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

AITCS

Vol. 7 No. 1 (2026) 1396-1413
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

FSKTM Faculty Room Reservation System

Muhammad Shahrul Ilhan[1], Nazri Mohd Nawi[1]*

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

*Corresponding Author: nazri@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2026.07.01.075

Article Info

Received: 12 June 2025
Accepted: 18 June 2026
Available online: 30 June 2026

Keywords

Room, FSKTM, Booking

Abstract

The room booking process at the Faculty of Computer Science and Information Technology (FSKTM), UTHM, still relies on manual methods, often leading to confusion, overlapping reservations, and delayed confirmations. The absence of an efficient system makes it difficult for administrators to manage room reservations, especially as the number of users increases. Therefore, this project develops a web-based information system, known as BookIT, to modernize and centralize the room booking process at the faculty. The primary objective of this project is to create an automated booking system that allows users, such as students, lecturers, and administrative staff, to make reservations more efficiently and systematically. This project adopts the Agile methodology, which involves phases of requirements gathering, development, testing, and maintenance in iterative cycles. The system is designed using web technologies such as HTML, CSS, and JavaScript for the user interface, while PHP and MySQL are used for backend development and database management. Key features include real-time availability, automated booking, schedule conflict management, as well as automated notifications and booking confirmations. The system is also equipped with security features such as user authentication and data encryption to ensure information integrity. BookIT can improve booking efficiency, reduce manual errors, and accelerate the reservation approval process. Furthermore, the system also able to reduce administrative workload and enhance user experience.

1. Introduction

Managing rooms and resources is essential for Fakulti Sains Komputer Teknologi Maklumat (FSKTM) at UTHM to ensure smooth academic operations. Lecture halls, meeting rooms, and other facilities are frequently in use, making it important to handle bookings efficiently. Without a proper system, scheduling conflicts, delays, and errors can disrupt daily activities and impact the academic experience.

Currently, FSKTM relies on manual room booking processes, including emails, in-person requests, and paper-based methods. These approaches are time-consuming and prone to mistakes, often resulting in double bookings, miscommunication, and administrative inefficiencies. Users frequently face difficulties in checking room availability, and the lack of real-time updates further complicates the process. With the use of systems in organizations, it has made a significant impact and helped in the elimination of delays and minimization of errors [1].

To address these challenges, this project proposes the development of an online Room Booking System for FSKTM. The system will provide a centralized platform where students and staff can check room availability and make bookings in real time. This will help prevent scheduling conflicts and ensure more effective resource

management. By streamlining operations and managing reservations efficiently, an online booking system can improve user satisfaction [2].

In addition to reducing administrative workload, the proposed system will optimize room usage and minimize the chances of overbooking. It will also support better communication between users and administrators, enhancing the overall experience. By automating the booking process, the system aims to create a more organized and efficient academic environment at FSKTM.

2. Literature Review

This chapter reviews the literature and technologies relevant to developing the FSKTM Room Booking System. It begins with a case study of FSKTM, outlining the current challenges with manual room booking and data management. The chapter then explores key technologies, such as web-based platforms and real-time availability tracking, and explains their suitability for the project. A comparison of three existing room booking systems follows, evaluating their features, strengths, and limitations. Finally, the chapter compares the proposed system with these existing solutions, highlighting improvements and key features that address the current challenges at FSKTM.

2.1 FSKTM Faculty Room Reservation System

The FSKTM Room Booking System is a web-based platform designed to enhance room reservation management at FSKTM, UTHM. The system focuses on key features such as real-time availability tracking, efficient booking management, and automated conflict resolution. Its user-friendly interface allows students, staff, and administrators to easily reserve rooms, check availability, and manage bookings with minimal effort. Unlike more complex commercial systems, this system is simple, cost-effective, and tailored to meet the specific needs of FSKTM, ensuring that it is both practical and efficient. It aims to address the issues of manual booking processes, such as delays, errors, and scheduling conflicts, without introducing unnecessary complexity.

The FSKTM Room Booking System draws inspiration from existing solutions like the GEON Meeting Room Booking System, UMPoint Room Booking System, and UTeM Library Room Booking System. These systems offer features that support efficient room reservation and management, and the proposed system incorporates similar functionalities to streamline the booking process. By centralizing the booking system, it improves administrative workflows, reduces the potential for errors, and enhances overall user satisfaction, making room reservations at FSKTM more organized and efficient.

2.2 Study of Existing Related Systems

The review of existing room booking systems reveals that each system offers essential features for room reservations, but they vary in the range of functionalities provided. Some systems, such as GEON Meeting Room System and UMPoint, include room booking management, availability checks, and booking notifications, but they lack features like autonomous confirmation and feedback collection. Other systems like UTeM Library Room Booking offer booking management and availability checks but do not support autonomous confirmation or data reporting.

The proposed FSKTM Room Booking System aims to combine the best features of these systems while addressing their limitations. It includes comprehensive modules such as login, room booking management, autonomous confirmation, booking notifications, feedback collection, and data reporting. This combination ensures that the system is user-friendly, efficient, and tailored to the needs of FSKTM faculty, staff, and students. The system will streamline room booking processes, improve communication, and enhance administrative oversight, making it a more effective solution compared to the existing systems. Table 1 summarizes the findings of the system's comparison.

Table 1 *System Comparison*

Systems / Features	UMPoint	Utem Library Room Booking	Geon Meeting Room System	BookIT
Login	√	√	√	√
Room Booking Management	√	√	√	√
Room Availability Check	√	√	√	√
Autonomous Confirmation	√	X	√	√
Booking Notifications	X	√	√	√
Feedback Collection	X	X	X	√

3. Methodology

The development model chosen for this project is Agile, which is a flexible and user-centered method for developing product prototypes in the course of development [3], allowing the system to evolve with continuous input from users. Agile is particularly well-suited for the FSKTM Room Booking System, where user needs and feedback play a critical role in shaping the system throughout its development. Agile development is structured around sprints, which are short, time-boxed cycles focused on delivering small, functional parts of the system. At the end of each sprint, the project team gathers feedback from stakeholders, which is then used to refine and improve the system in the next iteration. This approach allows the project to adapt quickly to evolving requirements and ensures that the final system closely meets the needs of FSKTM's faculty, staff, and students. The Agile model encourages continuous progress, making it easier to adjust priorities and enhance the system incrementally based on feedback. Table 2 will display the software development activities and their task.

Table 2 *Software Development Activities and Their task*

Phase	Task	Output
Requirement Gathering	• Analyzing current manual booking procedures. • Defining objectives and scope of the Faculty Room Booking System. • Identifying required features and functions. • creating the data flow diagram and entity relationship diagram	• Initial understanding of requirements. • Documented objectives and scope • Documented objectives and scope • List of proposed features and functionalities • Gantt chart and project schedule • DFD, ERD, Flowchart
Design	• Developing user interface wireframes and prototypes. • Defining system navigation structure and flow. • Designing database schema	• User-friendly interface design. • Defined navigation structure. • Database schema • System architecture • Data dictionary
Development	• Front-end and back-end development. • Implementing interactive features. • Integrating security features.	• Functional application interface. • Interactive components. • Secure login and data handling.

Table 2 (cont)

Phase	Task	Output
Testing	• Conducting alpha and beta testing phases. • Collecting feedback from users.	• Testing report with identified issues. • Summary of user feedback.
Deployment	• Deploying the system for use by FSKTM faculty and students. • Providing user training sessions.	• Fully operational system. • User training summaries.
Maintenance and Updates	• Monitoring system performance. • Collecting user feedback for improvements. • Implementing periodic updates and bug fixes	• Maintenance logs and update records. • Continuous system enhancement.

3.1 System Analysis

The system analysis stage involves understanding the users' requirements and evaluating the current manual room booking processes at FSKTM to identify areas for improvement. This stage includes discussions with faculty staff, students, and administrators, as well as observing the existing booking methods, such as in-person requests, emails, and paper-based records. The analysis focuses on areas like room availability checking, booking management, and communication between users and administrators. Based on these observations, user needs are documented, and key areas for improvement are identified, ensuring the new system effectively addresses the challenges faced by the current manual processes.

3.1.1 Context Diagram

Figure 1 shows the context diagram with two external entities: the user and the admin. The admin manages room availability by updating room details and handling user reports, such as technical issues, by contacting the technician. Users can make bookings, update their information, submit reports, and provide feedback. They also receive notifications once their reservation is complete.

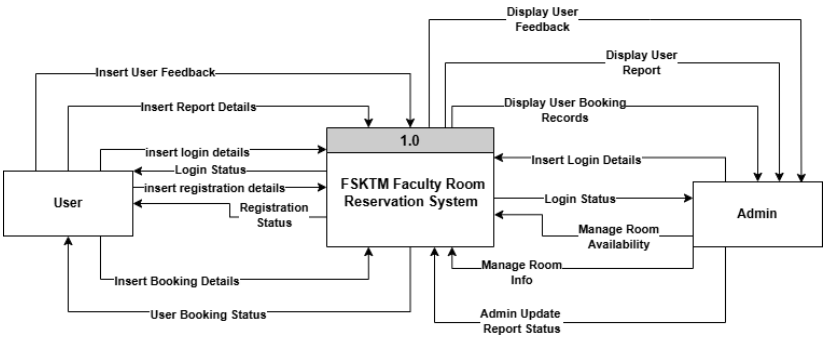

Fig. 1 *Context Diagram*

3.1.2 Data Flow Diagram

The FSKTM Room Reservation System consists of six key processes: User Login, User Registration, Room Availability Management, Room Booking, User Report Management, and User Feedback Management. These processes work together to provide an efficient solution for managing room reservations. The system uses seven data stores to manage and store information: User Data, Admin Data, Room Management Data, Room Reservation Records, Classes Data, User Reports, and User Feedback. Each data store ensures secure and organized storage of relevant data, such as user information, room availability, booking details, and feedback. Figure 2 shows how data flows between the processes and data stores, ensuring smooth operations and efficient room booking management at FSKTM.

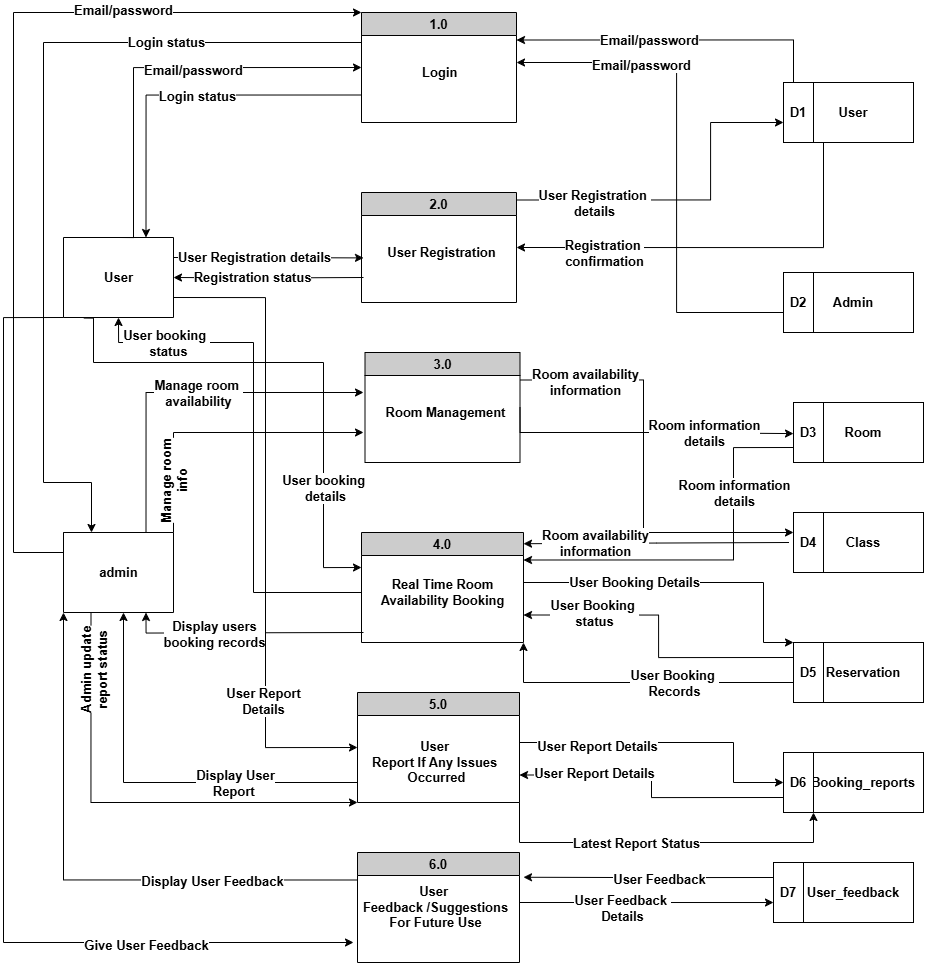

Fig. 2 *Data Flow Diagram*

3.1.3 Entity Relationship Diagram

An Entity Relationship Diagram (ERD) visually represents the relationships between different entities in a system. ERDs are essential for database design, as they help model the database structure and ensure data integrity. Figure 3 displays the ERD for the developed system, showcasing the relationships and cardinalities among the identified entities.

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1396-1413

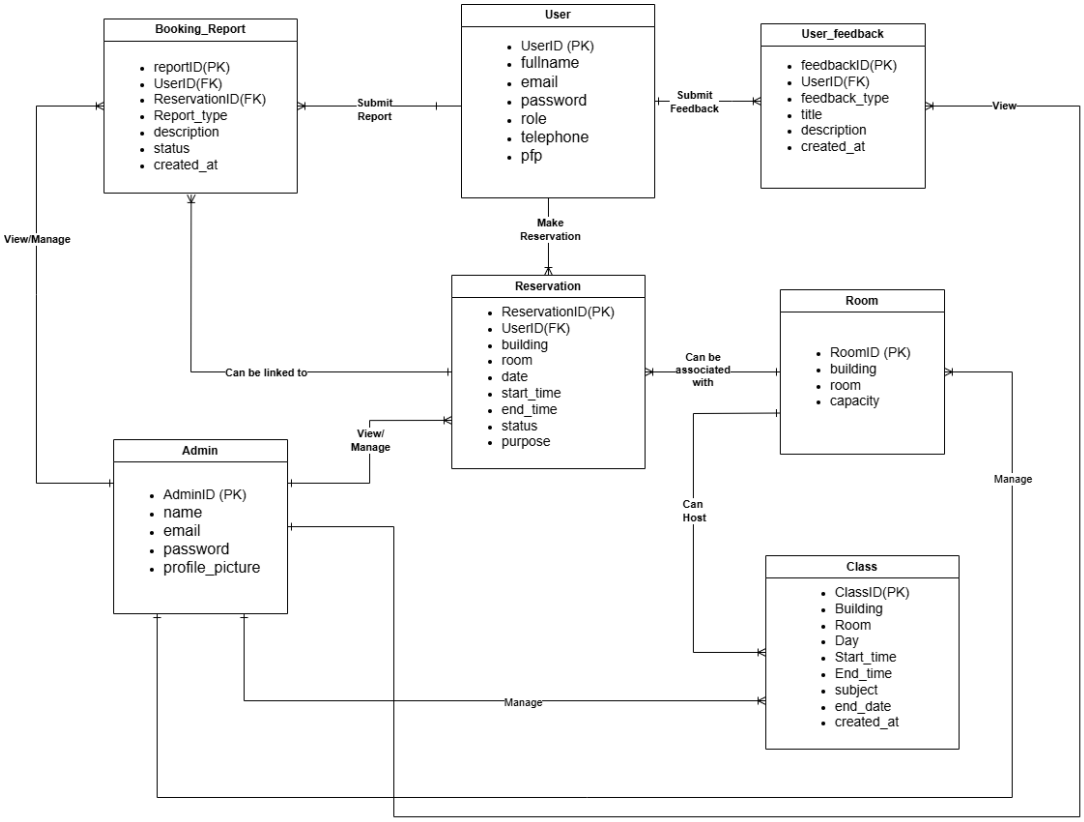

Fig. 3 *Entity Relationship Diagram*

4. System Design

The system design phase focuses on defining the architecture, database structure, and user interface (UI) of the FSKTM Faculty Room Reservation System. This phase ensures that the system is well-structured, efficient, and user-friendly.

4.1 System Architecture

The Faculty Room Booking System is designed using a three-tier architecture that separates the system into the Presentation Layer, Application Layer, and Database Layer. This structured approach ensures clear functionality, scalability, and efficient communication between users and the backend. Figure 4 illustrates the system architecture diagram.

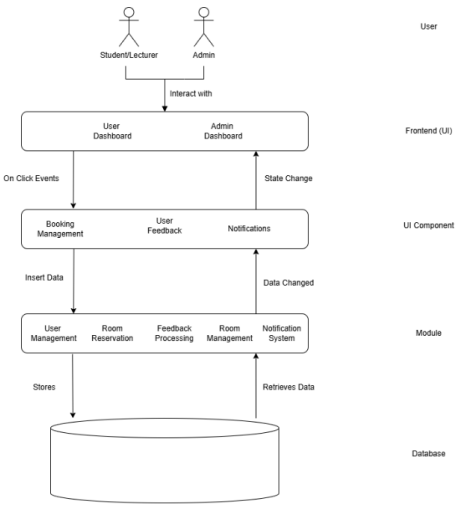

Fig. 4 *System Architecture Diagram*

4.2 Database Design

The data dictionary and schema table are provided in this section to shows the database design of proposed system.

i. `Tbl_user (userID (PK), fullname, email, password,role,telephone, pfp)`
ii. `Tbl_admin (adminID (PK), name, email, password, profile_picture)`
iii. `Tbl_room (roomID (PK), building, room, capacity)`
iv. `Tbl_reservation (reservationID (PK), userID (FK), building, room, date, start_time, end_time, status, purpose)`
v. `Tbl_class (classID (PK), building, room, day, start_time, end_time, subject, end_date, created_at)`
vi. `Tbl_user_feedback (feedbackID (PK), userID (FK), feedback_type, title, description, created_at)`
vii. `Tbl_booking_reports (reportID (PK), userID (FK), reservationID (FK), report_type, description, status, created_at)`

4.3 Interface Design

This section will show the user interface design of the proposed system. The user interface design will display the layout of the system interface that the user will interact with. Figure 5 represents the interface design to be developed.

Fig. 5(a) *Admin Login Interface Design*

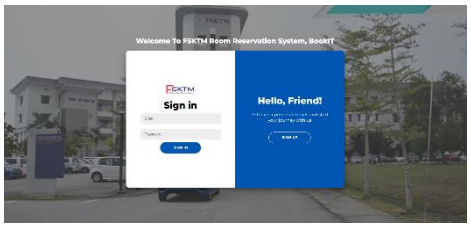

Fig. 5(b) *User Login Interface Design*

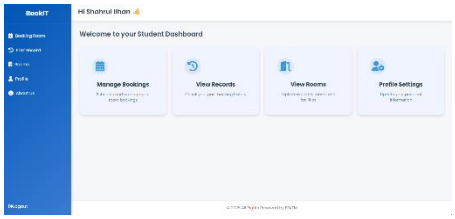

Fig. 5(c) *User Dashboard Interface*

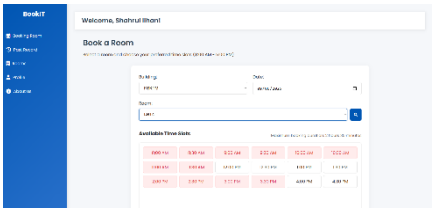

Fig. 5(d) *User Booking Interface*

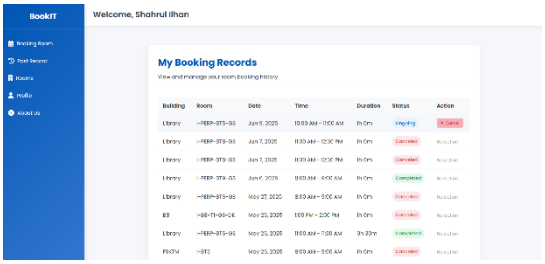

Fig. 5(e) *User Booking Records Interface*

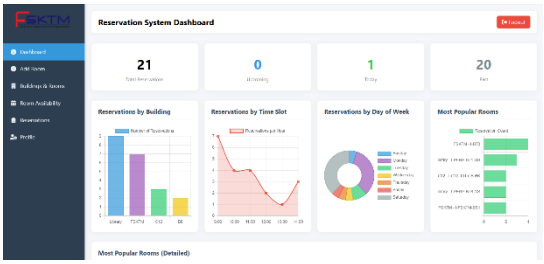

Fig. 5(f) *Admin Dashboard Interface*

5. Implementation

The implementation phase is the process of building the entire Faculty Room Reservation System. For websites, HTML, CSS, and PHP programming languages will be used for coding. Then, the software used is XAMPP. XAMPP is a distributor that provides server and database modules. The integrated development environment software used is Microsoft Visual Studio.

5.1.1 Login and Registration Module Development

Figure 6 shows the server-side coding for the account registration and user login. For the new user, they must register first. They must insert their information like user ID, name, email, password, role and their telephone number. The users (student and lecturer) have to login first to access the system. They need the email and password.

```php
<?php
session_start();
require_once 'database.php';
$message = ''; // Variable to store messages (error or success)

// Handle sign-up
if (isset($_POST['signup'])) {
    $user_id = $_POST['user_id'];
    $fullname = $_POST['fullname'];
    $email = $_POST['email'];
    $password = password_hash($_POST['password'], PASSWORD_DEFAULT);
    $role = $_POST['role'];
    $telephone = $_POST['telephone'];

    // Insert user into the database
    $query = $conn->prepare("INSERT INTO user (user_id, fullname, email, password, role, telephone) VALUES (?, ?, ?, ?, ?, ?)");
    $query->bind_param("ssssss", $user_id, $fullname, $email, $password, $role, $telephone);

    if ($query->execute()) {
        $message = "Account created successfully! You can now log in.";
    } else {
        $message = "Error: " . $conn->error;
    }
}

// Handle login
if (isset($_POST['login'])) {
    $email = $_POST['email'];
    $password = $_POST['password'];

    $query = $conn->prepare("SELECT * FROM user WHERE email = ?");
    $query->bind_param("s", $email);
    $query->execute();
    $result = $query->get_result();

    if ($result->num_rows > 0) {
        $user = $result->fetch_assoc();
```

Fig. 6 *Account Registration and User Login Source Code*

5.1.2 Real Time Availability Module

The Real-Time Availability Module helps users see which rooms are available at any given time. This feature is important because it avoids double bookings and ensures the rooms are used efficiently. It shows current bookings, open time slots, and any blocked times due to maintenance or other reasons. The admin can select the building, room, day, start time, end time, subject, and the end date for the availability setting. This information is used to block off the room for scheduled classes or events. Figure 7 shows the code that processes and stores the availability data into the system so that users can see the latest room status whenever they try to make a booking.

```php
<?php
session_start();
require_once 'database.php';

if (!isset($_SESSION['admin'])) {
    header("Location: admin_login.php");
    exit();
}

$admin = $_SESSION['admin'];
$message = '';
$selected_building = isset($_GET['building']) ? $_GET['building'] : '';
$search_query = isset($_GET['search']) ? $_GET['search'] : '';

// Handle form submission for adding/editing
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    if (isset($_POST['room'])) {
        // Add new unavailable slot
        $building = $_POST['building'];
        $room = $_POST['room'];
        $day = $_POST['day'];
        $start_time = $_POST['start_time'];
        $end_time = $_POST['end_time'];
        $subject = $_POST['subject'];
        $end_date = $_POST['end_date'];

        if (strtotime($end_time) <= strtotime($start_time)) {
            $message = "<div class='alert alert-danger'>End time must be after start time</div>";
        } else {
            $stmt = $conn->prepare("INSERT INTO class (building, room, day, start_time, end_time, subject, end_date) VALUES (?, ?, ?, ?, ?, ?, ?)");
            $stmt->bind_param("sssssss", $building, $room, $day, $start_time, $end_time, $subject, $end_date);

            if ($stmt->execute()) {
                $message = "<div class='alert alert-success'>Unavailable time slot added successfully!</div>";
            } else {
                $message = "<div class='alert alert-danger'>Error: " . $conn->error . "</div>";
```

Fig. 7 *Admin Room Availability Management Source Code*

On the user side, they can only book if the room is available. If a room is already booked or has a class scheduled, it will be marked as unavailable, and the booking option will be disabled. Figure 8 displays the backend code that checks room status in real time. This code ensures that when a user selects a room and time, the system

will check for conflicts and only allow the booking if the room is free. This module plays a key role in keeping the booking process smooth and reliable for all users.

booking_room.php

```php
if (isset($_GET['action'])) {
    switch ($_GET['action']) {
        case 'get_rooms':

        case 'check_availability':
            $building = $_GET['building'];
            $room = $_GET['room'];
            $date = $_GET['date'];
            $day = date('l', strtotime($date));

            // Get reservations
            $res_query = $conn->prepare(
                "SELECT start_time, end_time FROM reservation
                 WHERE building = ? AND room = ? AND date = ? AND status = 'ongoing'"
            );
            $res_query->bind_param("sss", $building, $room, $date);
            $res_query->execute();
            $res_result = $res_query->get_result();
            $res_bookings = $res_result->fetch_all(MYSQLI_ASSOC);

            // Get class schedules
            $class_query = $conn->prepare(
                "SELECT start_time, end_time FROM class
                 WHERE building = ? AND room = ? AND day = ? AND end_date >= ?"
            );
            $class_query->bind_param("ssss", $building, $room, $day, $date);
            $class_query->execute();
            $class_result = $class_query->get_result();
            $class_bookings = $class_result->fetch_all(MYSQLI_ASSOC);

            // Combine results
            $all_bookings = array_merge($res_bookings, $class_bookings);
            echo json_encode($all_bookings);
            exit();
    }
}
```

Fig. 8 *User Booking Source Code*

5.1.3 Automated Booking Module

The Automated Booking Module makes the room reservation process easier and faster for users. When users choose an available time slot and room, the system will automatically confirm the booking without needing admin approval. This helps save time and improves the overall user experience. Figure 9 shows the server-side code that handles this automatic booking process. It checks the selected time slot, confirms that the room is free, and then saves the booking details to the system instantly. Once they select the building, room, date, and time, the booking is automatically confirmed if the room is available. There's no need to wait for manual approval.

```php
booking_room.php
30  // Handle booking submission
31  if (isset($_POST['book_room'])) {
32      $building = $_POST['building'];
33      $room = $_POST['room'];
34      $date = $_POST['date'];
35      $purpose = trim($_POST['purpose']);
36      $selected_slots = $_POST['time_slots'];
37
38      // Get day of week from date
39      $day = date('l', strtotime($date));
40
41      // Convert selected slots to start and end time
42      $slots = explode(',', $selected_slots);
43      $start_time = $slots[0];
44      $end_time = end($slots);
45      $end_time = date('H:i', strtotime($end_time) + 1800);
46
47      // Calculate duration
48      $duration = (strtotime($end_time) - strtotime($start_time)) / 60;
49      $duration_text = '';
50      if ($duration < 60) {
51          $duration_text = "${duration} minutes";
52      } else {
53          $hours = floor($duration / 60);
54          $minutes = $duration % 60;
55          $duration_text = "${hours} hour" . ($hours > 1 ? 's' : '');
56          if ($minutes > 0) {
57              $duration_text .= " ${minutes} minutes";
58          }
59      }
60
61      // Format the time range for display
62      $formatted_time_range = date('g:i A', strtotime($start_time)) . ' - ' . date('g:i A', strtotime($end_time));
63
64      // Check if booking is in the past
65      $current_datetime = time();
66      $booking_datetime = strtotime("$date $start_time");
```

Fig. 9 *User Booking Auto Confirmation Source Code*

5.1.4 Conflict Management Module

The Conflict Management Module is designed to prevent double bookings and avoid scheduling issues. When a user tries to book a room, the system will check if the selected time slot is already taken. If it is, the system will not allow the user to proceed and will show that the slot is unavailable. Figure 10 shows the code that handles this feature. It checks the selected room and time against existing bookings or scheduled classes. If there is a conflict, the system blocks the action and notifies the user. This module is important because it helps keep the booking system accurate and reliable. Users can quickly adjust their selection without confusion or errors.

```php
booking_room.php
31  if (isset($_POST['book_room'])) {
68
69      // Validation checks
70      if ($booking_datetime < $current_datetime) {
71          $message = "You cannot book a room in the past. Please select a future time slot.";
72      } elseif ($current_datetime > $one_hour_before_booking) {
73          $message = "You must book at least 1 hour in advance. For example, to book at " . date('g:i A', $booking_datetime) .
74                     ", you need to book before " . date('g:i A', $one_hour_before_booking);
75      } elseif ($duration > 150) {
76          $message = "Maximum booking duration is 2 hours 30 minutes.";
77      } elseif (empty($purpose)) {
78          $message = "Please enter a purpose for your booking.";
79      } else {
80          // Check against reservations
81          $reservation_check = $conn->prepare(
82              "SELECT * FROM reservation
83               WHERE building = ? AND room = ? AND date = ? AND status = 'ongoing'
84               AND ((start_time < ? AND end_time > ?) OR (start_time < ? AND end_time > ?))"
85          );
86          $reservation_check->bind_param("ssssss", $building, $room, $date, $end_time, $start_time, $end_time, $start_time);
87          $reservation_check->execute();
88          $reservation_result = $reservation_check->get_result();
89
90          // Check against class schedule
91          $class_check = $conn->prepare(
92              "SELECT * FROM class
93               WHERE building = ? AND room = ? AND day = ? AND end_date >= ?
94               AND ((start_time < ? AND end_time > ?) OR (start_time < ? AND end_time > ?))"
95          );
96          $class_check->bind_param("ssssssss", $building, $room, $day, $date, $end_time, $start_time, $end_time, $start_time);
97          $class_check->execute();
98          $class_result = $class_check->get_result();
99
100         if ($reservation_result->num_rows > 0 || $class_result->num_rows > 0) {
101             $message = "The room is not available at the chosen time (already booked or scheduled for class).";
102         } else {
103             // Insert booking
```

Fig. 10 *User Booking Conflict Management Source Code*

5.1.5 Notifications and Alerts Module

The Notifications and Alerts Module helps keep users informed about their bookings. The system sends automatic emails to confirm bookings, remind users of upcoming reservations, and notify them if a booking is cancelled. This makes sure users always stay updated without needing to check the system manually. Figure 11 shows the code that generates and sends this confirmation email once the booking is completed.

```
booking_room.php
31   if (isset($_POST['book_room'])) {
79       } else {
102          } else {
110              if ($insert_query->execute()) {
113                  // Send confirmation email
114                  $mail = new PHPMailer(true);
115                  try {
116                      // Server settings
117                      $mail->isSMTP();
118                      $mail->Host       = 'smtp.gmail.com';
119                      $mail->SMTPAuth   = true;
120                      $mail->Username   = 'shahrulilhan7@gmail.com';
121                      $mail->Password   = 'mpjn ovej aerp sznu';
122                      $mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;
123                      $mail->Port       = 587;
124
125                      // Recipients
126                      $mail->setFrom('shahrulilhan7@gmail.com', 'FSKTM Room Booking');
127                      $mail->addAddress($user_email);
128
129                      // Content
130                      $mail->isHTML(true);
131                      $mail->Subject = 'Room Booking Details';
132
133                      $formatted_date = date('l, F j, Y', strtotime($date));
134                      $formatted_start = date('g:i A', strtotime($start_time));
135                      $formatted_end = date('g:i A', strtotime($end_time));
136
137                      $mail->Body = "
138                      <div style='font-family: Arial, sans-serif; color: #333;'>
139                          <h2 style='color: rgb(0, 0, 0);'>Room Booking Details</h2>
140                          <p>Your room booking has been <strong style='color: green;'>confirmed</strong>. Here are the details:</p>
141
142                          <table style='border-collapse: collapse; width: 100%; max-width: 600px; font-size: 14px;'>
143                              <tr style='background-color: #f2f2f2;'>
144                                  <th style='border: 1px solid #ddd; padding: 10px; text-align: left;'>Field</th>
145                                  <th style='border: 1px solid #ddd; padding: 10px; text-align: left;'>Details</th>
```

Fig. 11 *User Booking Confirmation Source Code*

If a booking is cancelled, the system will also send an email to inform the user. Figure 12 shows the code that handles this notification.

```
user_records.php
27   if (isset($_POST['cancel_booking'])) {
39       if ($result->num_rows > 0) {
49           if ($time_diff >= 60) {  // 60 minutes = 1 hour
57               if ($update_query->execute()) {
58                   // Send cancellation email
59                   $mail = new PHPMailer(true);
60                   try {
61                       // Server settings
62                       $mail->isSMTP();
63                       $mail->Host       = 'smtp.gmail.com';
64                       $mail->SMTPAuth   = true;
65                       $mail->Username   = 'shahrulilhan7@gmail.com';
66                       $mail->Password   = 'mpjn ovej aerp sznu';
67                       $mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;
68                       $mail->Port       = 587;
69
70                       // Recipients
71                       $mail->setFrom('shahrulilhan7@gmail.com', 'FSKTM Room Booking');
72                       $mail->addAddress($user_email);
73
74                       // Content
75                       $mail->isHTML(true);
76                       $mail->Subject = 'Booking Cancellation Confirmation';
77
78                       $formatted_date = date('l, F j, Y', strtotime($booking['date']));
79                       $formatted_start = date('g:i A', strtotime($booking['start_time']));
80                       $formatted_end = date('g:i A', strtotime($booking['end_time']));
81
82                       $duration = (strtotime($booking['end_time']) - strtotime($booking['start_time'])) / 60;
83                       $duration_text = '';
84                       if ($duration < 60) {
85                           $duration_text = "${duration} minutes";
86                       } else {
87                           $hours = floor($duration / 60);
88                           $minutes = $duration % 60;
89                           $duration_text = "${hours} hour" . ($hours > 1 ? 's' : '');
```

Fig. 12 *User Booking Cancellation Source Code*

To help users remember their upcoming bookings, the admin will send reminder emails before the scheduled time. Figure 13 shows the code used to send it. This module is useful to ensure users don't miss their bookings and are aware of any changes, making the whole booking process more reliable and user-friendly.

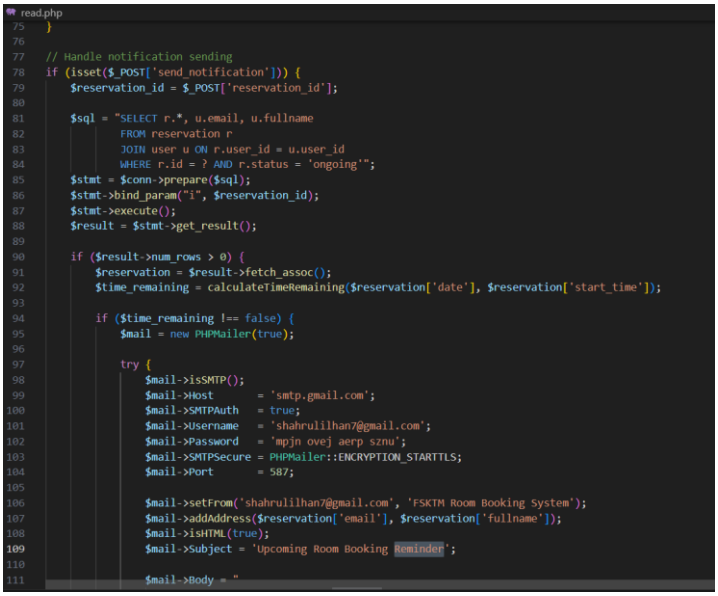

Fig. 13 *User Booking Reminder Source Code*

5.1.6 Reporting Module

The Reporting Module helps the admin monitor and analyse room usage in the system. This module shows useful data such as which buildings are used the most, popular time slots, and overall booking trends. It helps with decision-making and planning by giving a clear view of how the rooms are being used. It shows the admin dashboard with visual summaries, including Reservations by Building, Reservations by Time Slot, Reservations by Day of Week, and Most Popular Rooms. This makes it easier for the admin to understand booking patterns at a glance. Figure 14 will show the code.

Fig. 14 *Admin Report Dashboard Source Code*

5.2 Testing

The testing phase is carried out after the development of the finished system has been completed. This phase aims to evaluate and verify whether the system developed has achieved the project goals or not. If any problems are found during testing, the system should be improved until it reaches the set goals. Testing is divided into two namely functional testing and User Acceptance Testing (UAT)

5.2.1 Functional Testing

Functional testing is the testing of each module and the function of the system to ensure that each module can function as planned. This process is carried out aimed at identifying problems and errors that occur when the system is used. Table 3 shows the functional tests of the system for users and Table 4 shows the functional tests for admin and the results of those tests.

Table 3 *System function testing for users*

Module	Testing	Expected Outcome	Test Result
Registration	Register by entering name, phone number, email address, role and password.	A new user account is successfully registered, and the user is redirected to the login page.	Success
Login	Enter a valid email and password.	User is able to log in and redirected to the main system page.	Success
	Enter an invalid email or password.	Login failure message is displayed, and the user is prompted to enter valid credentials.	Success
Real-Time Availability	Check room availability based on date and time.	System displays available time slots, current bookings, and any scheduled maintenance.	Success
Automated Booking	User makes a booking by selecting an available date and time slot.	System successfully processes the booking and provides instant confirmation.	Success
Conflict Management	User attempts to book a room at a time slot that has already been booked.	System prevents double booking and displays a message indicating the slot is already reserved.	Success
Notifications & Alerts	System sends a notification after a booking is successfully made.	User receives confirmation and reminder notifications via the system or email.	Success

Table 4 *System function testing for admin*

Module	Testing	Expected Outcome	Test Result
Login	Enter a valid email and password.	User is able to log in and redirected to the main system page.	Success
	Enter an invalid email or password.	Login failure message is displayed, and the user is prompted to enter valid credentials.	Success
Real-Time Availability	Manage room availability based on date, time and classroom schedule.	System displays unavailable room that have been set to be unavailable until the end of semester	Success
Reporting Module	Admin views reports by building, time slot, and day of the week.	System displays booking statistics and allows the admin to download reports in PDF format.	Success

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1396-1413

5.2.2 User Acceptance Testing

User Acceptance Testing is the final stage in the process of testing a developed system. At this stage, users are given the opportunity to try out the system and assess whether it works according to their needs and expectations. This phase is very important to help the system developer to review the system programming code and identify any issues that may not have been noticed before.

For this purpose, the Faculty Room Reservation System was tested by 20 respondents through a questionnaire in Google Form. Respondents consisted of students and lecturers.

5.2.3 User Category

In this section, it indicates the category of respondents. Figure 15 shows the feedback from 20 respondents, it was found that the majority of system users consist of students, which is 80%, while the remaining 20% are the lecturers. This percentage shows that students are the main users of the system.

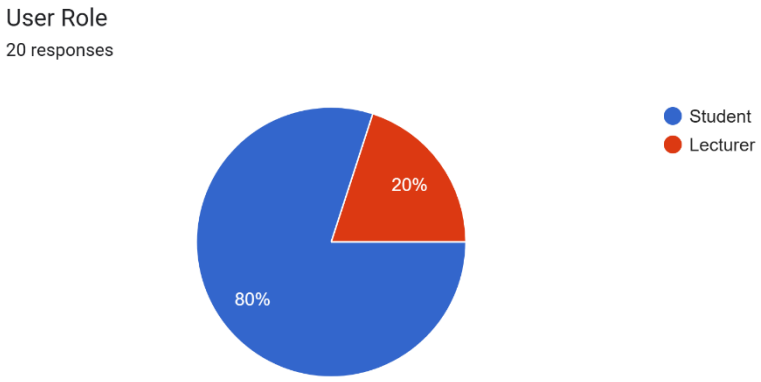

Fig. 15 *System User Category*

5.2.4 User Registration Testing

This section explains the results of the system feature testing, specifically for the user registration function. Figure 16 shows the users' feedback on how easy it was to register an account. The majority of respondents, which is 18 people (90%), rated the registration process as "Very Easy" (score 5). One respondent (5%) gave a score of 4, while another one (5%) gave a score of 3.

In conclusion, the registration process was well received by most users. The high percentage of users who rated it as very easy shows that the registration feature is user-friendly and simple to use. However, there is still a small percentage who felt it could be improved, suggesting that there is some room for enhancement to make the process even better for all users.

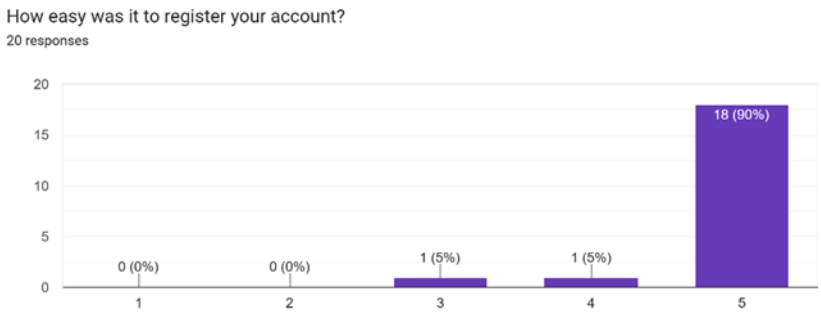

Fig. 16 *System User Registration Testing*

5.2.5 User Login Testing

This part presents the results for the user login feature testing. All 20 respondents (100%) reported that they did not face any issues during the login process. In conclusion, the login feature works smoothly and is reliable, as no problems were experienced by any of the users. Figure 17 shows the chart of user responses for this testing.

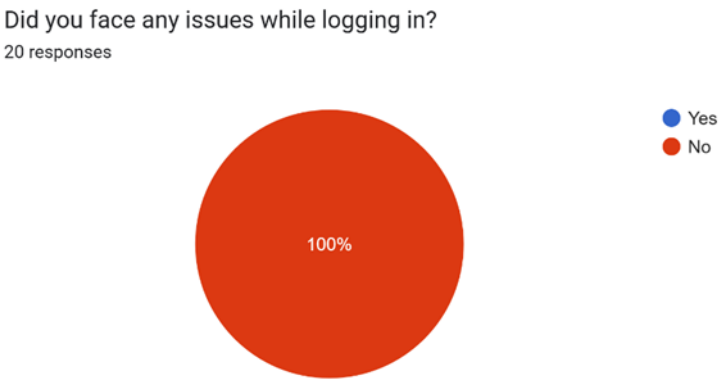

Fig. 17 *System User Login Testing*

5.2.6 System User Dashboard Rating

This section explains the results for the system user dashboard rating. A total of 10 respondents rated the dashboard interface with 5 stars, while another 10 respondents gave it 4 stars. This shows that all users gave positive feedback on the dashboard design. In conclusion, the dashboard interface is well-received by users, with a balanced rating between 4 and 5 stars. This indicates that the design is generally good, but there is still some room for improvement to achieve full user satisfaction. Figure18 presents the chart of user responses for this testing.

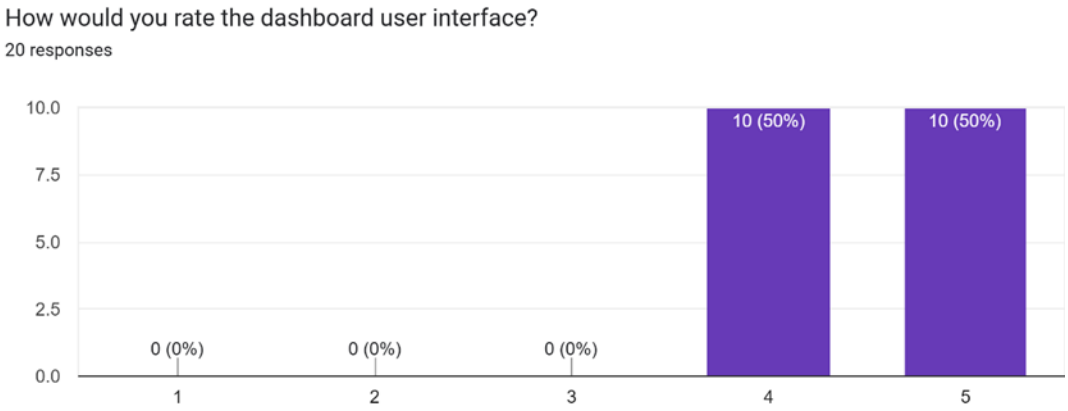

Fig. 18 *System User Dashboard Rating*

5.2.7 User Booking Process Testing

This section presents the results for the user booking process testing. A total of 16 respondents rated the process with a score of 5 (very easy), while 4 respondents gave a score of 4. This shows that users generally found the room booking process to be simple and user-friendly. In conclusion, the booking process received positive feedback from all users, with the majority finding it very easy. This indicates that the feature is effective, although minor improvements could further enhance the user experience. Figure 19 shows the chart of user responses for this testing.

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1396-1413

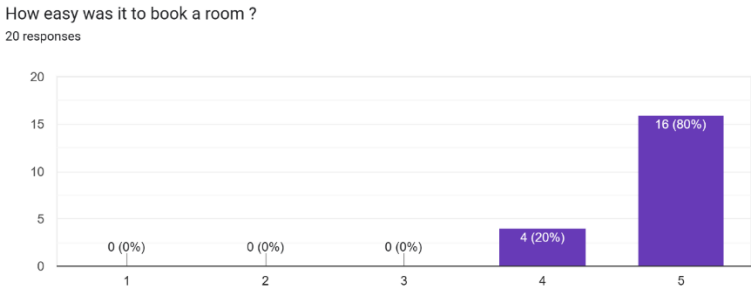

Fig. 19 *System User Booking Process Testing*

5.2.8 User Booking Experience Testing

This section presents the results for the user booking experience testing. All 20 respondents (100%) agreed that it was easy to identify which rooms were unavailable to book. In conclusion, this feature is working effectively, as all users were able to clearly see room availability without any issues. Figure 20 displays the chart of user responses for this testing.

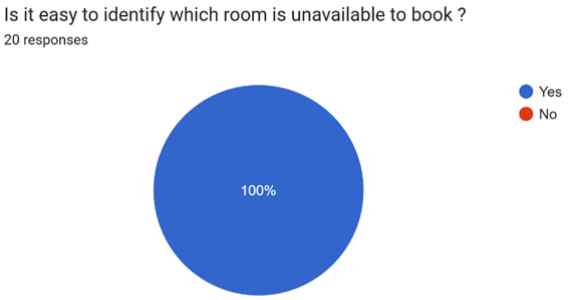

Fig. 20 *System User Booking Experience Testing*

5.2.9 User Booking Cancellation Testing

This section presents the results for the user booking cancellation testing. A total of 16 respondents rated the cancellation process with a score of 5 (very easy), while 4 respondents gave it a score of 4. This shows that most users found it easy to cancel a room booking. In conclusion, the cancellation feature is user-friendly and effective, though there is some room for slight improvement. Figure 21 displays the chart of user responses for this testing.

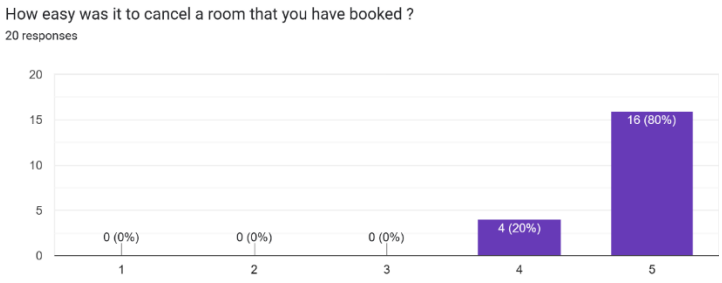

Fig. 21 *System User Booking Cancellation Testing*

5.2.10 User Booking Cancellation Confirmation Testing

This section presents the results for the user booking cancellation confirmation testing. All 20 respondents (100%) confirmed that they received an email notification when their booking was cancelled. In conclusion, the system's email notification for booking cancellations is working well, keeping users informed promptly. Figure 22 shows the chart of user responses for this test.

Fig. 22 *System User Booking Cancellation Confirmation Testing*

5.2.11 User Overall Satisfaction

This section presents the results for the overall user satisfaction with the website. Sixteen respondents gave a rating of 5 stars, while four respondents gave 4 stars. This indicates that most users are highly satisfied with the website. In conclusion, the website has been well received by users, with positive feedback reflecting a good overall experience. Figure 23 shows the chart of user responses for this testing.

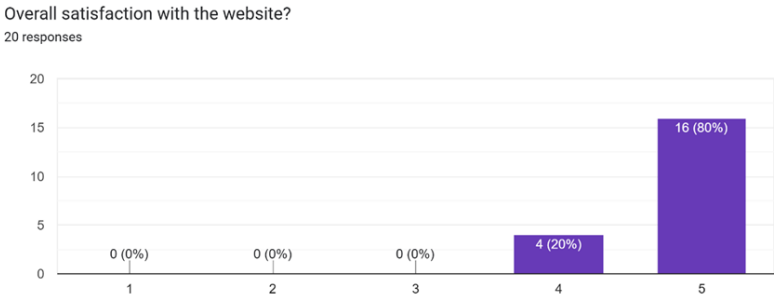

Fig. 23 *User Overall Satisfaction Rating*

6. Conclusion

The FSKTM Room Booking System's development has brought about a number of advantages that improve staff, instructors, and students' booking process's effectiveness, organisation, and user experience. A centralised and secure database for efficient reservation management; tracking of booking history and current statuses; report generation for usage analysis; automated email alerts that enhance communication and minimise misunderstandings; and the ability to book rooms online at any time within the permitted window—all of which eliminate manual paperwork—are some of its main benefits.

Nonetheless, a number of limitations have been noted. The UI is not completely optimised for mobile use, push notifications are not supported, and users are unable to change booking information once they have been submitted. These problems might make the system less accessible and less convenient for users.

Future improvements are suggested to fix these issues. These include creating a mobile-responsive interface, adding in-system notifications, enhancing administrative filtering tools, and enabling users to modify bookings without cancelling them. It is anticipated that putting these changes into practice would improve the system's general functionality, accessibility, and performance, providing all users with a more smooth and effective experience.

Acknowledgement

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

Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *Muhammad Shahrul Ilhan, Nazri Mohd Nawi;* ***data collection:*** *Muhammad Shahrul Ilhan, Nazri Mohd Nawi;* ***analysis and interpretation of results:*** *Muhammad Shahrul Ilhan, Nazri Mohd Nawi;* ***draft manuscript preparation:*** *Muhammad Shahrul Ilhan, Nazri Mohd Nawi. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] German, J. D., Yap, D. C. G., & Binoya, G. O. (2021, April). Design and Development of an Integrated Room Reservation System for Higher Education Institutions. In 2021 IEEE 8th International Conference on Industrial Engineering and Applications (ICIEA) (pp. 231-236). IEEE.

[2] D. Firmansyah, H. Purwanto, T. Wiharko, and B. Purbayanto, "Aplikasi Booking Barbershop Online Berbasis Web menggunakan Framework CodeIgniter," INTERNAL (Information System Journal), vol. 6, no. 2, pp. 146–155, Jan. 2024, doi: 10.32627/internal. v6i2.849.

[3] GeeksforGeeks. (2024, November 12). *Agile prototyping*. GeeksforGeeks. Retrieved from https://www.geeksforgeeks.org/agile-prototyping/