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

AITCS

Vol. 6 No. 2 (2025) 36-56
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Design and Development of Rent Anything Web-based System for UTHM Community

Aina Saffiya Ahmad Shukry[1], Norfaradilla Wahid[1]*

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

*Corresponding Author: faradila@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2025.06.02.003

Article Info

Received: 13 June 2025
Accepted: 3 November 2025
Available online: 30 November 2025

Keywords

Web-based system, rental platform, UTHM community.

Abstract

Rent-Anything is a web-based system specifically for the UTHM community at both Parit Raja and Pagoh campuses which proposes managing all rental activities within the UTHM community such as registration, the listing process, and as well as the renting process. The unavailability of a platform to list items for rent led to the development of a more streamlined system that simplifies the rental process, improves usability, and features a better graphical user interface (GUI). Incorporating such features would make the system more effective, user-friendly, and accessible, improving the overall experience for users and facilitating smoother transactions. The project is developed based on the Prototype Model methodology. As for the implementation, the system is implemented using HTML, CSS, PHP, and SQL as the database. At the end of the project, the Rent-Anything system successfully delivered a seamless and functional platform that enabled the UTHM community at both Parit Raja and Pagoh campuses to manage rental activities efficiently. System testing validated its effectiveness, confirming that the platform met user needs and offered a user-friendly experience.

1. Introduction

The increasing demand for flexible and affordable rental options has led to the development of a web-based rental platform specifically for the Universiti Tun Hussein Onn Malaysia (UTHM) community. Renting is becoming more popular, especially among students who seek economical and adaptable solutions for their daily needs [1]. This platform aims to simplify the rental process by combining various rental options into one easy-to-use interface. Factors like how long an item is needed and personal preferences, such as convenience and ownership desires, influence the decision to rent or buy [2]. Currently, UTHM lacks a centralized rental system, making it hard for users to access and manage rental resources efficiently. The Rent-Anything platform allows users from both Parit Raja and Pagoh campuses to upload and rent items, giving the entire community access to a wide range of rental options in one place.

1.1 Problem Statement

This project addresses several key issues. Students often struggle to find affordable short-term resources for academic, social, or extracurricular needs. Without a rental platform, they are forced to buy items unnecessarily, which adds to waste and environmental problems. The proposed system offers a cost-effective, sustainable, and efficient solution by making it easier to access items for temporary use.

1.2 Objectives

The objectives of this project are to analyse and design a web-based rental platform for UTHM community that integrates various rental needs into one convenient location, making it easier for users to find what they need without switching between multiple sites. Next, is to develop a user-friendly, secure, and accessible platform that addresses the core challenges in the rental activities, providing a seamless experience for both renters and item owners. Lastly, to evaluate the platform's functionalities and determine whether the developed system meets all the requirements and expectations of the target user.

1.3 Scope

The project scope includes develop a web-based rental system for UTHM students and staff at Parit Raja and Pagoh campuses. Users can register and log in using email or Google, manage their profiles and payment details, list or rent items, and track their rental history. The platform includes features such as advanced search by category and location, Wishlist and cart functions, flexible booking with calendar availability, secure payments with deposit and refund options, and in-app messaging for direct communication. An administrator dashboard will support the management of categories, user messages, and content to ensure smooth operations.

1.4 Report Organization

This paper has five sections. Section 1 introduces the topic. Section 2 reviews past studies and compares other systems with this one. Section 3 explains how the project was done step by step. Section 4 shows the results and talks about any problems faced. Section 5 ends the paper with a summary and suggestions to improve the system.

2. Related Works

Rental systems let people use items for a short time, helping them save money and reduce waste. Research shows that rental and sharing models can support sustainable lifestyles when designed to fit user needs [3]. Many online platforms today allow easy access to items like cars and electronics with features such as real-time booking and secure payments [4]. However, most of these platforms are not designed for university use and lack features like user checks, messaging, and proper management. At universities, students often rent items informally, which can be hard to manage. The Rent-Anything system is built for the UTHM community to solve these issues. It includes features like messaging, smart search, and admin tools. Some parts of the system have been tested, and more updates like image sharing are planned.

2.1 A Study on Similar Systems

To develop the Rent-Anything system tailored for the UTHM community, a comprehensive literature review was conducted to evaluate existing rental platforms and identify gaps that the proposed system aims to fill. Various studies on current rental systems were analyzed to ensure that the design and features of the Rent-Anything system effectively meet the unique rental needs of students and staff within UTHM's Parit Raja and Pagoh campuses. A detailed comparison was made against three notable existing platforms SmartRental [5], Just Rent It! Malaysia [6], and Carousell [7] to ensure that the proposed system offers distinct advantages and improvements over these solutions

SmartRental operates as a subscription-based service primarily focused on flexible IT product rentals [5]. It provides useful features such as Profile Management, Product Listings, clear Pricing information, Shopping Cart functionality, user Ratings, and a Wish List. However, it falls short in critical areas including Advanced Search and Filtering options, In-App Messaging for user communication, Notifications for timely updates, and Rental History tracking, which limits the overall user experience and convenience.

Just Rent It! Malaysia is a versatile platform offering a wide range of rental items across multiple categories [6]. While it supports Product Listings, Pricing transparency, and Shopping Cart functions, it lacks several essential features that contribute to ease of use and security. Notably, it does not support Profile Management, Advanced Search & Filters, In-App Messaging, Secure Payment Processing, Notifications, Ratings & Reviews, Rental History tracking, an Administrator Dashboard for platform management, Wish List functionality, or Refund handling. These omissions reduce its user-friendliness and reliability compared to more comprehensive systems.

Carousell primarily serves as a marketplace for buying and selling second-hand items but also includes rental options [7]. It excels in providing Advanced Search & Filters, In-App Messaging between users, Ratings & Reviews, Shopping Cart capabilities, and Wish List features. However, Carousell lacks clear Pricing and Rental Details, Rental History tracking, and full Profile Management, making it less tailored for organized rental management.

In comparison to SmartRental, Just Rent It! Malaysia, and Carousell, the proposed Rent-Anything system provides a more complete and tailored rental solution for the UTHM community, as shown in Table 1. It overcomes SmartRental's limitations by including advanced search, messaging, notifications, and rental history tracking. Unlike Just Rent It! Malaysia, it offers profile management, secure payments, ratings, and administrative tools to

enhance usability and reliability. While Carousell has useful features, it lacks clear pricing, rental history, and full profile management, which Rent-Anything addresses. Additional features like refund handling and an administrator dashboard make Rent-Anything a secure, user-friendly platform specifically designed for UTHM's Parit Raja and Pagoh campuses.

Table 1 *Comparison of existing systems with proposed system*

System / Feature	SmartRental [5]	Just Rent It! Malaysia [6]	Carousell [7]	Proposed System (Rent-Anything System)
Profile Management	√	X	√	√
Product Listings	√	√	√	√
Detailed Item Descriptions	√	√	√	√
Pricing and Rental Information	√	√	X	√
Advanced Search & Filters	X	X	√	√
In-App messaging	X	X	√	√
Secure Payment Processing	√	X	√	√
Shopping Cart	√	√	√	√
Rating & Reviews	√	X	√	√
Notifications	X	X	√	√
Rental History Tracking	X	X	X	√
Administrator Dashboard	X	X	√	√
Wish List Feature	√	X	√	√
Refund	X	X	√	√

2.2 Technology Used

The Rent-Anything platform is built using reliable technologies to ensure it runs smoothly, is easy to use, and can grow as needed. The backend uses PHP (Hypertext Preprocessor), a well-known server-side language that helps create dynamic websites and works well with MySQL, an open-source database used to store and manage data. For secure and automated payments, the platform uses ToyyibPay, a Shariah-compliant payment gateway widely used in Malaysia. It supports FPX (Financial Process Exchange)or online banking, making payment collection and transfers simple and safe [8]. Together, these tools give the platform a strong, efficient, and user-friendly system.

3. Methodology

3.1 Prototype Model

The prototype model is a software development approach that builds an early version of the system to understand requirements and test design ideas. It includes five main phases, i.e., planning, analysis, design, implementation, and testing. Figure 1 shows the Prototype Model used for the Rent-Anything System. Prototyping helps confirm that the system works as expected and allows changes based on user feedback [9]. This step-by-step process helps identify improvements early, reduce risks, and ensure the final system meets user needs better.

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

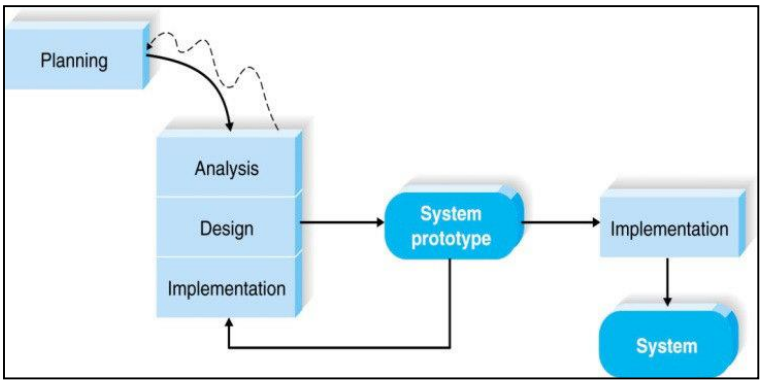


Fig. 1 *Prototype Model [10]*

The Planning Phase initiates the project by laying its foundation. Since UTHM lacks an existing rental platform, this phase focuses on creating one to address the gap. The project's scope, goals, and expected outcomes are clearly defined, and a Gantt chart (Appendix A) outlines the timeline and guides progress.

Next is the Analysis Phase, which the Analysis Phase identifies both functional and non-functional requirements to ensure the system meets user needs. Feedback from questionnaires distributed to the UTHM community helps define key features such as reservation management, payment systems, and inventory tracking, while also addressing performance, security, and scalability. Diagrams like DFDs, flowcharts, and ERDs are used to visualize the system's flow and structure.

During the Design Phase, the system's overall structure is planned based on the analysis, including the design of the architecture, user interface, and database. The goal is to build a reliable, scalable database alongside an easy-to-use interface. Wireframes created in Figma allow stakeholders to review and improve the design.

Next, in the Implementation Phase, a prototype of the system is developed according to the design. This prototype undergoes testing and refinement based on user feedback until it fully meets the required specifications. Once the prototype satisfies all requirements, the system is fully implemented. Finally, the project moves into the System Testing Phase, where the complete system is rigorously tested to confirm that it functions correctly and meets quality standards. This ensures the platform is reliable, user-friendly, and ready for deployment.

3.2 Summary of Activities

The proposed system development is structured into key phases: Planning sets project goals and identifies rental issues. Analysis gathers user needs via surveys and defines system requirements. Design plans system layout and database structure. Implementation develops the system, integrates with the database, and provides ongoing updates. System Testing ensures error-free functionality and readiness for launch, ensuring effectiveness and user-friendliness throughout.

Table 2 *System Development Workflow of the Developed System*

Phase	Task	Deliverables
Planning	▪ Identify current rental problem in UTHM, project scope, objectives, project significance and expected results. ▪ Research on rental topics.	▪ Project proposal tailored to UTHM context. ▪ Development timeline in the form of a Gantt chart.
Analysis	▪ Construct and distribute questionnaire ▪ Gather information about user requirements from UTHM community. ▪ Comparison between existing system and proposed system. ▪ Requirement gathering. ▪ Identify functional and non-functional requirements. ▪ Identify relationships, between processes, data and entities. ▪ Update the prototype based on future user feedback and system needs.	▪ Questionnaire results . ▪ User requirements. ▪ Comparison report. ▪ Requirements document. ▪ Functional and non-functional requirements. ▪ Flowchart for all users. ▪ Context Diagram. ▪ Data Flow Diagram (DFD) ▪ Entity Relationship Diagram (ERD).

Table 2 *(cont)*

Design	▪ Design wireframe. ▪ Database design. ▪ The prototype is finalized.	▪ Wireframe ▪ Database schema design. ▪ Data Dictionary.
Implementation	▪ Connect system with database. ▪ The Rent Anything is developed. ▪ Provide ongoing maintenance and support for the system. ▪ Update the prototype based on future user feedback and system needs.	▪ Fully developed proposed system (Rent Anything for UTHM Community). ▪ System update documentation.
System Testing	▪ System testing with UTHM community. ▪ Identity areas for improvement. ▪ Finalizing the latest development. ▪ Provide ongoing maintenance and support for the system.	▪ Correcting and improving defects. ▪ The most recent system has been completed and is set for release.

4. Result and Discussion

4.1 System Analysis

System analysis plays a key role in shaping the structure and direction of the proposed system. It begins with a Context Diagram that displays the interaction between external entities and the system, followed by a Data Flow Diagram (DFD) that provides a deeper view of how data moves within the system. These diagrams help in understanding the system's needs and creating effective workflows. This section begins by stating both functional and non-functional requirements. Table 3 presents the functional requirements, which describe the tasks, actions, or features the system must carry out to fulfill user needs and objectives.

Table 3 *Functional requirements of the proposed system*

No	Module	Submodules	Functional Requirements
1	User Management	User Login and Register	▪ The system should allow users to register via email, phone, or social media (Google) with secure password management and recovery. ▪ The system should allow users to log in using their registered credentials via email and password. ▪ The system should provide mechanisms for users to recover forgotten passwords securely
		Profile Management	▪ The system should allow users to create profiles, including personal details, payment methods, and account settings. ▪ The system should allow users to edit and manage profiles, including updating personal details, payment methods, and account settings. ▪ The system should allow users to create profiles, including personal details, payment methods, and account settings. ▪ The system should allow users to edit and manage profiles, including updating personal details, payment methods, and account settings. ▪ The system should allow users to delete their profiles.
2	Advanced Search & Filter		▪ The system should provide advanced search functionality with filters by category and locations.

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

Table 3 *(cont.)*

3	Rental	▪ The system should allow the user for booking requests, calendar availability and flexible pick-up/return date selection. ▪ The system should allow users to track and display rental history and their item listing. ▪ The system should allow user to add the items to cart or Wishlist. ▪ The system should allow the user to list items for rent. ▪ The system should allow user receive notifications.
4	Payment Processing	▪ The system should integrate with multiple payment gateways to support various payment methods. ▪ The system should handle security deposits. ▪ The system should allow users to request for a refund.
5	In-App Communication	▪ The system should provide an in-app messaging system for direct communication between renters and owners. ▪ The system should allow users to chat to ask questions, discuss rental details, and clarify any concerns about the item or transaction.
6	Review	▪ The system should allow user to leave a review.

Based on Table 4 below shows the non-functional requirements, which define the quality characteristics, performance criteria, or limitations a system must meet for efficient and satisfactory operation. These include scalability, security, reliability, usability, performance, and maintainability, ensuring the system meets user expectations beyond basic functionality.

Table 4 *Non-functional requirements of the proposed system*

No	Modules	Non -Functional Requirements
1	Security & Compliance	▪ The system should allow only valid users to log in into their accounts using their username and password. ▪ The system should use encryption to protect sensitive data. ▪ The system should provide mechanisms for users to recover forgotten passwords securely.
2	Performance	▪ Ensure the system operates efficiently, handling many users and transactions simultaneously. ▪ The system should be able to respond fast and perform the desired output to users. ▪ The system should be able to use anytime.
3	Operational	▪ The system should be user-friendly ▪ The system should be run across all devices

4.1.1 Flowchart

A flowchart provides a visual guide to the system's processes. Figure 2 presents the user flowchart for the Rent-Anything System, showing how users interact with the system. The process begins with account registration or login, including an option to reset the password if needed. Once logged in, users arrive at the homepage, where they can perform actions such as renting items, listing products, checking rental history, canceling rentals, updating their profile, or viewing the "About" section. Each step follows a logical order, ending either at the homepage or with the user logging out.

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

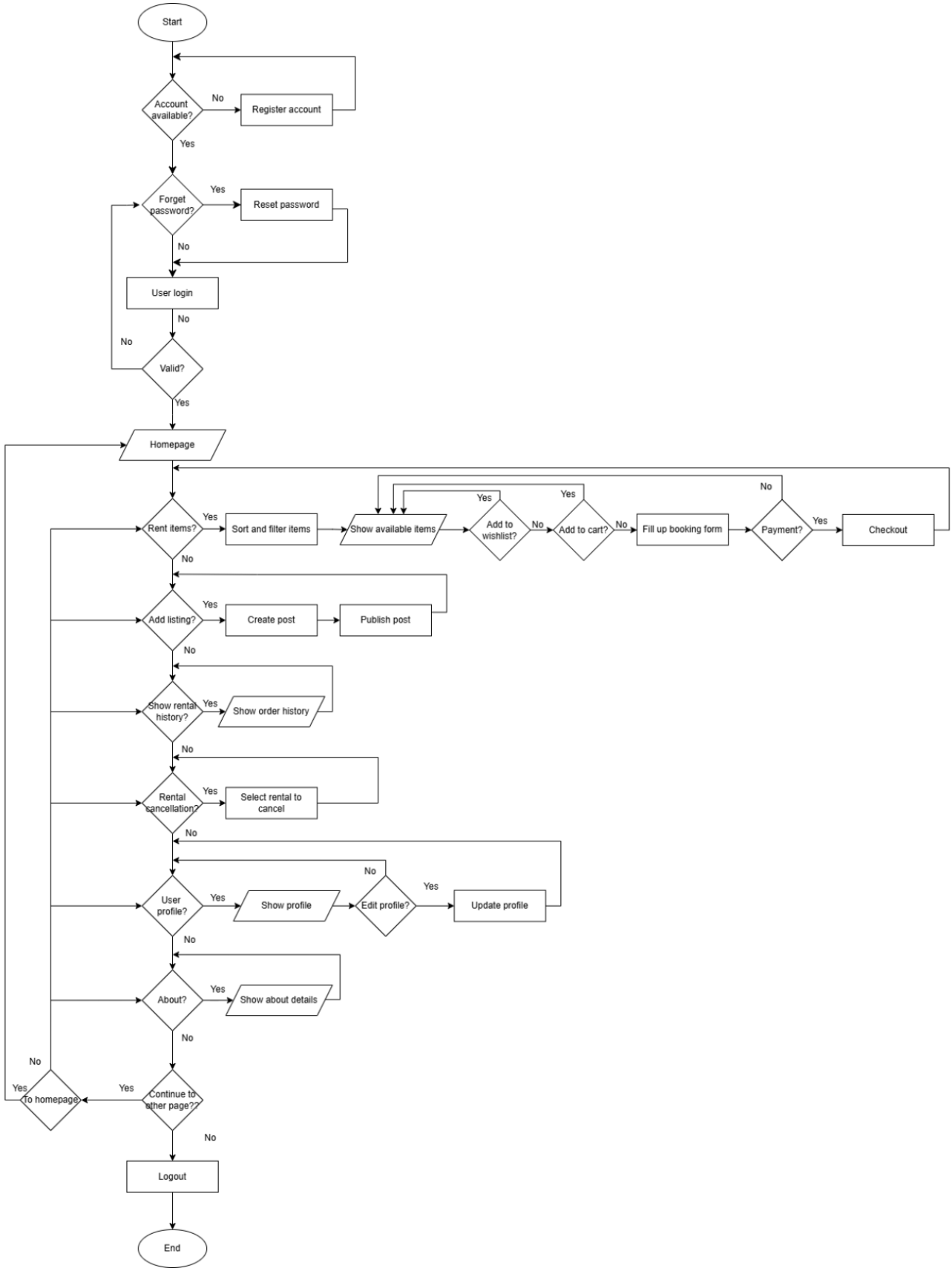


Fig. 2 User *flowchart of the Rent-Anything System*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

4.1.2 Context Diagram and Data Flow Diagram (DFD)

The Context Diagram shows how external entities interact with the system. It gives an overview of how the entire process works. In the Rent-Anything System, users can register, log in or update their profile, view the about page, browse or list items for rent, select item categories, complete rentals, make payments, check booking statuses, access invoices, view rental history, and leave reviews. Administrators manage rental posts, update the about page, view payments, monitor and remove inappropriate content, and oversee rental items. The system acts as the central hub, ensuring smooth communication and transactions between users and administrators.

A Data Flow Diagram (DFD) is a visual representation of how data flows within a system. With its hierarchical structure, a DFD can present various levels of system detail, making it a valuable tool for system analysis and design. The main components include external entities, processes, data flows, and data stores. There are seven processes in Level 0 DFD, which are register, login, manage about, manage item for rental, rent item, make payment, leave feedback and reviews. The data stores in this system include account data store, about data store, item data store, rental data store, invoice data store, payment data store and feedback data store. There are two entity which are user and administrator. The Level Data Flow Diagram Level 0 is provided in Appendix B.

The DFD Level 1 in Appendix C highlights the main processes involved when users interact with the system to list items and rent items. For renting an item, users search items, select an item, add to cart or Wishlist and complete the rental process by providing necessary payment details, with the system storing rental and payment information in their respective data stores. Meanwhile, for listing an item, users must first log in, after which they can provide item details, which are stored in the item data store.

Moreover, DFD Level 2 explains the password management process. Users or administrators begin by entering their email to reset the password. The system verifies account information, allows password changes, stores the new details securely, and sends a notification ensuring secure and verified password management.

4.1.3 Entity Relationship Diagram

The Entity Relationship Diagram (ERD) is intended to help in understanding the fundamental data and information that will be stored in the database. An Entity-Relationship Diagram (ERD) outlines the structure of a system designed to represents the entities and the relationships among the tables in a database. It plays a crucial role in designing a well-structured database by providing a clear framework for organizing the overall system. The ERD for this system includes various tables such as user, administrator, about_page, bookings, cart, feedback_reviews, invoice, items, message, payment, payment_methods, rental, and wishlist. Each table is designed with specific attributes and functionalities, and they are interconnected through defined relationships. The Entity Relationship Diagram (ERD) of the proposed system is shown in Appendix D .

4.2 System Design

After gathering requirements during the analysis phase, the system moves on to the design phase, during which the database and user interface designs are developed. The design of the database is critical for the effective organization, storage, and retrieval of data, serving as the foundation for the system's functionality. Likewise, the design of the user interface is vital for fostering an intuitive and user-friendly experience, allowing users to navigate the system with ease. These designs not only set a clear and operational framework for the system but also significantly contribute to streamlining the development process, ensuring that the final product effectively addresses user needs.

4.2.1 Database Design

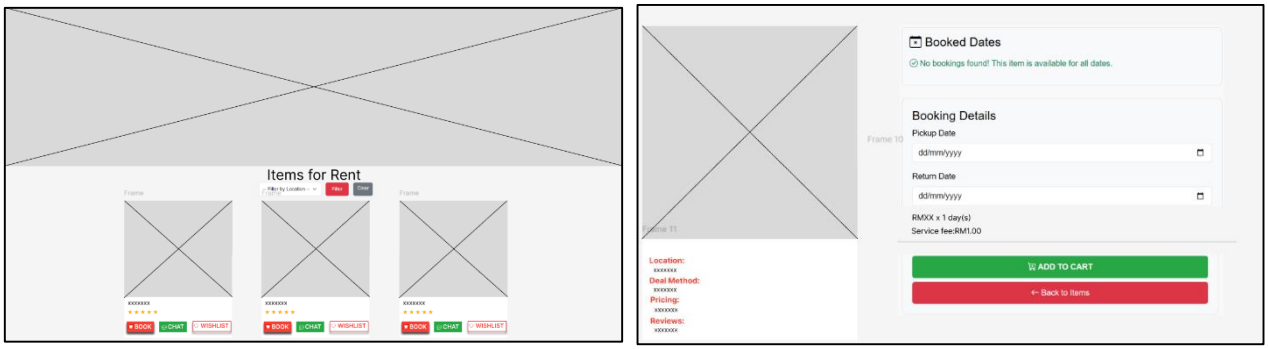
Database design involves organizing, storing, and managing data efficiently to ensure accuracy, performance, and scalability. It includes creating a schema that defines how data tables relate to each other and meets the system's functional needs. Key elements like normalization, indexing, and query optimization help improve data handling and retrieval. This approach also explains why certain design choices are made, offering deeper insight into how the database works. A total of 17 database tables were developed for the system. Due to limitations in page length, the full set of tables is not displayed here. Instead, a single table is shown in the Appendix E to exemplify the overall design and structure.

4.2.2 User Interface Design

User interfaces (UI) are designed to help users interact with a system in a clear, easy, and engaging way. A good UI makes the system simple to use, easy to navigate, and visually appealing, which improves the overall user experience. Wireframes play a key role in UI design by showing a basic layout of the interface. They highlight where elements like buttons, menus, and forms will go, helping designers focus on function and flow before adding

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

detailed styles like colors and fonts. This ensures the design supports user needs before moving to the final design stage.



(a) **(b)**

Fig. 3 *User interface design (a) item listing; (b) booking page*

5.0 System Implementation and Testing

This section presents the implementation and testing of the web-based application. Although the system consists of several modules, only the rental feature is highlighted here due to page constraints. It focuses on key user functions within the rental process and showcases testing conducted on this feature to demonstrate the system's primary functionality.

5.1 Implementation

The primary goal of the implementation phase is to ensure that the Rent-Anything System for the UTHM community is developed based on the specifications defined during the system analysis and design phase. Figure 4 shows the homepage of the Rent Anything for UTHM community.

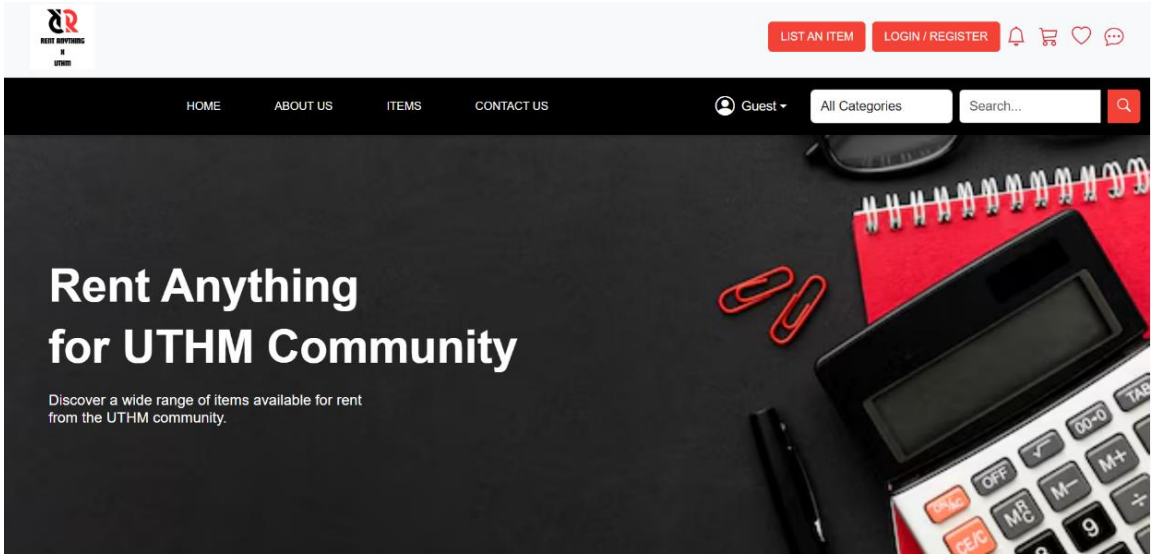


Fig. 4 *Homepage interface*

Figure 5 illustrates the registration form used in the Rent Anything system, where users enter their personal information to create an account.

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

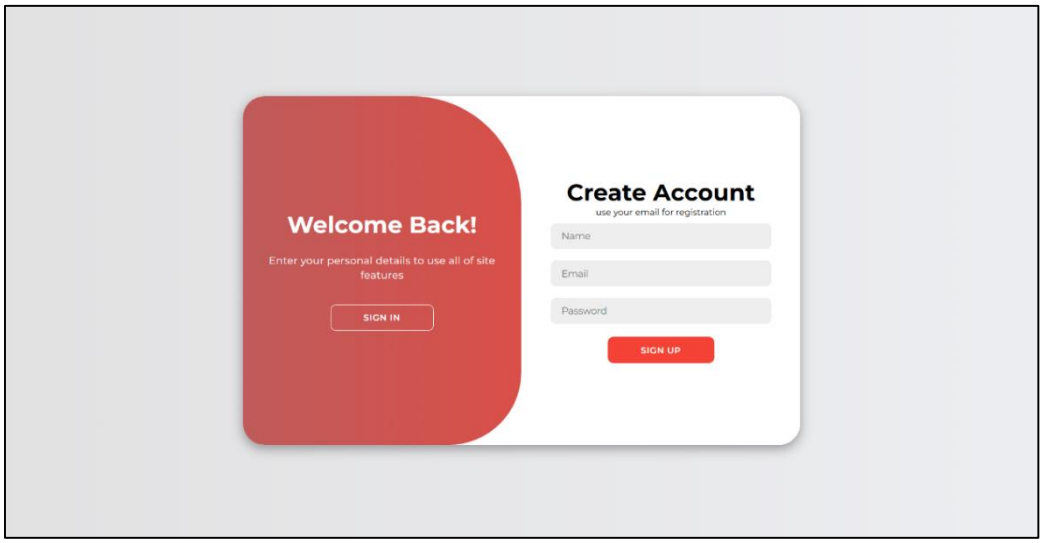


Fig. 5 *User registration interface*

Figure 6 shows the part of the code responsible for handling the registration process, including email validation, duplicate checks, and securely saving the user's information

```php
<?php
require 'db_connection.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = $_POST['name'];
    $email = $_POST['email'];
    $password = password_hash(password: $_POST['password'], algo: PASSWORD_DEFAULT);

    $stmt = $conn->prepare(query: "INSERT INTO users (fname, email, password) VALUES (?, ?, ?)");
    $stmt->bind_param(types: "sss", var: &$name, vars: &$email, $password);

    if ($stmt->execute()) {
        header(header: 'Location: loginpage.php');
        exit;
    } else {
        echo "Error: " . $stmt->error;
    }
}
?>
```

Fig. 6 *Code segment of user registration*

To access the Rent Anything platform, users need to provide their email and password. Figure 7 shows the login interface of the Rent Anything system, where users enter their email and password, which are verified with the database to grant access or prompt a retry if incorrect.

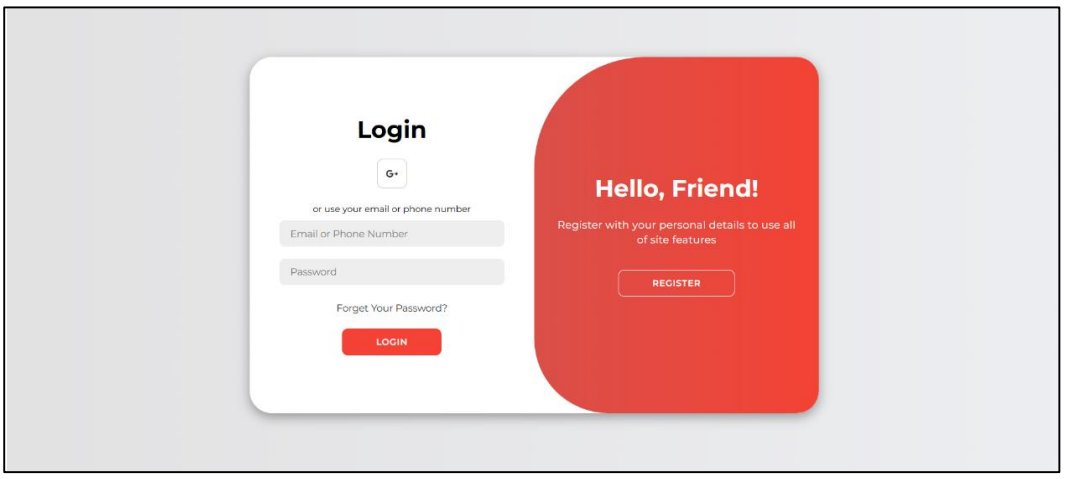


Fig. 7 *User login interface*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

Figure 8 displays the code that manages the login process. This section includes the logic for verifying user input checking for matching records in the database, and determining whether to log the user in or prompt for correction.

```php
// Check if the user exists in the database by email
$sql = "SELECT * FROM users WHERE email = ?";
$stmt = $conn->prepare(query: $sql);
$stmt->bind_param(types: "s", var: &$email);
$stmt->execute();
$result = $stmt->get_result();

if ($result->num_rows > 0) {
    // User exists, fetch the data
    $user = $result->fetch_assoc();
    $_SESSION['user_id'] = $user['user_id'];
    $_SESSION['fname'] = $user['fname'];
    $_SESSION['lname'] = $user['lname'];
    $_SESSION['email'] = $user['email'];
} else {
    // User does not exist, insert a new record
    $password = password_hash(password: uniqid(), algo: PASSWORD_DEFAULT); // Generate a random password
    $phone_num = null; // Default null as not provided by Google
    $gender = null; // Default null as not provided by Google
    $campus = null; // Default null as not provided by Google

    $insert_sql = "INSERT INTO users (fname, lname, email, password, phone_num, gender, campus) VALUES (?, ?, ?, ?, ?, ?, ?)";
    $insert_stmt = $conn->prepare(query: $insert_sql);
    $insert_stmt->bind_param(types: "sssssss", var: &$first_name, vars: &$last_name, $email, $password,
     $phone_num, $gender, $campus);
```

Fig. 8 *Code segment of user login*

Figure 9 displays the "List an Item" page, which allows users to contribute to the platform by offering items for rent and supporting the community. Users are required to provide the item name, description, price, and preferred deal method. They also need to upload an image, select a category assigned by the administrator, and choose the relevant campus location. This helps ensure that each item listing is informative and easy for renters to browse.

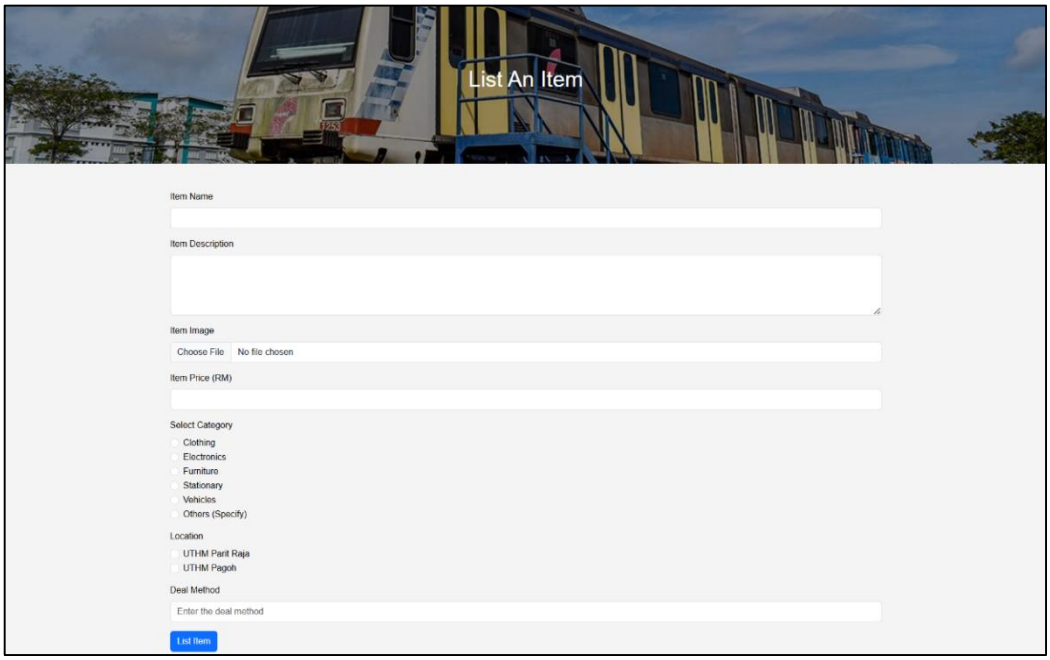


Fig. 9 *List an Item page interface*

Figure 10 shows the code section that manages new item submissions to the platform. It verifies the details entered by users including the item name, description, price, deal method, uploaded image, chosen category, and campus location to ensure all necessary information is complete and accurate. After successful validation, the code saves the item data into the database, allowing the item to be listed and accessible for rental. This process ensures that listings are reliable and helps expand the platform's offerings.

```
// Insert item into the Items table (no category_name)
$stmt = $conn->prepare(query: "INSERT INTO items (item_id, user_id, name, description, price, image, category_id,
locations, deal_method) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
$stmt->bind_param(types: "ssssdssss", var: &$item_id, vars: &$user_id, $itemName, $itemDescription, $itemPrice, $uploadPath,
$category_id, $locationsJson, $dealMethod);

if ($stmt->execute()) {
    $successMessage = "Item successfully listed!";
} else {
    $errorMessage = "Failed to list item. Please try again.";
}
```

Fig. 10 *Code segment of list an item page*

Next, the booking page offers an easy-to-use interface where users can choose pick-up and return dates for an item through a live calendar, which shows the item's current availability. To assist users in making informed choices, the page clearly presents essential details about the item, such as its name, a comprehensive description, the campus location where it is available, and the selected deal method. This information helps users understand what they are renting and where the transaction will take place, making the booking process clear and straightforward.

Furthermore, the system prevents double bookings by verifying whether the selected dates overlap with any existing reservations. If a user tries to book dates that are already taken, the system will block the booking and display an error message informing the user that those dates have been reserved by others. This ensures no scheduling conflicts occur and provides fair access to items on the platform. Overall, the booking page combines detailed item information with a secure date selection process to create a smooth and reliable rental experience. Figure 11 shows the booking page interface and its features.

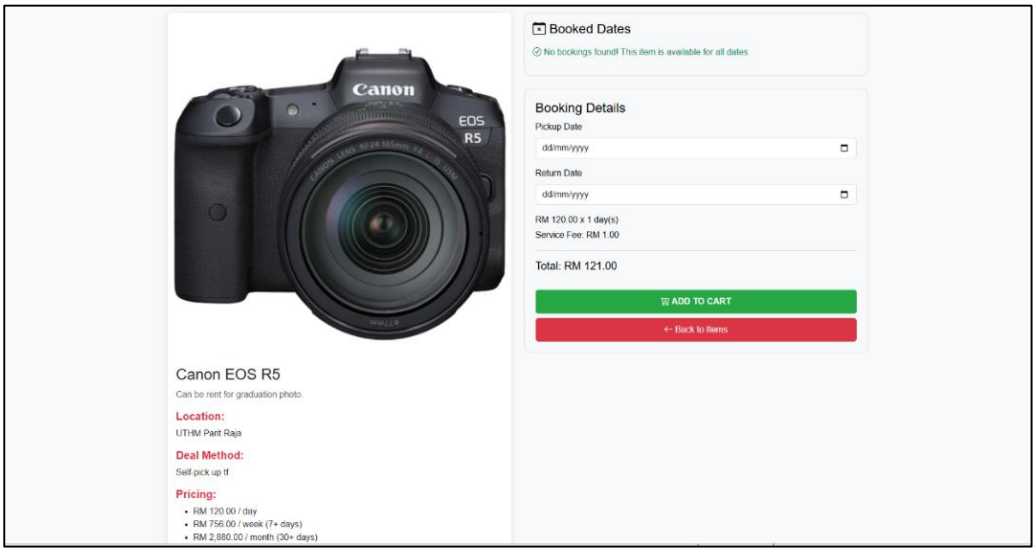


Fig. 11 *Booking page interface*

The platform integrates with various payment gateways, giving users multiple ways to pay for their transactions quickly and securely. This setup helps ensure a hassle-free payment experience. Figure 12 illustrates the payment interface, where users can pick their preferred payment method and complete their payment safely.

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

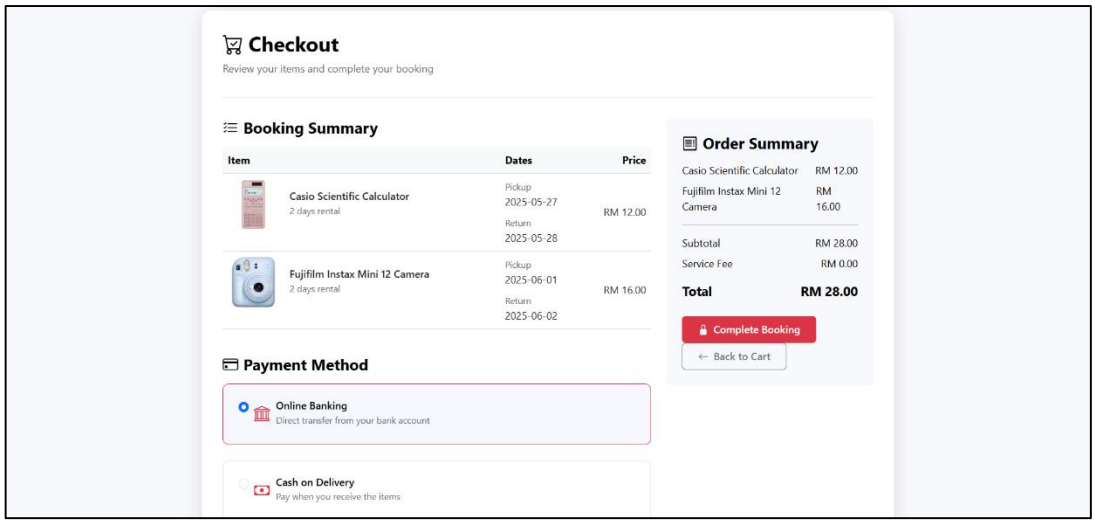


Fig. 12 *Payment checkout interface*

Figure 13 shows the code responsible for handling payments made through online banking and cash on delivery (COD). Once the payment is completed successfully, the system saves the transaction information in the booking history for record-keeping and tracking purposes.

```
    // Redirect to payment gateway
    header(header: "Location: https://toyyibpay.com/" . $billcode);
    exit;
  } else {
    $errorMsg = $obj[0]['msg'] ?? 'Unknown error occurred';
    error_log(message: "ToyyibPay Bill Creation Failed: " . $errorMsg . " | Full Response: " . $result);
    $_SESSION['error_message'] = "Failed to initialize payment gateway: " . $errorMsg;
    die($_SESSION['error_message']);
    header(header: "Location: checkout.php");
    exit;
  }
}
// For COD payment method
else {
  $success_message = 'Booking and payment recorded successfully!';
```

Fig. 13 *Payment checkout interface*

Users can view the status and history of their refund requests through a dedicated section, as shown in Figure 14, ensuring transparency and keeping them informed throughout the process.

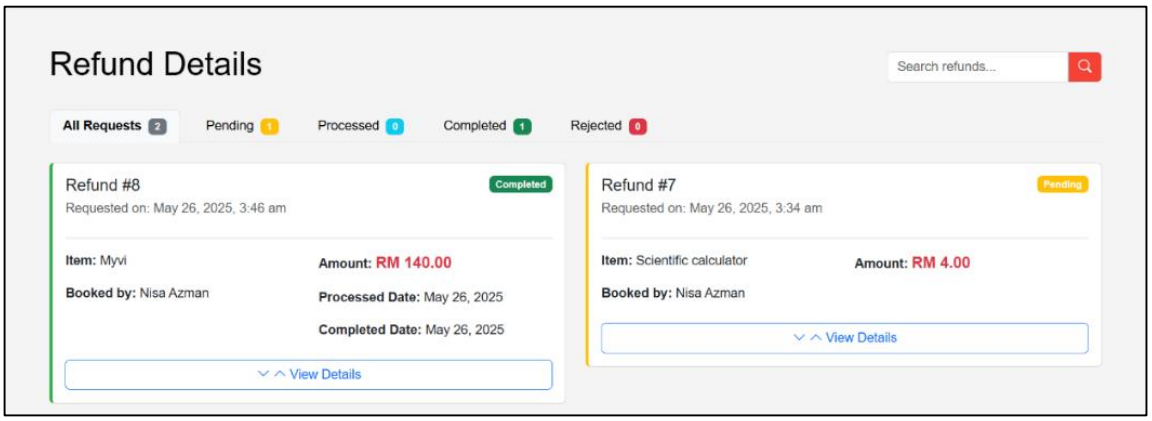


Fig. 14 *Refund details interface*

Figure 52 displays the code segment that fetches the refund information from the database for users who request a refund.

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

```
$query = "SELECT r.*, u.fname, u.lname, i.name AS item_name, b.pickup_date, b.return_date,
          i.user_id AS owner_id, i.price, DATEDIFF(b.return_date, b.pickup_date) AS rental_days
          FROM refunds r
          JOIN users u ON r.user_id = u.user_id
          JOIN items i ON r.item_id = i.item_id
          JOIN bookings b ON r.booking_id = b.booking_id
          WHERE r.user_id = ?";
```

Fig. 15 *Refund details interface*

5.2 System Testing

System testing is a vital stage in the software development process, aimed at verifying that the application works as intended. For the "Rent Anything for UTHM Community" system, two types of testing were conducted: system testing and user acceptance testing (UAT). System testing involved functional testing of six key modules: User Management, Advanced Search Filter, Rental, Payment Processing, In-App Communication, and Review Module. Each module was tested using structured test cases, all of which passed successfully, confirming the system met its functional requirements. Next, UAT was carried out with 16 respondents from the UTHM community, consisting of 15 users and 1 administrator. This phase assessed the interface, functionality, user experience, and system performance. Feedback, collected through questionnaires, showed a high level of satisfaction, especially in usability, design clarity, and successful operation of core features like renting, payments, and profile management. Overall, the testing confirmed the system's readiness from both technical and user perspectives.

5.2.1 Functional Testing

Functional testing is carried out by executing targeted test cases for each system module to verify that it performs as intended. This process includes outlining test scenarios, developing corresponding test cases, and comparing the actual results with the expected ones to identify and address any errors or inconsistencies.

5.2.2 Test Plan

A test plan is a comprehensive document that defines the testing objectives, strategies, timeline, required resources, deliverables, and estimated efforts. Its main purpose is to verify that the developed system meets all functional requirements. Each module is tested using predefined test cases to ensure the actual results align with the expected outcomes. While the full test plan includes six tables covering different modules, only the test plan for the rental module is shown below due to page limitations. There were no failures during the testing process; all test cases passed successfully.

Table 5 *Application Testing for Rental Module*

No	Function (User Role)	Test Case	Expected Result	Actual Result
1	Book Item for Rent (User)	Choose item, select duration, and confirm rental	Rental request is submitted and confirmation is shown	Pass
2	View Rental History (User)	Navigate to rental history page	A list of past and current rentals with dates and item names is displayed	Pass
3	Add to Wishlist (User)	Tap heart icon on item	Item is added to user's personal Wishlist section	Pass
4	Add Item to Cart (User)	Press "Add to Cart" on listing	Item appears under user's cart and is stored for checkout	Pass
5	Item Listing (Owner)	Upload item name, description, image, price, deal method and choose category and location.	Item is posted live in the marketplace and visible to others	Pass
		Edit listing by update details and click save changes to update the listing	Changes are saved and updated listing is shown in marketplace	Pass
		Remove listing by clicking the remove button	Listing is removed from the marketplace and no longer visible to users	Pass

5.3 User Acceptance Testing

A test plan outlines the strategy, resources, and timeline for testing each module to ensure the system functions correctly, with results compared to expected outcomes.

5.3.1 User UAT

This section discusses users' overall experience and feedback on the functionality and usability of the developed system.

Table 6 *Result of overalled system developed from user*

No.	Features	Ranking					Total
		1	2	3	4	5	
1	Are you happy with how the app works overall?	-	-	-	4	11	15
2	Do you think the app is useful for renting and managing items?	-	-	-	2	13	15
3	Would you recommend this app to your friends or family?	-	-	-	2	13	15

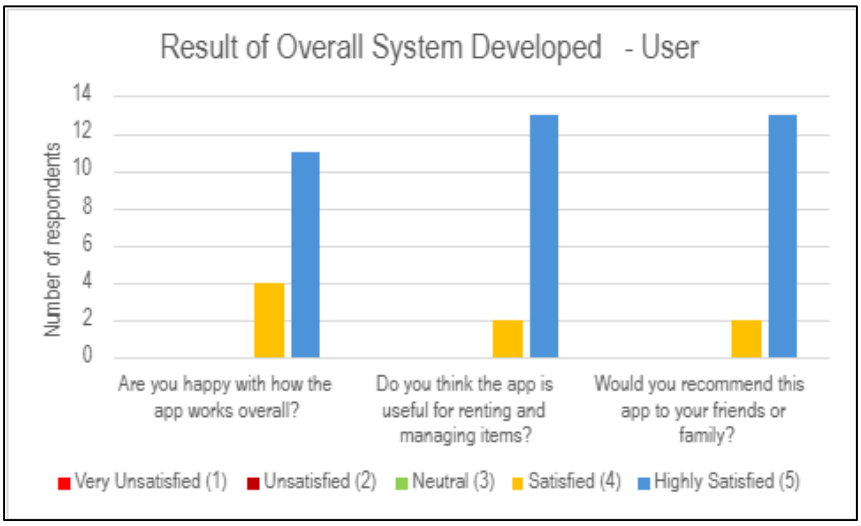


Fig. 16 *Bar chart diagram created for Overall System Developed user section*

5.3.2 Administrator UAT

This section presents the administrator's overall experience and feedback regarding the functionality and usability of the developed system.

Table 7 *Result of overalled system developed from administrator*

No.	**Features**	**Ranking**					**Total**
		1	2	3	4	5	
1	The system effectively meets administrative needs.	-	-	-	-	1	1
2	The admin panel provides sufficient control and access.	-	-	-	-	1	1

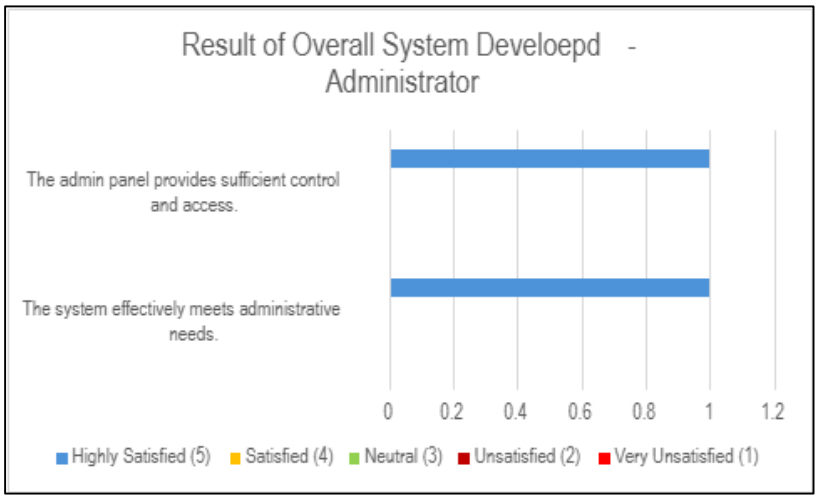


Fig. 17 *Bar chart diagram created for Overall System Developed administrator section*

6. Conclusion

The Rent Anything for UTHM Community system was successfully developed to meet its primary objectives, offering an easy-to-use and fully functional web platform that facilitates rental transactions among UTHM students and staff. It features essential components such as secure user management, advanced search and filtering options, real-time payment processing, in-app messaging, and user reviews, which together ensure a smooth rental process. User feedback during system testing indicated a high level of satisfaction with its core features and usability, affirming its effectiveness in streamlining rental activities within the campus community. Results from the User Acceptance Testing (UAT) showed that all 15 participants rated the system positively, with most awarding the highest score for overall satisfaction, usefulness, and likelihood of recommending it to others. In addition, the administrator's UAT response indicated complete approval, emphasizing that the system adequately fulfills administrative requirements and offers appropriate control through the admin interface. These outcomes collectively validate the system's performance and usability from both end-user and administrator perspectives. Despite its strengths, the system has some limitations, including a basic messaging feature lacking support for media, no automated email reminders for item return deadlines, and the absence of automatic account deactivation for users who have left the university. Future improvements are suggested, such as enhanced messaging capabilities, improved mobile compatibility, email notifications, broader user testing, advanced administrative tools, and increased security measures. In summary, the system provides a solid foundation for a convenient and sustainable rental service within the UTHM community, with clear opportunities for further enhancement.

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

This journal 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:*** *Aina Saffiya Ahmad Shukry Norfaradilla Wahid;* ***data collection:*** *Aina Saffiya Ahmad Shukry;* ***analysis and interpretation of results:*** *Aina Saffiya Ahmad Shukry, Norfaradilla Wahid;* ***draft manuscript preparation****: Aina Saffiya Ahmad Shukry, Norfaradilla Wahid. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] Cross, M., & Reitz, C. (2023, September 26). When Renting Is Smarter Than Buying. Kiplinger.com; Kiplinger. https://www.kiplinger.com/article/real-estate/t010-c047-s002-when-renting-is-better-than-buying.html.

[2] Pichierri, M., Scarpi, D., & Pizzi, G. (2018). To buy or to rent? An experimental study on the antecedents of consumers' acquisition-mode decisions. *MERCATI & COMPETITIVITÀ*, *1*, 63–92. https://doi.org/10.3280/mc2018-001005

[3] Sustainability Performance of Rental and Sharing Models: a Life Cycle Assessment. (2024). *Mondragon.edu*. https://hdl.handle.net/20.500.11984/6421

[4] Agnieszka Napiórkowska-Baryła, Świdyńska, N., & Mirosława Witkowska-Dąbrowska. (2024). Owning versus Renting a Home—Prospects for Generation Z. *Sustainability*, *16*(11), 4715–4715. https://doi.org/10.3390/su16114715

[5] KHOO, D. (2024, September 4). Renting expected to gain popularity. The Star. https://www.thestar.com.my/business/business-news/2024/09/05/renting-expected-to-gain-popularity.

[6] *Smart Rental - All Products*. (2024). Smartrental.asia. https://www.smartrental.asia/en/products?category=Laptop

[7] Just Rent It! Malaysia – Your event solution provider. (n.d.). https://www.justrentitmalaysia.com/

[8] Carousell. (n.d.). *Carousell - Snap to List, Chat to Buy*. Www.carousell.com.my. https://www.carousell.com.my/

[9] "toyyibPay - Quick & easiest online payment solution." https://toyyibpay.com/main/

[10] Diefenbach, S., Christoforakos, L., Maisch, B., & Kohler, K. (2019). The State of Prototyping Practice in the Industrial Setting: Potential, Challenges and Implications. *Proceedings of the Design Society: International Conference on Engineering Design*, *1*(1), 1703–1712. https://doi.org/10.1017/dsi.2019.176

[11] Kendall, K. E., & Kendall, J. E. (2014). Systems analysis and design. In *thuvienso.hoasen.edu.vn*. Pearson. https://thuvienso.hoasen.edu.vn/handle/123456789/11193

Appendix A: Gantt Chat

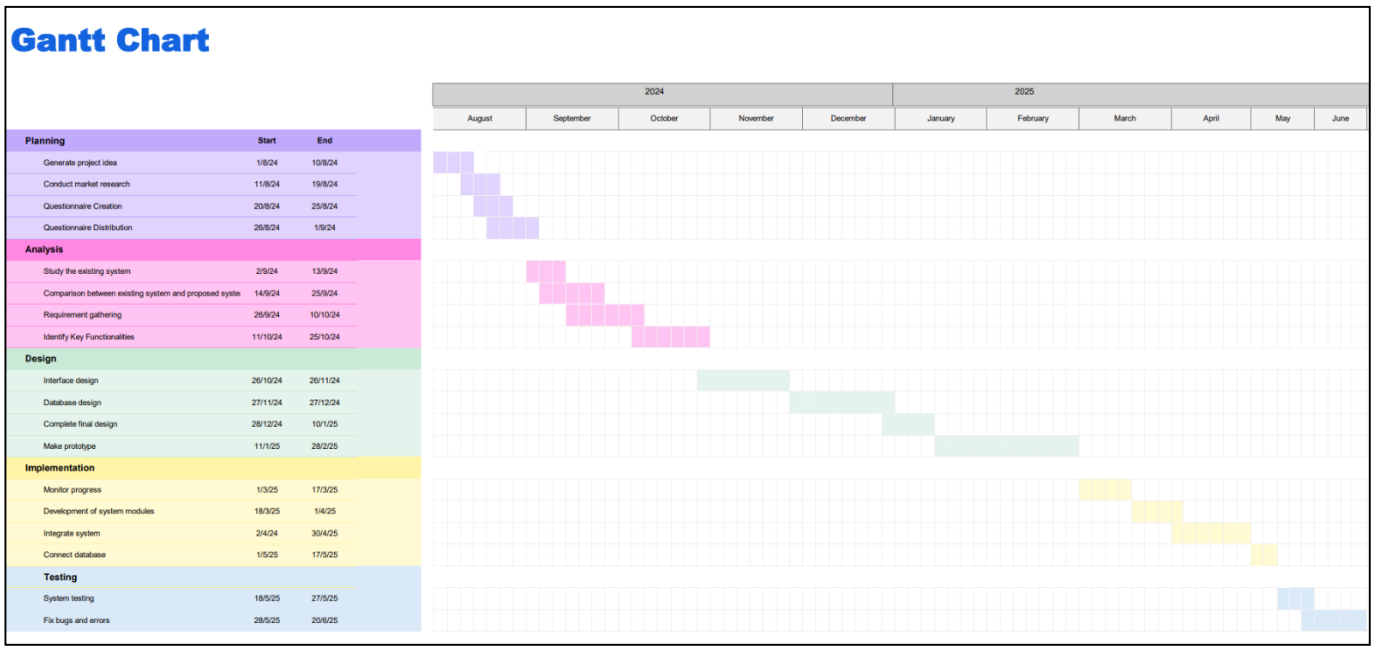


Fig.18 *Gantt Chart of the Rent Anything*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

Appendix B: Data Flow Diagram (DFD) Level 0

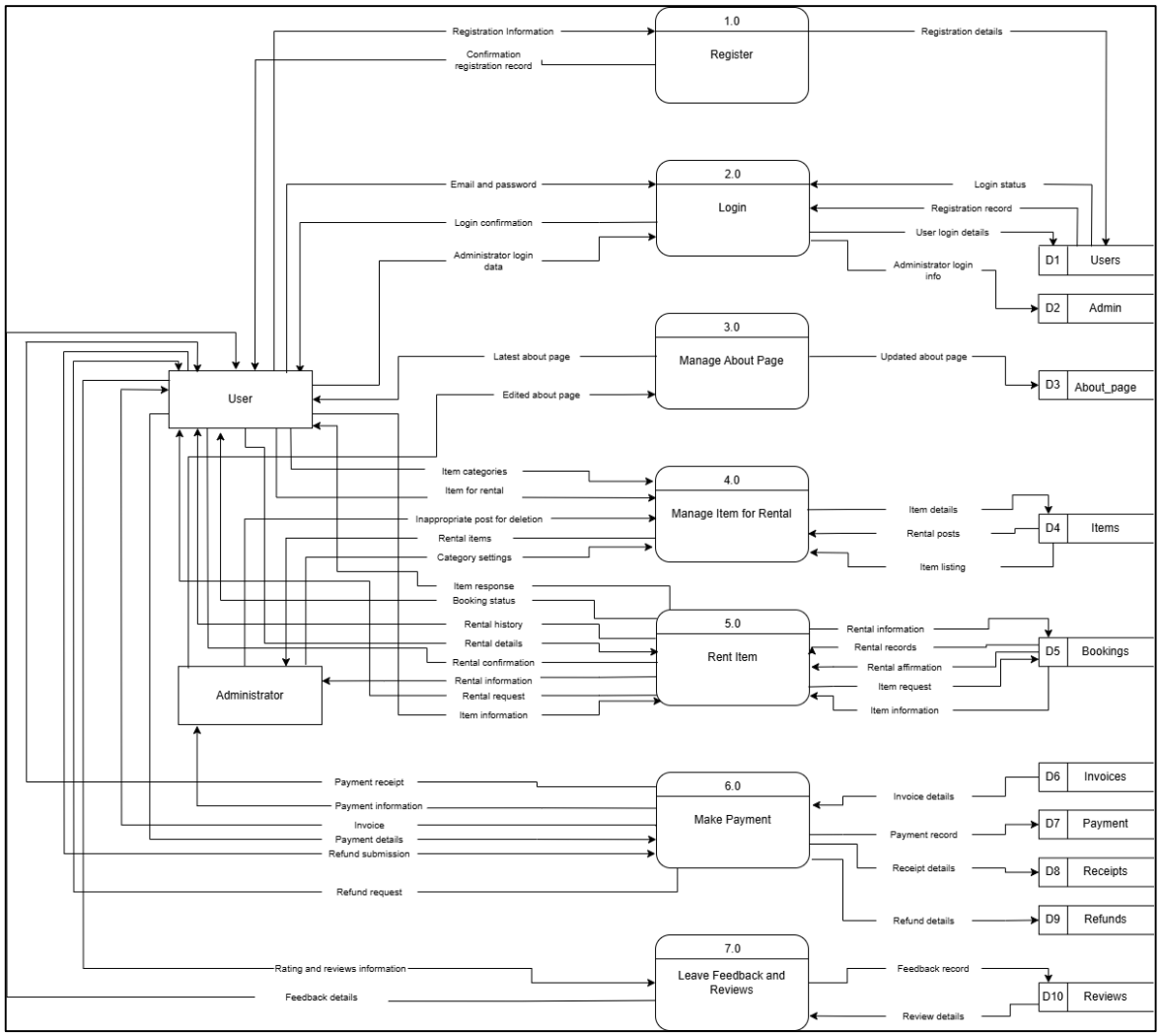


Fig.19 *Data Flow Diagram (DFD) Level 0 of the Rent Anything*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

Appendix C: Data Flow Diagram (DFD) Level 1

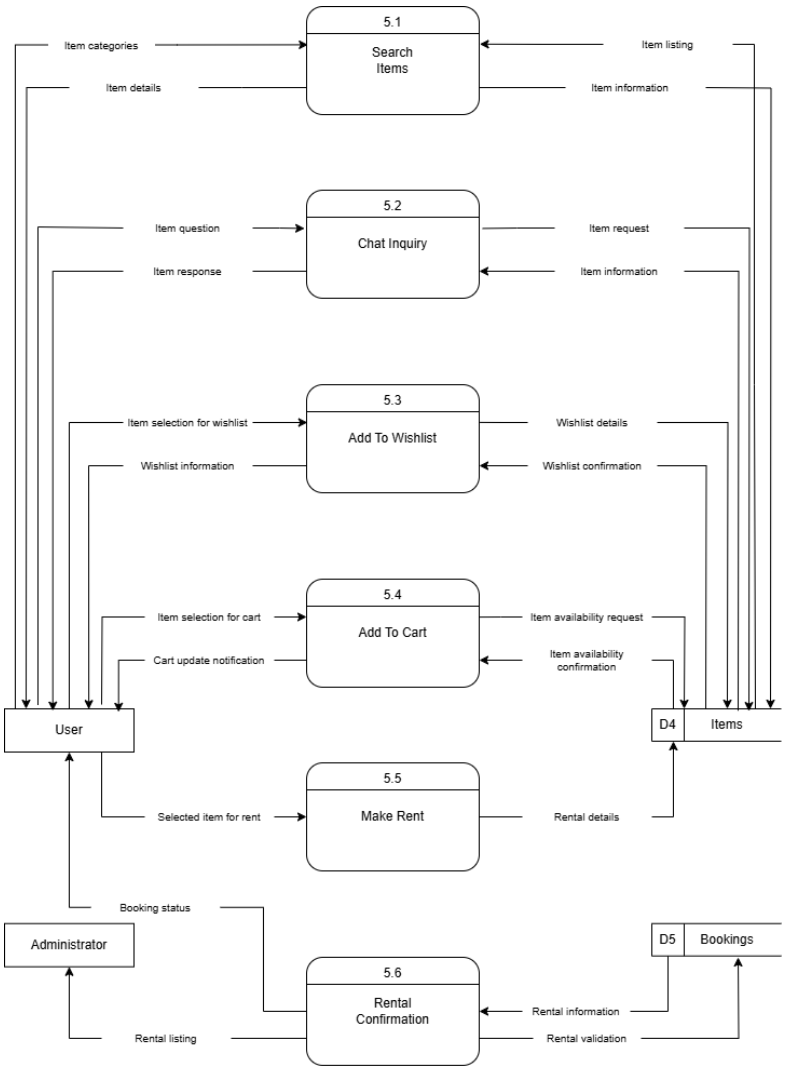


Fig.20 *Data Flow Diagram (DFD) Level 1 of Rent Item*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 36-56

Appendix D: Entity Relationship Diagram (ERD)

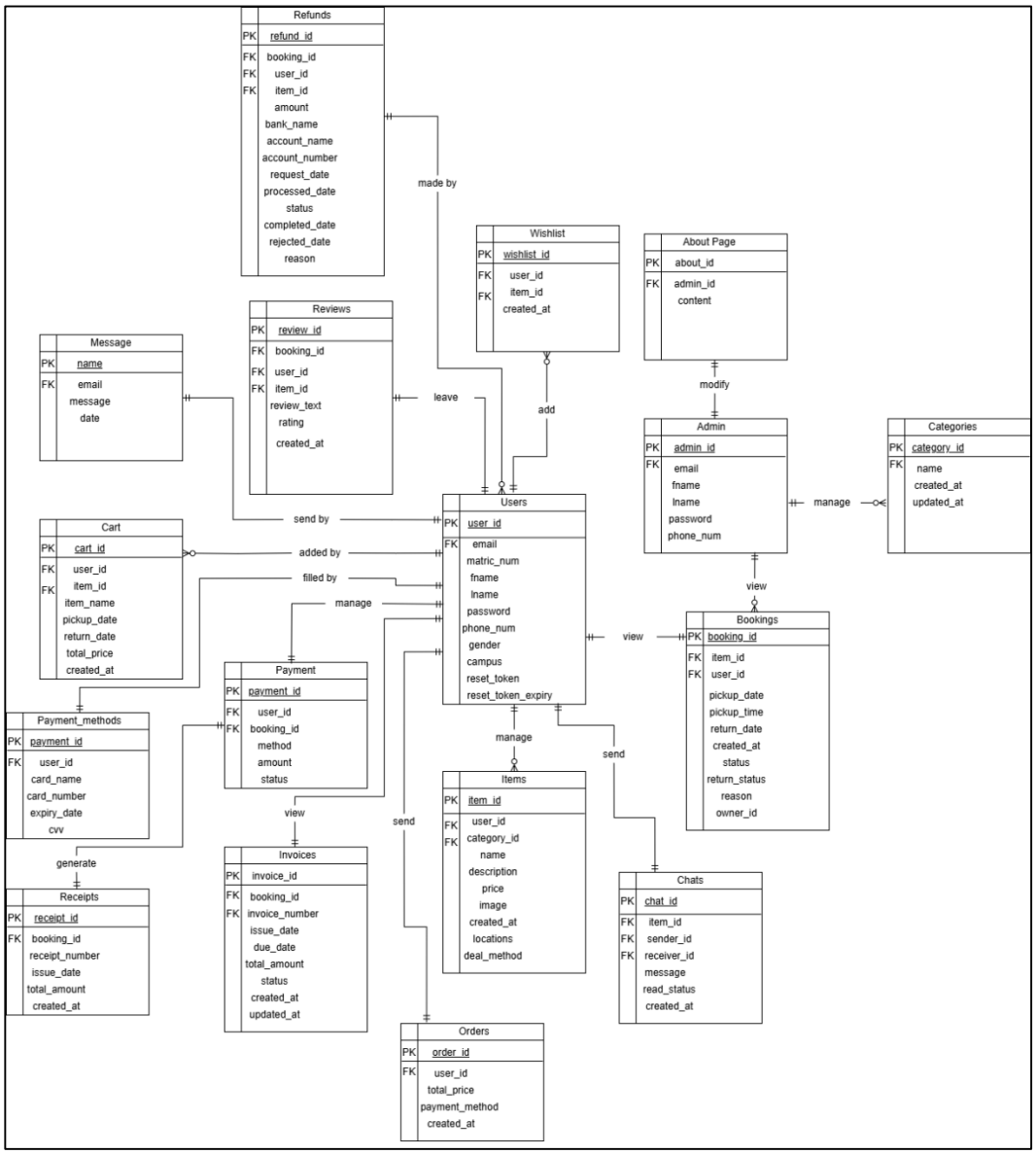


Fig.21 *Entity Relationship Diagram (ERD) for Rent Anything*

Appendix E: Database Design

Table 8 *Database Dictionary for Orders*

Attributes	Data Type	Key	Description
order_id	int(11)	PK	Unique ID of the order
user_id	int(11)	FK	Unique ID of the user
total_price	decimal(10,2)		Total price of the order
payment_method	varchar(50)		The payment method chosen
created_at	datetime		The date for order created