Applied Information Technology And Computer Science Vol. 4 No. 2 (2023) 1431-1450

AITCS

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

The Development of a Mobile Application for a Reward-Based Student-Lecturer Appointment Using Ethereum Blockchain and Smart Contracts

Muhamad Syamim Irfan Ahmad Shokkri[1], Nur Ariffin Mohd Zin[1]

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

DOI: https://doi.org/10.30880/aitcs.2023.04.02.081
Received 23 June 2023; Accepted 09 November 2023; Available online 30 November 2023

Abstract: WeMeet DApps is an application that utilizes Blockchain technology and Smart Contracts to facilitate student-lecturer appointments and provide rewards. The primary objective of this application is to foster connections between UTHM students and their respective teachers. Furthermore, it offers the possibility of granting prizes to students who demonstrate a strong commitment to their academic pursuits. The UTHM Token, valued at 1 Ringgit Malaysia (RM) per unit, is awarded as a reward. By utilizing the smart contract address, all transactions involving this token can be effectively monitored and tracked. The application is developed using an object-oriented approach, employing the Prototype model to ensure compatibility with the system development process. The software used for development is Visual Studio Code, a code editor, which is based on Android Technology. Additionally, a MySQL server is employed to store data in a database. Ultimately, the anticipated outcome of WeMeet DApps is to provide optimal utilization and benefits to UTHM students and lecturers alike.

Keywords: WeMeet Dapps, UTHM Token, SepoliaETH, Ethereum, Ringgit Malaysia, Android Technology, Web-Based System, MetaMask, Prototype Model, Object-Oriented Approach, Multitier Architecture, encryption, bcrypt, Email OTP, QR Code, Flutter, Bootstrap, Node.js, MySQL, Hardhat, iOS Technology

1. Introduction

Universiti Tun Hussein Onn Malaysia (UTHM) is a Malaysian public university established in 1993, specializing in engineering and technology. Throughout the academic semester, students need to interact with their professors regarding assignments, projects, and lessons. However, a significant number of UTHM students encounter challenges in managing their time effectively and seldom engage with their lecturers. Previously, students had to manually arrange meetings by contacting their lecturers through phone calls or messaging applications like WhatsApp. These meetings typically took place at

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

the faculty office, with some lecturers relying on students approaching their desks randomly for attendance purposes. This endeavor aims to identify dedicated students, reward their commitment, and provide them with additional credit for their efforts. Unfortunately, the existing system lacks a means of storing documents or data. In line with this, students will receive a reward for attending appointments, which will be based on a cryptocurrency known as the "UTHM" token. The total supply of this token will be backed by the resources available in Pusat Waqf and Endowment. Lecturers will be responsible for coordinating the distribution of rewards to the students. The token is built on the Ethereum network, enabling the coordinator to track its flow.

The primary objective of this project is to develop WeMeet Dapps, an application based on an object-oriented approach, utilizing blockchain and smart contracts, for rewarding student-lecturer appointments. The application is designed using Android Technology to assess the functionality of WeMeet Dapps. The scope of this project is focused on UTHM students, lecturers, and coordinators. Several modules are incorporated within the application, including the login module, view profile module, manage slot module, booking appointment module, manage booking appointment module, manage live chat module, manage attendance module, manage appointment history module, manage user module, and manage reward module.

2. Literature Review

2.1 Introduction to Decentralized Application, Blockchain, and Smart Contracts.

According to [1], decentralized applications (dApps) are a specific type of open-source software that operates on a peer-to-peer (P2P) network of computers rather than a single computer. The key distinction between dApps and other applications accessible through websites or mobile devices is their ability to leverage P2P technology. Decentralized applications function on blockchain networks, as opposed to centralized applications which are owned and operated by a single entity. This fundamental contrast between decentralized centralized applications grants decentralized applications the advantage of not relying on centralized storage or a single server, thereby alleviating concerns related to compromised data or server downtime [2].

Blockchain, the underlying technology supporting decentralized applications, is constructed on a peer-to-peer (P2P) network and comprises block hashes, which are 256-bit hash values of the preceding blocks [3]. Consequently, each user within the blockchain network functions as a node and assumes the responsibility of verifying the authenticity of newly added blocks. The use of blockchain technology within a P2P network effectively safeguards databases against fraudulent and inaccurate transactional data. Moreover, the immutability of blockchain ensures data integrity and prevents tampering, establishing a system where asset provenance can be tracked to determine their location and history within the blockchain framework [4].

As explained by [5], a smart contract is a software component that executes on the Ethereum blockchain. It encompasses both data and computer programs stored at a specific address within the Ethereum blockchain. The data represents the state of the smart contract, which falls under the category of Ethereum accounts. In the case of UTHM Token, which is built on top of the immutable Ethereum network and incorporates Smart Contract addresses, it becomes feasible to trace the flow of transactions initiated by students.

2.2 Android Technology

The Android operating system, developed by Google, is specifically designed for touchscreen devices such as smartphones, tablets, and other mobile devices. These devices offer an intuitive user experience through finger movements that mimic common actions like pinching, swiping, and tapping. Google utilizes Android software to provide a distinct user experience across various devices, including watches, cars, and televisions [6]. Today, the Android operating system is globally recognized, with a staggering user base of nearly 3 billion, accounting for 39% of the world's population, and over 2.5 billion active devices worldwide.

In the realm of Android, there exists a vast ecosystem of applications, with a minimum of 3 million available for users. While Google Play features 11 commonly encountered applications, these apps are also accessible online. Android applications are specifically developed to function on Android hardware or emulators, and they are packaged in the form of an APK (Android Package). An APK contains all the necessary elements, including metadata, resources, and application code, bundled together in a zip format. Android applications can be written in Java, C++, or Kotlin and executed on virtual machines (VMs) [7]. Notably, mobile users spend only 12% of their online time on mobile websites, unequivocally indicating a preference for applications over web pages. This preference stems from the numerous advantages offered by applications, including faster access and convenience. Mobile users can access applications with greater speed and ease compared to websites, and the ubiquity of mobile devices like Android enables users to access applications anytime and from anywhere [8].

2.3 Comparison of the existing system

The examination of the existing system is conducted by analyzing previously developed systems. Within this project, three specific systems have been chosen as points of reference for the development of the current system. These systems serve as valuable sources of guidance to enhance the quality of the present project. The selected systems include the Patient Appointment Application for Mawar Medical Centre Unit Dialysis, the Hotel Booking System of Ridel Hotel Kota Bharu, and the PKU System: Development of Online UTHM Health Center Appointment System. A comparative analysis between these three related systems and the system proposed in this project has been summarized in Table 1, considering various features.

Table 1: Comparison of Existing System and Proposed System

Feature/Sistem	Patient Appointment Application for Mawar Medical Centre Unit Dialysis	Hotel Booking System for Ridel Hotel Kota Bharu	PKU System: Development of Online UTHM Health Center Appointment System	WeMeet Dapps: A Reward-Based Student-Lecturer Appointment using Blockchain and Smart Contracts
Platform	Mobile application	Web-based system	Mobile Application	Mobile Application
Technology/Algorithm	Android	Web-based architecture	Android	Android, Blockchain, Ethereum, and Smart Contracts
Login	Yes	Yes	Yes	Yes
Profile	Yes	No	Yes	Yes
Appointment/Booking	Yes	Yes	Yes	Yes
Manage Appointment/ Booking	Yes	No	No	Yes
Live Chat	Yes	No	No	Yes
Manage Appointment/	Yes	Yes	Yes	Yes

Feature/Sistem	Patient Appointment Application for Mawar Medical Centre Unit Dialysis	Hotel Booking System for Ridel Hotel Kota Bharu	PKU System: Development of Online UTHM Health Center Appointment System	WeMeet Dapps: A Reward-Based Student-Lecturer Appointment using Blockchain and Smart Contracts
Booking History				
Manage Reward	No	No	No	Yes
Manage User	Yes	Yes	Yes	Yes
Users	Patient	Customer	Students, Doctors, and Admin	Students, Lecturers, and Admin
Design Interface	Dynamic	Dynamic	Dynamic	Dynamic

3. Methodology

The study conducted by [9] suggests the utilization of the Prototype model for developing the proposed system. The Prototype model is chosen due to its ability to address user requirements effectively. The development process will follow a sequential and simultaneous approach, encompassing the analysis phase, design phase, and implementation and testing phase. This ensures that the system meets the desired objectives and adequately fulfills user needs.

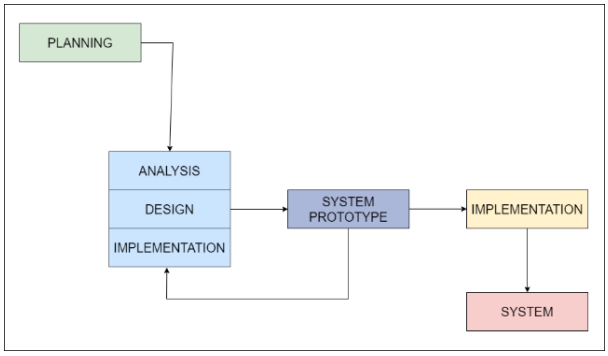


Figure 1: System Prototyping Model

3.1 Project Planning

Table 2 shows the list of tasks performed at each phase in this project.

Table 2: List of tasks in the System Prototype Model

Phase	Task
Planning	• Proposed the project. • Determine the project background, problem statement, objectives, scope, expected result, and project significance. • Interview the stakeholders.

Phase	Task
	• Develop a project proposal. • Prepare a schedule of timeline, tasks, and output.
Analysis	• Analyze system requirements. • Investigate the existing system. • Analyze hardware and software requirements. • Determine the programming language, database, framework, and library that will use. • Choose the best methodology. • Develop the use case diagram, activity diagram, sequence diagram, and class diagram.
Design	• Design the interface design. • Design the database.
Prototype Implementation	• Develop prototypes with Figma. • Collect feedback from stakeholders for the prototype. • Refined the prototype. • Develop the second prototype. • Collect feedback from stakeholders for the second prototype. • Finalize the design and prototype.
Final Implementation	• Develop the program code for the system interface. • Develop the program code for blockchain and smart contracts. • Develop the program code API to connect the backend and database. • Refactor the code and debug the code. • System Testing (User Acceptance Testing). • Roll out.

4. System Analysis and Design

This section presents an analysis and design overview of the system, encompassing functional requirements, non-functional requirements, and user requirements for the business process. The system design adopts an object-oriented approach, incorporating the design of use case diagrams, class diagrams, and a multitier architecture for the application.

4.1 Requirement Analysis

The system analysis phase plays a crucial role in comprehending and identifying the requirements of the system. To present a comprehensive overview, Table 3 and Table 4 outline the functional and non-functional requirements of the system, respectively. Additionally, the user requirements outline the specific demands and expectations of the system's users, which are depicted in Table 5. These tables provide a clear understanding of the system's requirements from different perspectives.

Table 3: Functional Requirements

Requirements	Description
Login Module	• The student and lecturer shall login and logout of their account by email and password. • The coordinator shall login and logout by password only. • The student and lecturer shall reset the password.
Manage Profile	• The student shall view their profile and lecturer profile. • The lecturer shall update the information for their profile. • The lecturer shall view the updated information for their profile.
Manage Slot	• The lecturer shall add and update the slot. • The lecturer shall view the slot. • The coordinator shall delete all the slots.
Book Appointment Module	• The student shall search or filter the faculty to choose a lecturer. • The student shall create the appointments by choosing the lecturer, date, and time as they wanted. • The lecturer shall accept or reject the requested appointments from the students.
Manage Appointment	• The student shall update the created appointments. • The student shall update the number of students with the same date and time for the created appointments. • Students are not allowed to update the lecturer. • The lecturer can cancel the accepted appointments if they want to cancel the appointments.
Manage Live Chat	• The student and lecturer shall send and receive the message with each other. • The student and lecturer shall copy the message. • The student and lecturer shall delete the message. • The message will receive in real-time in the chat section.
Manage Attendance	• The lecturer shall determine whether students attend or are absent from the appointments.
Manage Appointment History	• The student shall view the status of attendance and other information in their appointment. • The student shall meet again with the same lecturer.
Manage User	• The coordinator shall register the student and lecturer accounts. • The coordinator shall delete the student and lecturer accounts.

Requirements	Description
Manage Reward	• The coordinator shall give 1 UTHM token for the attended appointment. • The coordinator shall track the transaction that has been made. • The student shall scan the QR code with the seller or anyone's account. • The student shall transfer the UTHM token to their account.
Performance	• It is reasonable to anticipate that the operating system will operate and respond quickly. • The response time shall be less than 10s
Operational	• The application should work on the Android platform and any web browser. • The application shall be able to use anytime with the presence of an internet connection.
Security	• Students and lecturers shall log in to the application by email and password based on their roles. • Coordinator shall log in to the system with a password. • The password must be at least 8 characters in the application. • The OTP will be sent in the email to reset the password. • The session in coordinator pages will reset every 7 days and requires login to continue. • The password needs to be encrypted in the database.
Usability	• The application shall give a good user-friendly and easy to understand for users.
Integrity	• The system will appropriately manage and protect the application's database from corruption and unreadable data.
Availability	• The application is guaranteed to be simple to use and accessible at all times.
Maintainability	• All the implementation such as source code will be kept on the Git repository. So, whenever the programmer wants to make some modifications, it will not affect the existing system that is running.

Table 5 shows the user requirements that define the system user demands and expectations for the system functions.

Table 5: User Requirements

No	User Requirements
1	Students and lecturers should be able to input the email and password for login to the application.
2	Students should be able to sign up and log in to the MetaMask application.
3	Students will be able to read the user manual to import UTHM tokens in the MetaMask application.
4	The coordinator should be able to input the password for login to the system.
5	The coordinator should be able to register the student and lecturer accounts.
6	The coordinator should be able to delete the student and lecture accounts.
7	Students and lecturers should be able to reset their password in the application.
8	Students should be able to search and filter for the lecturer's name on the booking page.
9	Students should be able to create new booking appointments in the application.
10	Lecturers should be able to accept or reject the requested appointment from the students.
11	Students should be able to update the booking appointment except for the chosen lecturer.
12	Lecturers should be able to delete the booking appointment to cancel the meeting.
13	Lecturers should be able to determine whether students attend or are absent from the meeting.
14	Students should be able to search for the lecturer's name on the chat page.
15	Lecturers should be able to search for the student's name on the chat page.
16	Students and lecturers should be able to send and receive messages on the chat page.
17	Students should be able to display the previous appointments.
18	Students should be able to book again with the same lecturer on the history page.
19	The coordinator should be able to create the total of tokens in the system.
20	The coordinator should be able to update the total of tokens in the system.
21	The coordinator should be able to set the total of tokens for each appointment in the system.
22	Students should be able to claim the token after successfully attending the appointment.
23	Students should be able to scan the QR Code to use the token.

4.2 System Analysis

During the system analysis and design phase, the structure and behavior of the system are thoroughly examined and represented using UML diagrams. One such diagram is the UML Use Case diagram, which illustrates the behaviors of the system and the interactions between actors and these behaviors. By analyzing this diagram, stakeholders can gain a comprehensive understanding of the system's functionalities and how users engage with them. Additionally, other UML diagrams such as class

diagrams and system architecture diagrams contribute to the system's design by depicting its underlying structure and multi-tier architecture. These UML diagrams serve as valuable tools for capturing, documenting, and communicating the system's key components, relationships, and interactions, facilitating a robust system analysis and design process.

The class diagram provides a static representation of the system's structure and the interactions between its components. It serves as a valuable tool for visualizing, describing, and documenting the various components of the system, as well as for generating executable code for software applications. Figure 3 in Appendix A presents the class diagram for the WeMeet DApps, which is a reward-based student-lecturer appointment system utilizing blockchain and smart contracts. The diagram illustrates the relationships and attributes of the system's classes, aiding in the understanding and implementation of the system's functionalities.

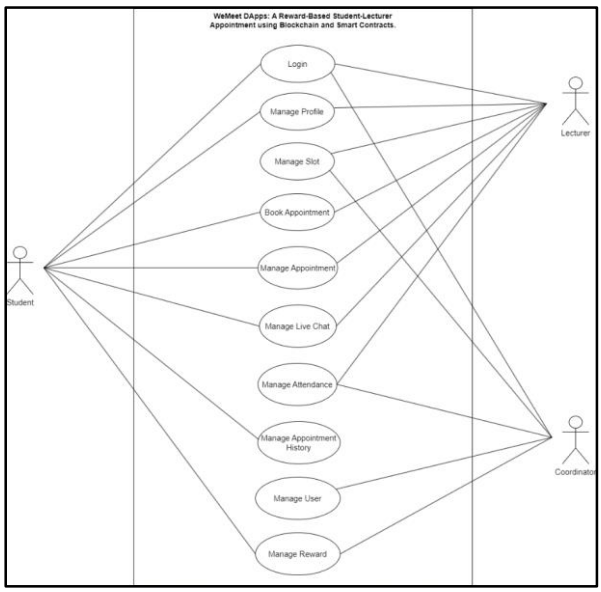


Figure 2: Use Case Diagram

Figure 2 provides a visual representation of the use case diagram, which outlines the actions performed by actors when interacting with the system. The diagram includes actors such as students, lecturers, and coordinators. The login module facilitates user authentication and authorization, enabling coordinators to register student and lecturer accounts for application access, as well as delete accounts for individuals who are no longer affiliated with UTHM.

The booking appointment, manage booking appointment, and manage appointment history modules allow students and lecturers to schedule and manage their appointments. These modules incorporate elements for creating, reading, updating, and deleting appointment records.

Furthermore, the manage reward module enables coordinators to establish the initial token value within the application. Administrators are responsible for controlling the rewards allocated for each successfully attended appointment by students, while also monitoring all token transactions between students and the application.

4.3 System Design

The system analysis and design phase involve the development of a comprehensive system architecture, which provides a conceptual diagram illustrating the organization and functioning of various components and subsystems within the system. This includes hardware, software, network devices, and other equipment, all described using the Architecture Description Language (ADL). Figure 3 visually presents the system architecture, offering a clear representation of its structural framework and relationships between different elements.

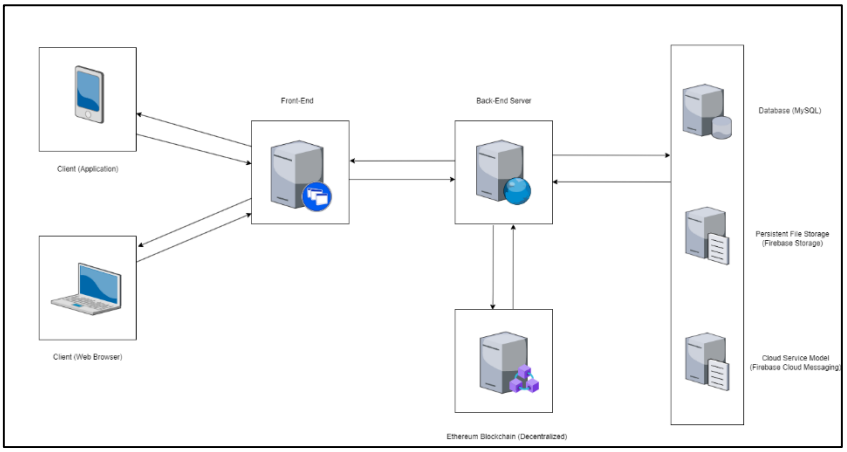


Figure 3: System architecture (Multitier)

The database is designed to receive and store data for the application. In this section, the database schema will describe the entities that hold data for the application. The database schema for the application is shown:

i. **Student** (matricNo, icNumber, tokenAddress, studName, studTelephoneNo, studEmail, studPassword, studImage,studImageFirebase, faculty, program,createdDate, status)
ii. **Lecturer** (staffNo, lecturerName, icNumber, lecturerTelephoneNo, lecturerEmail, lecturerPassword, lecturerImage, lecturerImageFirebase, faculty, department createdDate, status)
iii. **Lecturer Information** (staffNo, floorLvl, roomNo academicQualification1, academicQualification2, academicQualification3, academicQualification4)
iv. **Slot** (slotId, staffNo, day, slot1, slot2, slot3, slot4, slot5)
v. **Booking** (bookingId, matricNo, staffNo, lecturerName, numberOfStudents, date, time, statusBooking)
vi. **Chat** (matricNo, staffNo, messageText, sendTextTime, statusMessage)
vii. **Attendance** (attendanceId, staffNo, status, lecturerName, numberOfStudents, date, time)
viii. **Reward** (rewardId, matricNo, UTHMToken, studentTokenAddress)
ix. **Hash** (hashId, matricNo, trackingHash)
x. **Transaction** (keyTransaction, amountToken, sellerTokenAddress, studentTokenAddress)

5. Implementation

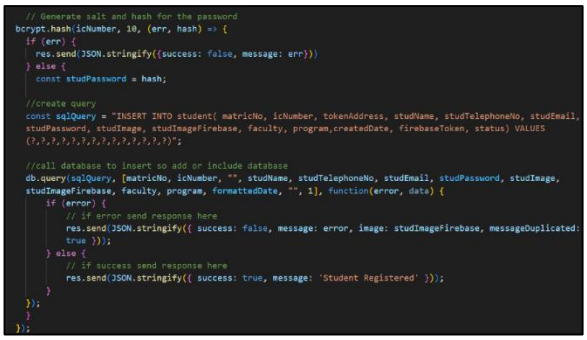
This section discusses the technology used to complete the project, the implementation of the security properties, the implementation of book appointments, and the implementation of the transfer of the UTHM token.

5.1 Technology

The application's interface design is implemented using the Flutter framework and the Dart language. Similarly, the system's interface design utilizes the Bootstrap framework along with HTML, CSS, and JavaScript languages. For the backend development, JavaScript serves as the backend language, supported by the Node.js framework. The smart contract for the Ethereum Blockchain network is coded using Solidity and JavaScript programming languages, with the assistance of the Hardhat framework. This allows for the deployment of the smart contract address for the UTHM token and the integration of its ABI code into the backend of the proposed system.

5.2 Implementation of security properties

Figure 4 and Figure 5 show the encryption of the password by using hash and salt. The password should contain at least 8 characters and the password need to salt into 10-bit of salt rounds in the bcrypt package inside of Node.js.



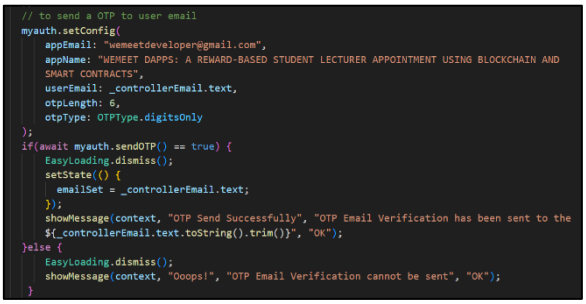
```
// Generate salt and hash for the password
bcrypt.hash(icNumber, 10, (err, hash) => {
  if (err) {
    res.send(JSON.stringify({success: false, message: err}))
  } else {
    const studPassword = hash;

  //create query
  const sqlQuery = "INSERT INTO student( matricNo, icNumber, tokenAddress, studName, studTelephoneNo, studEmail,
  studPassword, studImage, studImageFirebase, faculty, program,createdDate, firebaseToken, status) VALUES
  (?,?,?,?,?,?,?,?,?,?,?,?,?,?);";

  //call database to insert so add or include database
  db.query(sqlQuery, [matricNo, icNumber, "", studName, studTelephoneNo, studEmail, studPassword, studImage,
  studImageFirebase, faculty, program, formattedDate, "", 1], function(error, data) {
      if (error) {
          // if error send response here
          res.send(JSON.stringify({ success: false, message: error, image: studImageFirebase, messageDuplicated:
          true }));
      } else {
          // if success send response here
          res.send(JSON.stringify({ success: true, message: 'Student Registered' }));
      }
  });
  }
});
```

Figure 4: JavaScript source code to encrypt the password

studPassword
$2b$10$0U1sln2iROAj3p2o8cxCvuVY00z5CeEHg3do9yixVzNDFyZZOzlL6

Figure 5: Password encrypted from the database

Figure 6 shows the code in the front end from Dart language to send email OTP to the user. Figure 7 shows the OTP has been sent to the email.



```
// to send a OTP to user email
myauth.setConfig(
    appEmail: "wemeetdeveloper@gmail.com",
    appName: "WEMEET DAPPS: A REWARD-BASED STUDENT LECTURER APPOINTMENT USING BLOCKCHAIN AND
    SMART CONTRACTS",
    userEmail: _controllerEmail.text,
    otpLength: 6,
    otpType: OTPType.digitsOnly
);
if(await myauth.sendOTP() == true) {
    EasyLoading.dismiss();
    setState(() {
      emailSet = _controllerEmail.text;
    });
    showMessage(context, "OTP Send Successfully", "OTP Email Verification has been sent to the
    ${_controllerEmail.text.toString().trim()}", "OK");
}else {
    EasyLoading.dismiss();
    showMessage(context, "Ooops!", "OTP Email Verification cannot be sent", "OK");
}
```

Figure 6: Dart source code to send OTP in users email

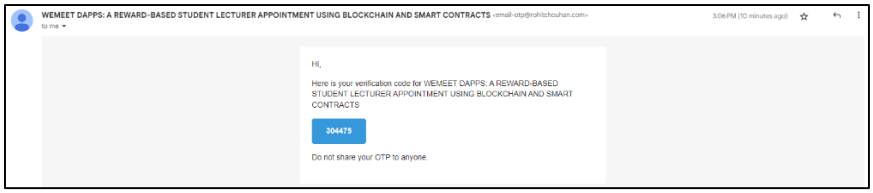


Figure 7: OTP sent to the email

Figure 8 and Figure 9 show the code to end the session when the coordinator does not logout from the system in 7 days.

```
var expirationTimestamp = currentTimestamp + (7 * 24 * 60 * 60 * 1000); // 7 days

sessionStorage.setItem("loginTimestamp", expirationTimestamp);
```

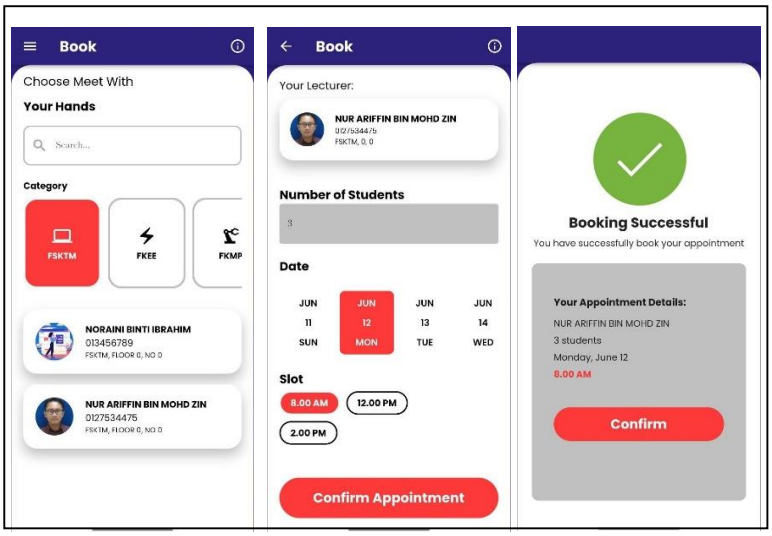
Figure 8: Set the expiration date in 7 days

```
function checkSessionExpiry() {
var loginTimestamp = sessionStorage.getItem("loginTimestamp");

if (loginTimestamp) {
    var currentTimestamp = new Date().getTime();
    var storedTimestamp = parseInt(loginTimestamp);

    if (currentTimestamp > storedTimestamp) {
        sessionStorage.removeItem("password");
        sessionStorage.setItem("logoutFlag", 'true');
        sessionStorage.removeItem("loginTimestamp");

    session_popup.style.display = "block";
        r_close.onclick = function() {
            session_popup.style.display = "none";
            location.href = "../index.html";
        }
    }
} else {
    session_popup.style.display = "block";
        r_close.onclick = function() {
            session_popup.style.display = "none";
            location.href = "../index.html";
        }
}
}

// Check the session expiry periodically
setInterval(checkSessionExpiry, 1000);
```

Figure 9: Counting every 1 second to end the session

5.3 Implementation of book appointments

This section discusses the implementation of the book appointments. These include creating the appointments, accepting, or rejecting the appointment, receiving the appointment, updating the appointment, and deleting the appointment. Figure 10 shows the flow of creating the booking appointment interface on the student's side.



(a) **(b)** **(c)**

Figure 10: The (a) first book page, (b) second book page, and (c) booking successful page

5.4 Implementation of transfer UTHM token

This section focuses on the implementation of UTHM token transfers within the system. The coordinator is responsible for sending UTHM tokens to students who have successfully booked and attended appointments with lecturers. Upon successful token transfers, the system records the transaction hash, which can be accessed on the track token page. By clicking on the etherscan.io link within the Sepolia network, the coordinator can track the token's progress.

Additionally, the coordinator can monitor student transactions by utilizing their smart contract or token addresses on the specified website. By copying and pasting the UTHM token address (e.g., 0xac60Ae1E7BeE92cf1b8BBd35D73EDa77eae1b413) into etherscan.io, specifically within the Sepolia network, the coordinator gains comprehensive visibility into all UTHM token transactions. This

transparency feature enables the organization to effectively oversee activities like transfers and receipts, ensuring proper management of the funds.

Moreover, students can view the amount of UTHM tokens in their apps, which is based on their smart contract address within the MetaMask application. To receive and transfer tokens, students need to install the MetaMask application from the Google Play Store. When students wish to transfer or use UTHM tokens, they can conveniently scan the QR code on the seller's or recipient's MetaMask account. The application will then redirect them to the MetaMask application, automatically navigating to the sending page for token transfers.

Figure 11, Figure 12, Figure 13, Figure 14, and Figure 15 show the process of giving or transferring the token to the students that book and attend the appointment.

Figure 11: Data of student list on the reward page

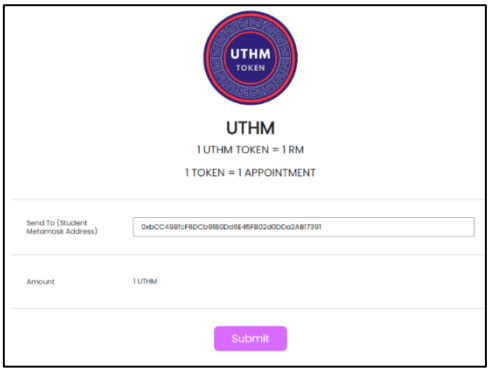


Figure 12: Enter the student's smart address contract to start transferring the token

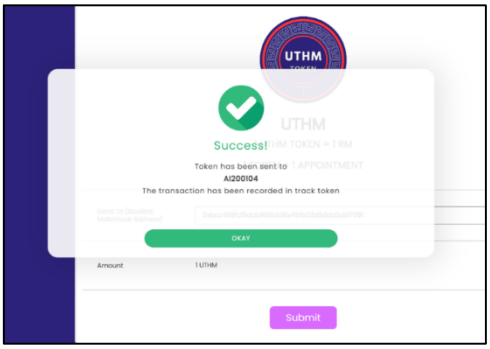


Figure 13: Screen popup tells that the token has been successfully sent.

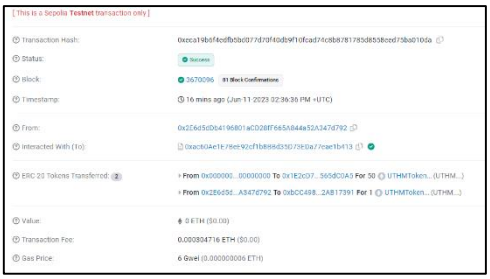


Figure 14: Track the information from the Sepolia etherscan.io

Transaction Hash	Method	Block	Age	From		To	Value	Txn Fee
0x5789e95c4cd9e3fb4...	Transfer	3671042	50 mins ago	0xbCC498...2AB17391	OUT	0xac60Ae...eae1b413	0 ETH	0.00009462
0x1ca34c0ff64a715c2f...	Transfer	3497786	26 days 4 hrs ago	0x87c9B0...CF3cd9Bf	IN	0xbCC498...2AB17391	0.5 ETH	0.0000945

Figure 15: Track the student's activities in the Sepolia etherscan.io

Figures 16, Figure 17, and Figure 18 depict the process of transferring tokens to sellers or other users within the application. The earned tokens can be used by students to make purchases from UTHM's affiliated shops by scanning a valid QR Code using the MetaMask application, which seamlessly redirects to the transaction page.

Figure 16: Interface the number of tokens for students in the application

Figure 17: Scan the QR Code

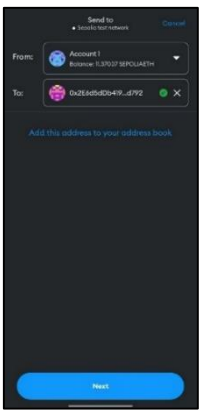


Figure 18: The application redirects to the transaction page in the MetaMask application

6. Result and Discussion

The implementation phase includes functionality testing and user acceptance testing. The results and discussion of the proposed system, including test cases and user acceptance testing results, are presented in this section. It focuses on the system specifications and requirements, ensuring that all requirements are thoroughly tested and met. This approach aids in effective testing and requirements tracing.

6.1 Test case result

The test case is designed to evaluate the functional requirements of the proposed system. It involves comparing the behaviour and functionality of the application against the expected results to ensure its effectiveness and proper functioning. Table 6 provides a summary of the test case results derived from the functional testing process.

Table 6: Summary of test case

Test ID	Requirement ID	Test Case Description	Result (Pass/Fail)
TC01	FR01	Login	
TC01-01	FR01-01	The application and system can verify the users.	Pass
TC01-02	FR01-02	The application should redirect validated users to the respective homepage or dashboard based on their identity.	Pass
TC01-03	FR01-09	The application and system shall let the user to logout to the login page.	Pass
TC02	FR02	Manage Profile	
TC02-01	FR02-01	The application shall provide the information for all users.	Pass
TC02-02	FR02-02	The application shall save the current data if the lecturers are not complete the form.	Pass
TC03	FR03	Manage Slot	
TC03-01	FR03-01	The application shall disable the update button if 5 workdays are not inserted.	Pass
TC03-02	FR03-02	The application shall disable the add button when 5 workdays are inserted.	Pass
TC04	FR04	Book Appointment	
TC04-01	FR04-01	The application should let the students enter the integer number in the number of the student section.	Pass
TC04-02	FR04-02	The application shall redirect to the booking successful page.	Pass
TC04-03	FR04-03	The application shall allow the lecturer to reject the appointment	Pass

Test ID	Requirement ID	Test Case Description	Result (Pass/Fail)
TC04-04	FR04-04	The accepted appointment must be inside the attendance page on the lecturer side.	Pass
TC05	FR05	Manage Appointment	
TC05-01	FR05-01	The application shall cancel the appointment when the lecturer cancels the appointment.	Pass
TC05-02	FR05-02	The application should redirect to the update booking page when clicking the update button.	Pass
TC05-03	FR05-03	The application should delete the appointment when completing the appointment by pressing the attend/absent button.	Pass
TC06	FR06	Manage Live Chat	
TC06-01	FR06-01	The application shall display sending messages on the chat page.	Pass
TC06-02	FR06-02	The application shall display receiving messages on the chat page.	Pass
TC07	FR07	Manage Attendance	
TC07-01	FR07-01	The application shall differentiate the attend or absent appointment based on the button given.	Pass
TC08	FR08	Manage Appointment History	
TC08-01	FR08-01	The application shall show the status of attending or absent from the meeting.	Pass
TC08-02	FR08-02	The application shall show the UTHM token gained.	Pass
TC09	FR09	Manage User	
TC09-01	FR09-01	The system shall let the coordinator enter specific information to register the users	Pass
TC09-02	FR09-02	The system shall not allow the coordinator to submit an incomplete form to create an account for the student and lecturer.	Pass
TC09-03	FR09-03	The system shall allow the coordinator to delete the user information by clicking the trash icon.	Pass
TC10	FR10	Manage Reward	
TC10-05	FR10-05	The system shall let the coordinator submit 1 UTHM token for 1 attended appointment.	Pass
TC10-07	FR10-07	The system shall let the coordinator track the transaction that has been made.	Pass

Test ID	Requirement ID	Test Case Description	Result (Pass/Fail)
TC10-11	FR10-11	The application shall display the current amount of tokens on the reward page for each student.	Pass
TC10-12	FR10-12	The application shall open the Scan QR code after the student clicks the Scan QR button.	Pass

6.2 User Acceptance Testing

User acceptance testing is then performed with the anticipated user. The system's module and interface design are the aspects that are being tested. The test will be conducted via a questionnaire form that utilized Google Forms. There is a coordinator who will test the system from the website. Students and lecturers will test the application to get their satisfaction. The main module of the system that will test is book appointment, manage appointment and manage reward module. Each of the modules has obtained feedback from the user.

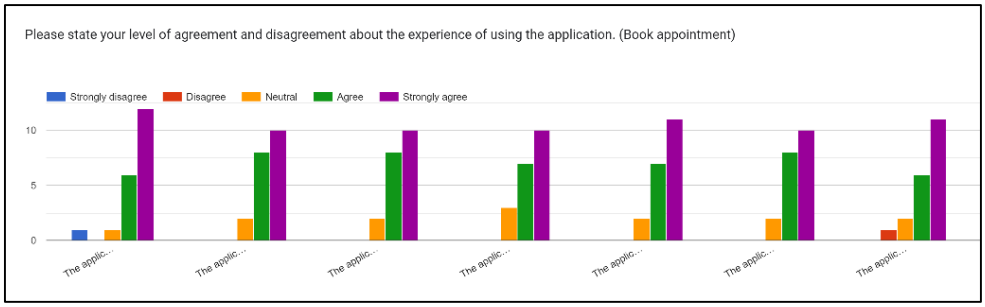


Figure 19: Book appointment module feedback user

Users were required to provide their opinion on a scale of 1 to 5, ranging from strongly disagree to strongly agree, regarding their experience with the application. The technology utilized for booking appointments with lecturers was well-received by all students. The booking section effectively displayed available slots based on the lecturer's schedule, and the differentiation of slots by day was clear. Once a slot was booked by a student, it became unavailable until the appointment was either rejected or cancelled by the lecturer. Overall, the system's functionality received positive feedback from the three users, as indicated by their frequent selection of the highest rating (5) to express strong agreement with the performance of each module.

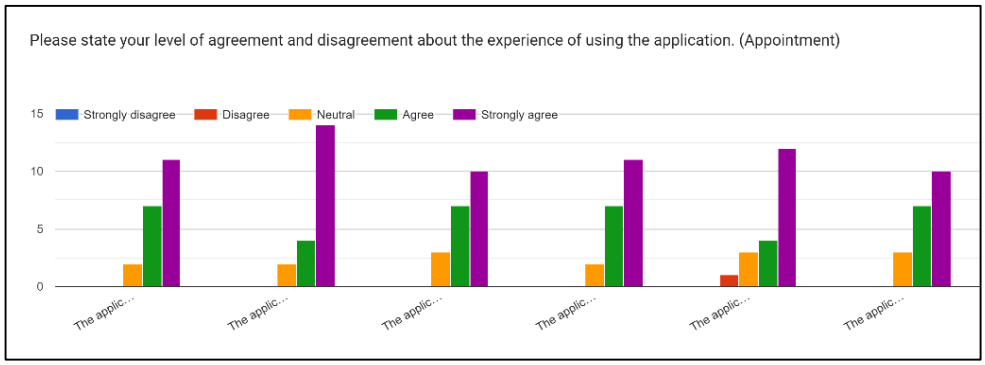


Figure 20: Manage appointment module feedback user

The manage appointment feature provided students and lecturers with the capability to update or cancel their appointments. Overall, the system received positive feedback from both students and lecturers regarding the functionality of this module, as indicated by many of them expressing agreement or strong agreement with its performance.

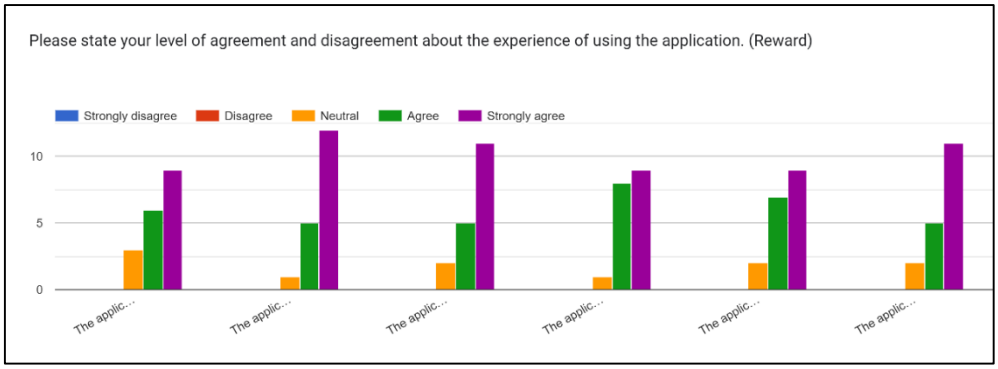


Figure 21: Manage reward module feedback user

1

Furthermore, the utilization of Web3 technology has been well-received and embraced by all students, as they actively engage with the MetaMask wallet to generate UTHM tokens. Additionally, students are gaining valuable knowledge about cryptocurrencies and the Web3 or blockchain concept, as they are required to mine or acquire SepoliaETH to facilitate the transfer of UTHM tokens. The scan QR code functionality has also been positively received, as no negative comments or feedback were received regarding this feature. As a result, the implementation of WeMeet Dapps has successfully achieved its objectives and goals in addressing the issues faced by students and lecturers within the university context.

7 Conclusion

The project has been developed using Android technology, specifically leveraging the Flutter framework by Google for efficient translation of the native language. In addition, Bootstrap was employed to create a web-based system that facilitates effective management by the coordinator. The server-side functionalities and database operations were implemented using Node.js and MySQL, ensuring optimal handling of requests and secure storage of information. Throughout the project's development, various technologies were utilized to enhance its functionality and performance. To create the UTHM token, the Hardhat framework was employed, enabling the generation and management of the token within the project. It is essential to prioritize the responsiveness of the application and system to minimize any potential issues or defects during usage. Furthermore, adopting a reliable cloud computing solution is crucial for hosting the server and ensuring uninterrupted accessibility via an internet connection. Considering the current availability of the application on Android technology, future development efforts should encompass iOS technology to cater to the university's student population utilizing iOS devices.

Acknowledgement

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

Appendix A

Please find the appended class diagram displayed below.

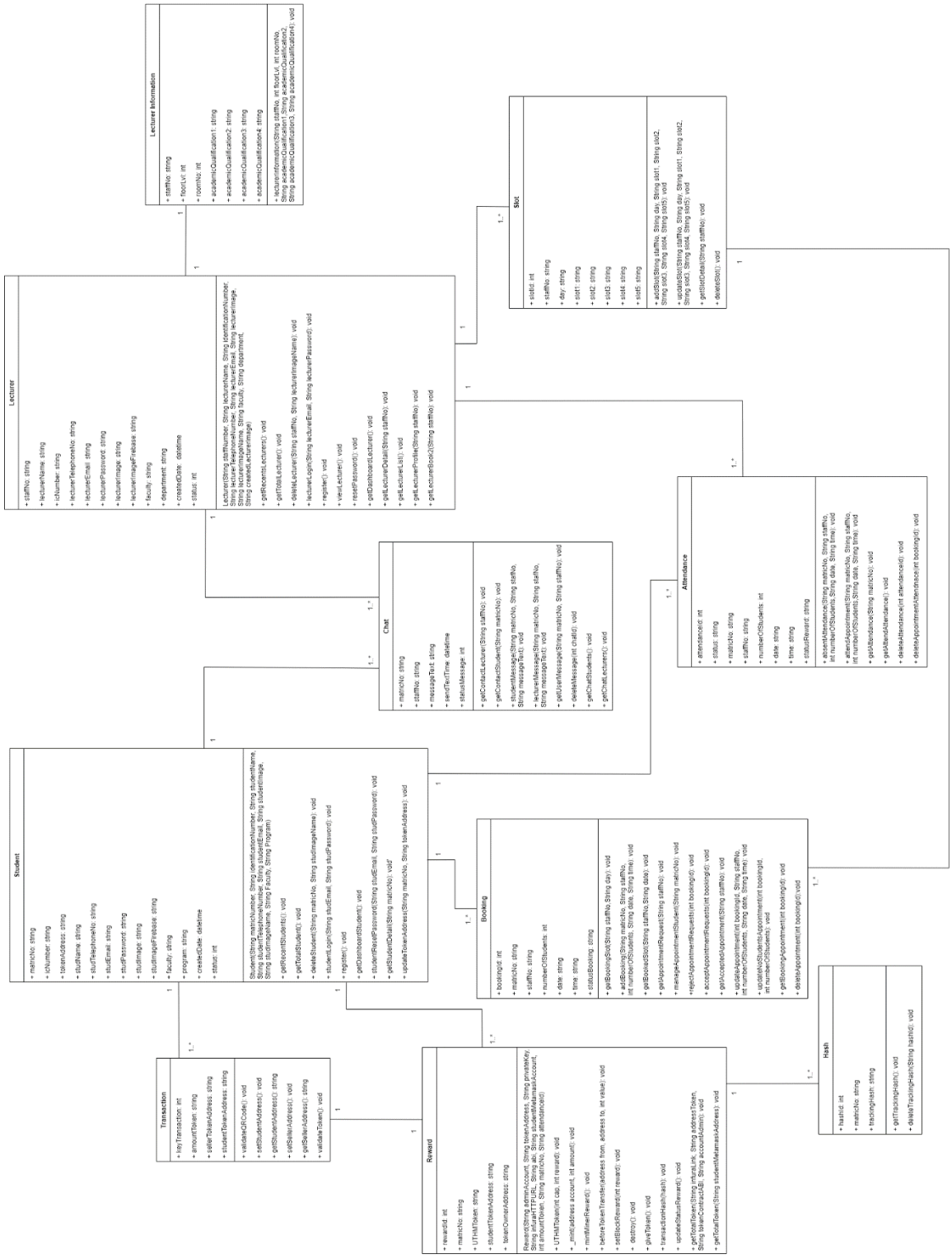


Figure 22: Class Diagram for the project

References

[1] A. S. Gillis, & Bernstein, C. “What is a decentralized application (DAPP)? – IoT. IoT Agenda”

[2] J. Frankenfield. “What is ethereum and how does it work? Investopedia.”

[3] J. X. Yang. Liu, and X. Li, “Research and analysis of blockchain data,” J. Phys. Conf. Ser., Vol. 1237, No. 2, P. 8, 2019, Doi: 10.1088/1742-6596/1237/2/022084.

[4] R. Cole, M. Stevenson, and J. Aitken, “Blockchain technology: implications for operations and supply chain management,” Pp. 1–34, 2019, [Online].

[5] P. Wackerow. Introduction to Dapps and smart contract technologies - pintu academy. Introduction to DApps and Smart Contract Technologies - Pintu Academy. 2023

[6] J. Chen, “Android Operating System (OS): Definition and how it works. Investopedia.”

[7] Y. Wurmser. “Mobile app versus mobile website statistics: 2020 and beyond. JMango360.”

[8] A. Hossen. Mobile website vs. Mobile App (application) – which is best for your organization? HSS Web Technologies

[9] A. Dennis. Wixom, B. H., & Roth, R. M.,Systems analysis and design. 5th ed. New Jersey