;* ***data collection****: Muhammad Ali Uzair Marzele, Rozita Abdul Jalil;* ***analysis and interpretation of results****: Muhammad Ali Uzair Marzele, Rozita Abdul Jalil;* ***draft manuscript preparation****: Muhammad Ali Uzair Marzele, Rozita Abdul Jalil. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] J. Doe, "The Efficiency of Google Forms in Event Planning," Journal of Event Management, vol. 5, no. 2, pp. 123-130, 2022. https://www.eventmanagementjournal.com/google-forms-efficiency
[2] K. Johnson, "Communication Tools in Event Management Platforms," Journal of Digital Communication, vol. 4, no. 2, pp. 98-105, 2020.
[3] R. Lee, "Sustainability Standards in Digital Event Planning," Sustainable Events Quarterly, vol. 2, no. 4, pp. 67-74, 2023.
[4] F. Bouchon, "EVENT MANAGEMENT EDUCATION AND EVENT INDUSTRY: a CASE OF MALAYSIA," Sep. 06, 2017. https://adum.um.edu.my/index.php/MOJEM/article/view/6091
[5] L. Surya, Patel, D. M. Patel, and R. T. Yarlagadda, PHP For Beginners, Lunawada: Red'Shine Publication, 2021. [E-book] Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3834055.
[6] Robi Yanto. " Manajemen Basis Data Menggunakan MySQL ". Yogyakarta: Deepublish. 2016. ISBN 978-602-475-256-9.
[7] Eventbrite, "Eventbrite," *Eventbrite*. https://www.eventbrite.com/
[8] "CVENT | event platform for in-person, virtual, and hybrid events & webinars." https://www.cvent.com/
[9] "#1 Event management software for B2B conferences | Bizzabo," *Bizzabo*, Dec. 21, 2023. https://www.bizzabo.com/
[10] H. Chen, R. Kazman and S. Haziyev, "Strategic Prototyping for Developing Big Data Systems," IEEE Software, vol. 33, no. 2, pp. 36-43, 2016, doi: 10.1109/MS.2016.36.
[11] R. S. Pressman, Software Engineering: A Practitioner's Approach, 8th ed. New York, NY, USA: McGraw-Hill Education, 2014.