Applied Information Technology And Computer Science Vol. 5 No. 1 (2024) 071-090

AITCS

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

A Bus Schedule Application Using Fingerprint

Siti Hanisah Lakman[1], Nur Ziadah Harun[1]*

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

DOI: https://doi.org/10.30880/aitcs.2024.05.01.005
Received 04 August 2023; Accepted 23 May 2024; Available online 30 August 2024

Abstract: A bus schedule mobile application with fingerprint verification is developed for UTHM students who use the shuttle bus service to make getting to class easier. Students previously encountered challenges such as missing the bus owing to a misunderstanding between the bus driver and the student. This proposed application acts as a communication medium between the bus driver and the student since it includes a location module that allows the bus driver to share the bus's current location and the student to track it. The scheduling module where the bus timetable is provided, does help students arrive on time. Fingerprint characteristics are used to enhance the user experience by reducing login times and enhancing security. The app follows the prototyping model for development. Overall, this application makes it easier for students to obtain the bus schedule, and the usage of fingerprints improves security since fingerprint patterns are unique to each user and difficult to duplicate. Fingerprints are also typically consistent over the course of a person's life, making them a dependable form of identification.

Keyword: Bus Schedule Application, Fingerprint, Mobile Application

1. Introduction

The 450,000 yellow school buses are the most visible and safe mode of school transportation, as well as the nation's largest form of public transportation[1]. Despite the fact that many students drive themselves to and from school, there are still students who use school transportation[1]. Sunway University has implemented a number of programs to help reduce greenhouse gas emissions by encouraging students to use public transportation. This demonstrates one of the advantages of university transportation, not only for students but also for the environment[2].

Furthermore, Universiti Tun Hussein Onn Malaysia (UTHM) provides a bus shuttle for students used where they can view the bus location at UTHM Public Shuttle Tracking by KATSANA[3]. The UTHM bus tracking system is used by both students and bus drivers. This system is primarily concerned with how the bus driver and students interact, as it provides an up-to-date location while allowing students to track the current bus location. This is a system that operates in real-time. However, students find the UTHM bus tracking system challenging to use because they must constantly connect to the internet and wait for the page to load because it is only a web-based system. This requires them to wait for a while and consider what it would be like to have a bad internet connection. The UTHM bus

*Corresponding author: nurziadah@uthm.edu.my

tracking system occasionally is not available or does not respond at certain time because of the high volume of service requests.

In this project, a bus schedule mobile application is created by implementing fingerprints as an authentication method instead of just using a username and password. Mobile applications, on the other hand, tend to have better functionality due to the presence of supportive features in mobile phones[4]. Furthermore, this application provides high security by employing a biometric system specifically a fingerprint scanner, as the second method of login. Because fingerprints are physical characteristics, the data of the user cannot be easily stolen, and the data obtained from biometric security is quite accurate, there is no doubt about the user's identity. Aside from that, because one of the users is a bus driver, they are unfamiliar with the login page that requires a username and password due to their age, and they genuinely think that creating a password is already difficult. As a result, fingerprint is the best option. Other than that, this application implements One Time Password (OTP) as an extra layer of security because OTPs are intended for one-time use, they are immune to replay attacks in which an attacker intercept. Moreover, the current location of the bus is displayed in this bus schedule application so that users, particularly students, can easily track where the bus is. To allow for better time management, a tracking application that can be implemented as a mobile application for students is required.

The objectives of the proposed application are first to design a bus scheduling application with fingerprint feature. The second is to develop a bus scheduling application with a fingerprint feature and the last objective is to test the functionality of the bus schedule application through user acceptance testing and security testing

The proposed application has three target users: admin, bus driver, and student. This application is divided into three modules as shown in Table 1: sign up ,login, and bus management.

Table 1: Application Scopes and User Scopes

User	Module	Description
Admin	Bus management	Display bus schedule
Bus driver	i. Sign Up ii. Login iii. Bus management	i. The bus driver must register to the application first. ii. The bus driver must log into the application by entering the username and password, or they can use the fingerprint scanner to make the process faster and easier. iii. The bus driver is available to update the bus's location.
Student	Bus management	i. Student can easily view the bus schedule ii. Track the bus's current location.

2. Related Work

2.1 Android technology

The proposed application is available for the Android operating system only. Android was created by Google and later Open Handset Alliance as a software platform and operating system for mobile devices and based on the Linux kernel [5]. Android is an open source and is designed for smartphones, tablets, and Android wearables like watches. Since its release, Android has been updated on a daily basis. These updates to the base operating system are primarily focused on bug fixes as well as the addition of new features to provide a more comfortable environment. Android is gaining ground faster than the iPhone. Its platform has expanded dramatically in the last two years. There are some differences between Android and iPhone in terms of syncing, security, and other factors. In terms of security, Android outperforms the iPhone because each application runs independently and has pre-defined permissions and authorizations for each feature. While iPhone users have some privileges that limit the addition of new software, each application runs on a single UNIX kernel[6]. As a result, if a problem occurs while running the application, the entire system may be affected.

2.2 Fingerprint Authentication

The confidentiality, integrity, and availability of all information were guaranteed by information security. There are numerous techniques available to help with information security management improvement. Fingerprint technology is applied in this proposed application since its more secure because everyone has a unique fingerprint. A fingerprint is a representation of the friction ridges on all or part of the finger[7]. There are numerous benefits to using fingerprint biometrics. The first benefit is high security and assurance because it verifies a tangible as something the user has as well as something the user is. Passwords were most likely compromised as a result of a data breach. Following that, the user experience is simpler and quicker by simply placing a finger on a scanner and unlocking an account in seconds rather than typing a long password. There is a one in 64 billion chance that someone's fingerprint will exactly match another person's, according to [8]. This ensures that fingerprints cannot be replicated using current technology.

2.3 One-Time Passwords (OTP)

Moreover, a one-time password (OTP) is also applied in this proposed application. OTP is a string of characters that is generated automatically for each authentication session [9]. Once the password has been used, it is no longer valid, and any subsequent attempts to use the same password for authentication will fail. Short messaging service (SMS) is used in this proposed application to provide an OTP via text. Employing OTP numbers for applications improves security, convenience, and protection against password-based attacks and unauthorized access. OTPs are simple to add into an application's authentication flow, and their flexible generating methods make them useful for a wide range of use cases.

2.4 Review of Existing System

There are three existing systems to be compared which are the RFID-Based Campus Bus System Using Internet of Things (IoT), IoT Based MRT Feeder Bus Tracking System, and the Delivery Management System.

2.4.1 RFID-Based Campus Bus System Using Internet of Things (IoT)

The first existing system, the RFID-Based Campus Bus System Using Internet of Things (IoT) [10] aimed to express and prove the role of Radio-Frequency Identification (RFID) technology in improving student campus bus systems, as well as to discuss the use of a GPS module. RFID used a card and reader combination for identification approval [11] and saves it into the database. The RFID card contained a code, and the card was then attached to a physical object [11]. The RFID tags are used before boarding the bus to monitor the movements of students who enter and exit the bus. The system then includes features that allow students to track, and predict bus arrival times, and measure total distance for the company’s purpose by using a GPS module. The system is web-based and contains two distinct web pages, public and private in order to differentiate the authorization between the student and the admin.

2.4.2 IoT Mass Rapid Transit (MRT) feeder bus tracking

The next system is an IoT Mass Rapid Transit (MRT) feeder bus tracking system to address feeder bus issues. According to this paper, MRT is used to transport large numbers of passengers from one location to another, and feeder buses serve as medium transportation between MRT stations and specific bus stops [12] . This system monitors the bus's real-time location, the most recent bus stop status, and the estimated time of arrival (ETA). The GPS is also used in this system to continuously update the bus location on Google Maps. Furthermore, this system is developed in the form of a web application, and it consists of two web pages: real-time location and the most recent bus stop. According to the MRT feeder bus tracking system, this system does not have a login page because it is open to the public. It

would be preferable if they provided some security features, even if only a login page, to make the system more secure.

2.4.3 Delivery Management System

The last existing system, Delivery Management System[13], was developed to assist drivers and the Happy Moments Florist staff store in efficiently connecting to deliver product orders to recipients. Previously, the arrangement process was done manually on a spreadsheet. Aside from that, the delivery schedule and task completion are not handled in a systematic manner, where the order details and delivery destination are written on paper for the driver to deliver the order. This causes a few issues, including the possibility of the paper going missing, the failure to determine the current driver location, and the difficulty to update the status of the product, whether it has been delivered or not. As a result, this system was created in order to make everything run more smoothly and easily. This system's main feature is delivery scheduling and order management. Where the system can automatically calculate the shortest path to ensure that the order is shipped within the specified timeframe. This feature allows the driver to take alternate routes if there are traffic jams, accidents, or other factors affecting delivery time. Furthermore, the login module was used by the system to generate user authentication for the system's admin and drivers. The security measures in place are intended to keep unauthorized people from accessing the system. To be authorized, all actions performed within the system require administrator approval, and the system will display error messages to alert the user of invalid input. This system is designed for two types of systems: web-based systems for administrators and mobile apps for drivers.

This proposed application is contrasted with three other systems: RFID-Based Campus Bus System Using Internet of Things (IOT), IoT-Based MRT Feeder Bus Tracking System, and a Delivery Management System. A few features and security measures are included in the comparison as illustrated in Table 2.

Table 2: Comparison Table

	RFID-Based Campus Bus System Using Internet of Things (IOT)	IoT-Based MRT Feeder Bus Tracking System	Delivery Management System	Proposed Application: A Bus Schedule Application Using Fingerprint
Register/Login	✓	x	✓	✓
Real-time Location	✓	✓	x	✓
Fingerprint	x	x	x	✓
One Time Password	x	x	x	✓
User	Students	Public	Rider	Students
System type	Web app	Web app	Web and mobile app	Mobile app

Referring to Table 2, except for the Delivery Management System, all existing systems can track the current bus location in real-time. The only exception is the IoT Based MRT Feeder Bus Tracking System, which lacks a login and registration page. Aside from that, only the proposed application provides fingerprint and OTP to increase system security. Furthermore, the IoT Based MRT Feeder Bus Tracking System is designed for public use, whereas the Delivery Management System is designed for rider use, and the other two systems are designed for student use. Finally, the proposed application is intended for a mobile application, while Delivery Management System is implemented for both web and mobile app and lastly the others are intended for a web application only.

3. Methodology/Framework

Methodology aids in the management and control of system development by breaking down large problems into sub-problems. This proposed application was developed using the prototype methodology. Prototype methodology is a systems development process in which a prototype is implemented, tested and the improved until an acceptable output is obtained from which the entire system or product can be developed. In comparison to the traditional methodology, where each phase is fully finished before moving to the next stage, the traditional methodology makes it more difficult to modify something that was not discovered in the system [14]. This is why the prototyping model is been chosen. Figure 1 shows the prototype model.

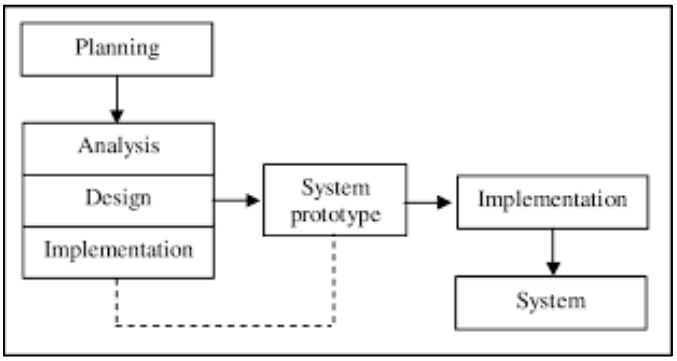

Figure 1: Prototyping Model

3.1 Planning and Analysis Phase

Early in the development of a Bus Schedule Application using Fingerprint, the planning and analysis phase is completed. The proposed project and the project title were completed during this phase. Following that, all relevant information about the proposed project proposal was gathered and searched. Besides, the problem statements of the existing system were discussed. In order to address the problem statements, the objectives of a Bus Schedule Application using Fingerprint authentication have been discussed and created. Following that, the project's scope was determined in order to make it easier to meet the needs of the users. To make the benefits of the proposed project clear, the significance of the project is discussed. The information is gathered by observing the existing system

3.2 Design Phase

The design phase is where a Bus Schedule Application using Fingerprint is designed in order to meet the user requirements. During the design phase, all the results from the analysis phase are used to meet the user needs, which are the flow of the application and the database. Furthermore, the database for the proposed application is designed in this phase, which is ERD and related notation is intended to demonstrate the flow of the application and avoid redundancy data.

3.3 Implementation

The implementation phase consists of carrying out the activities of a Bus Schedule Application using Fingerprint authentication that were planned and designed during the project planning and design phases. The user requirements, functional requirements, non-functional requirements, and implementation process of the proposed application are discussed in this phase. Furthermore, the application will be built with Android Studio, which offers the quickest tools for developing apps for any Android device. Aside from that, the firebase real-time database is used to keep and sync data in real-time between users.

3.4 System Prototype

At this step, the proposed application's prototype is delivered to the user for preliminary testing according to Table 3. It is beneficial to investigate the strengths and shortcomings of the performance model. In order to improve the prototype, user feedback and recommendations are gathered.

Table 3: Application Testing

Category	Expected result	Actual Result
Sign up	Bus driver can easily sign up to the application	Pass/Fail
	The user should be able to create a strong password	Pass/Fail
	Bus driver get the OTP number	Pass/Fail
Login	Error messages appear such as “Incorrect password” or “incorrect username”	Pass/Fail
	The user should be able to input email and password	Pass/Fail
	The user should be able to login using the correct email and password	Pass/Fail
	The user should not be able to login with an invalid email and password	Pass/Fail
	The user should be able to login using a valid fingerprint	Pass/Fail
	The user should not be able to login using an invalid fingerprint	Pass/Fail
Bus schedule	User should be able to click on the button to view the bus schedule	Pass/Fail
	User should be able to view the bus schedule	Pass/Fail
Bus location	Bus driver can share their current location	Pass/Fail
	Student can track the bus location	Pass/Fail

Following that, the present prototype is refined in response to the comments. With the new information provided by the user, an improved prototype is created. The improved prototype is examined in the same manner as the prior prototype. This method is repeated until all of the user's criteria are met.

4. System Analysis and Design

In this section, the system architecture, data flow diagram, database design, and system flowchart are been discussed.

4.1 System Architecture and Flowchart

Figure 2 shows the system architecture for the proposed application and Figure 3 shows the system flowchart.

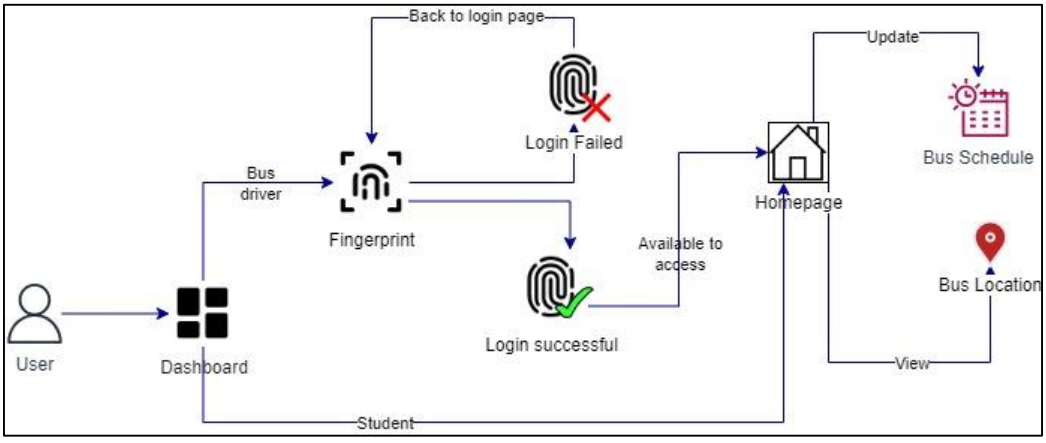

Figure 2: System Architecture

The system architecture of a Bus Schedule Application using Fingerprint authentication is depicted in Figure 3. To begin, all users will see the dashboard page, after which students may simply click on the menu option that they want to access; homepage, location, and schedule because they do not have to login to the application. Aside from that, the bus driver must login to the application because they

will be sharing their present location, and authentication is required to secure the data from unauthorized usage. If the login fails, the bus driver will return to the login page and attempt again. They can view the bus schedule and share the location if the login is successful.

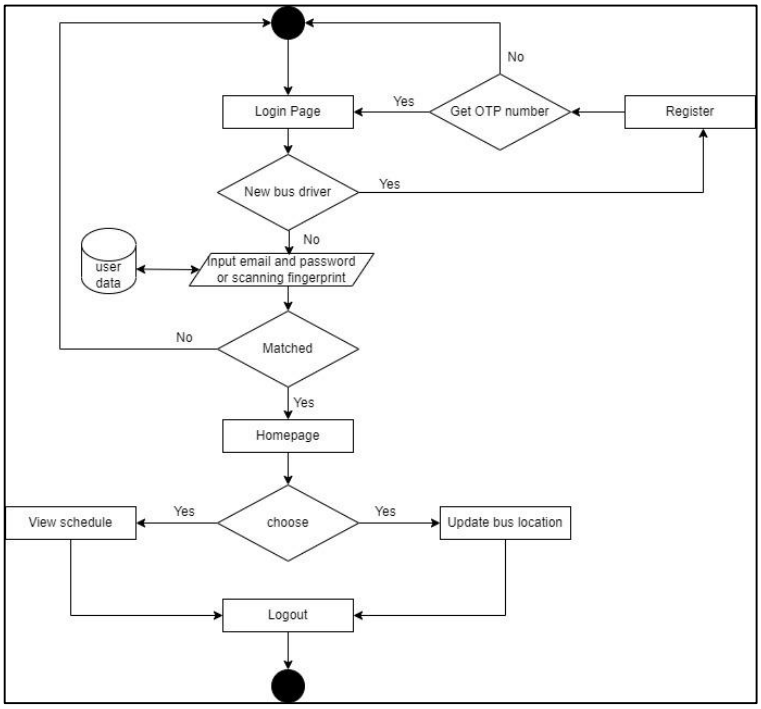

Figure 3: Flowchart for The Bus Driver

Referring to Figure 3, firstly, the bus driver has to click on the login button that, then if they do not have an account, they must register. The registration is successful if they receive an OTP number, but if not, they must try again. If they already have an account, they can simply login using their email and password or fingerprint, then the database checked the user authentication if matched then they have successfully entered the homepage while if the fingerprint does not match with the database, then they need to try login again. After successful login, bus driver can choose whether they want to update the bus location or view the bus schedule. Lastly, they just need to log out from the application to protect their authentication and authorization.

4.2 Data Flow Diagram (DFD)

A data flow diagram depicts how information flows through a process or system, including data inputs and outputs, data storage, and the various subprocesses that the data passes through.

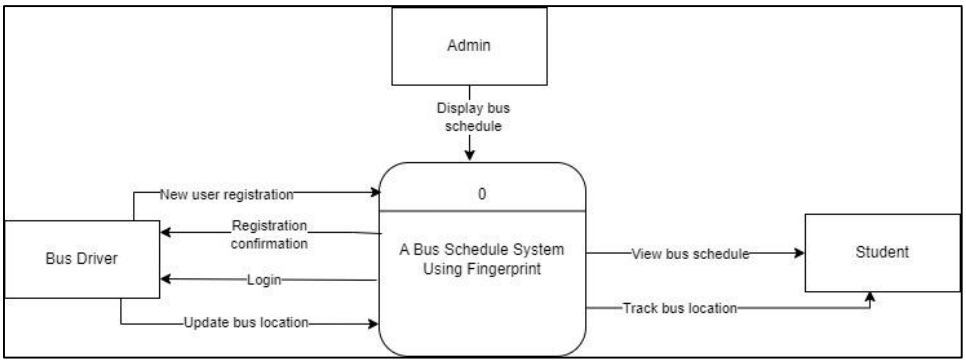

Figure 4: Data Flow Diagram (Context Diagram)

Figure 4 shows the context diagram which the flow of the process in the Bus Schedule Application Using Fingerprint authentication. This application involves three entities which is admin, students, and bus driver. Admin is the person who manage the bus schedule. Bus driver needs to login to the application to update the bus location and access the bus schedule after they successfully register. Besides, student also can access the bus schedule and track the bus location.

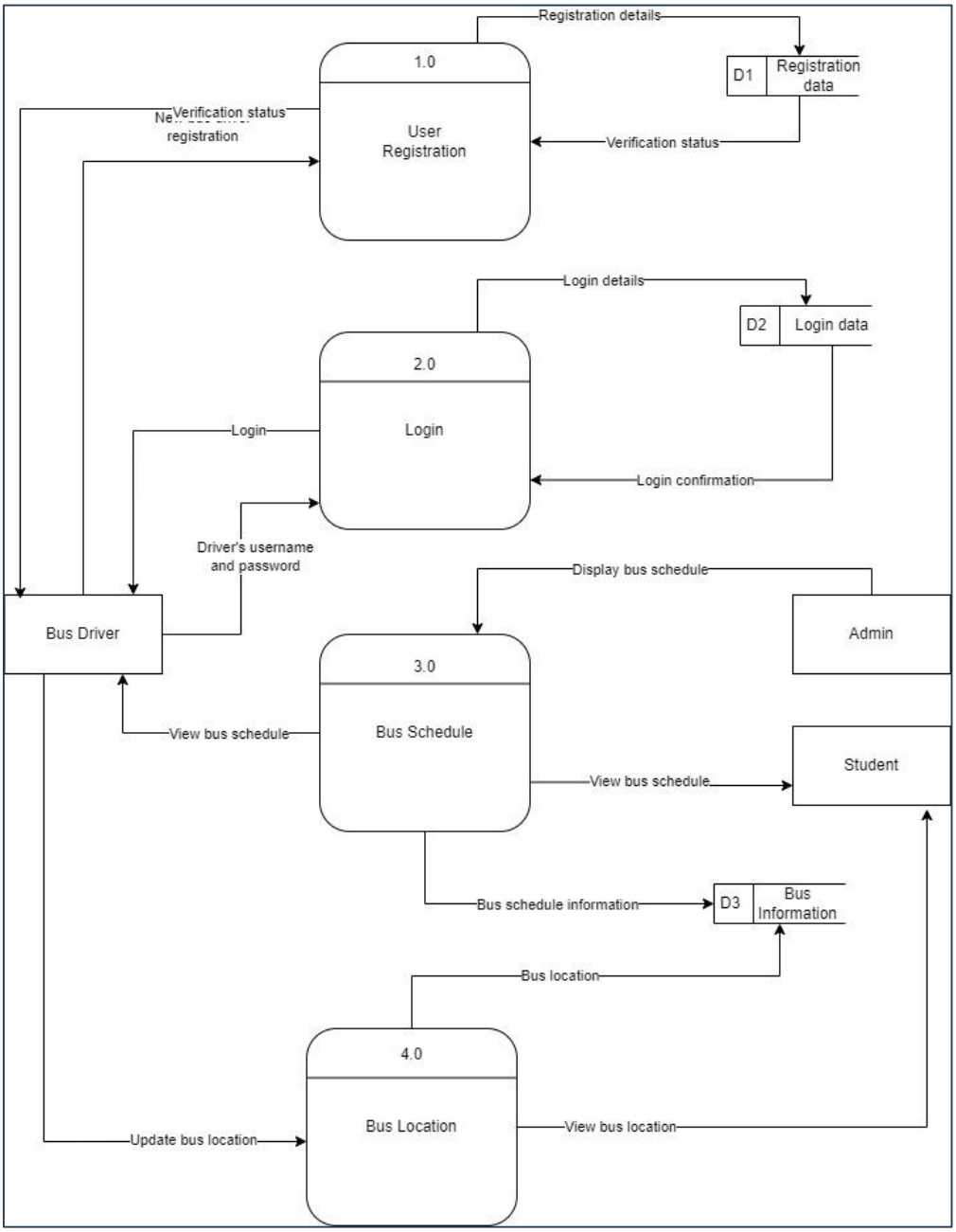

Figure 5: Data Flow Diagram (Level 0)

Data Flow Diagram (DFD) level 0 as shown in Figure 5, it shows all the processes that existed in the proposed application. It contains four processes which are registration, login, bus schedule and bus location. Aside from that, this application has three data stores which are registration data, login data and bus information. Registration data stores the details of the bus drive while login data stores the login details of the users and send the login confirmation. Next, bus location and bus schedule are store in bus information database.

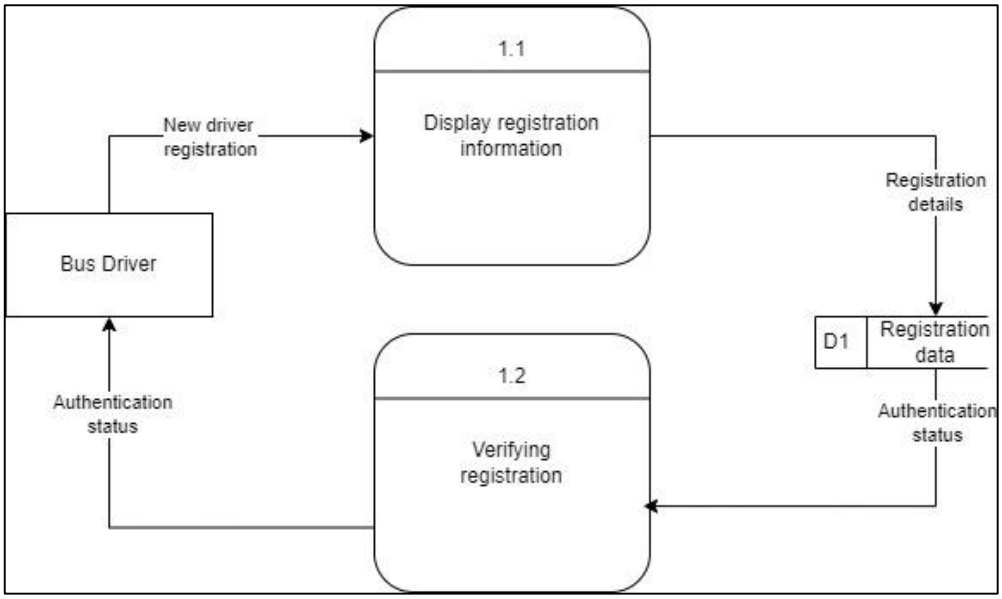

Figure 6: Data Flow Diagram (Level 1) – Bus Driver Registration

Figure 6 shows DFD level 1 for bus driver registration. Bus driver must insert the details needed such as bus number, email, password, and phone number. Then, the bus driver will get the authentication status.

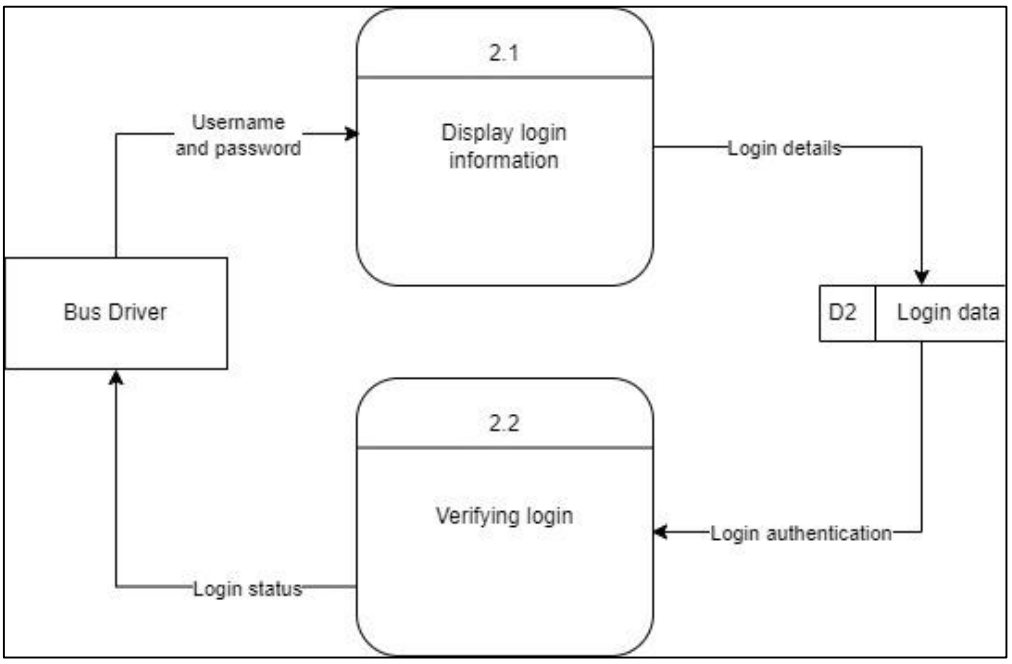

Figure 7: Data Flow Diagram (Level 1) – Login

In the DFD level 1 login process, only the bus driver has to login to the application as shown in Figure 7 by entering the correct username and password or the other option is scanning their fingerprint. Next, the database will verify whether the input match with the database registered or not.

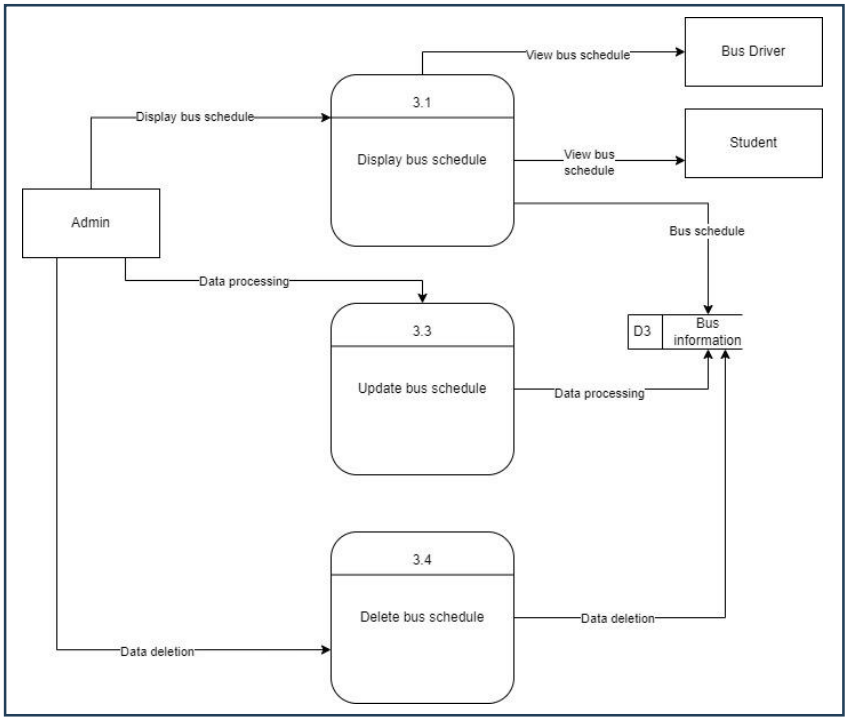

Figure 8: Data Flow Diagram (Level 1) – Bus Schedule

There are three processes consisting in DFD (Level 1) Bus Schedule as shown in Figure 8. All the process has been made by the admin where the admin can update, delete, and display the bus schedule. Information on the bus schedule is then stored in bus information databases. The other user which is bus driver and student has the authorization to view the bus schedule that they can access in the application.

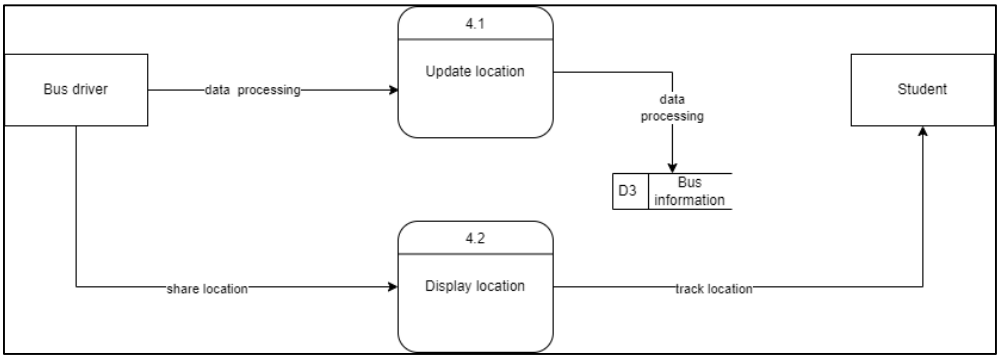

Figure 9: Data Flow Diagram (Level 1) – Bus Location

Figure 9 shows DFD (Level 1) bus location which includes the bus driver and student because the bus driver updates the current location and releases it to the student to access the location. The bus location is stored in bus location databases.

4.3 Entity Relationship Diagram (ERD)

Aside from that, the database design for the proposed application, which is ERD is intended to demonstrate the flow of the application and avoid redundancy of data. To ensure that a perfect application is developed, all modules and features are listed.

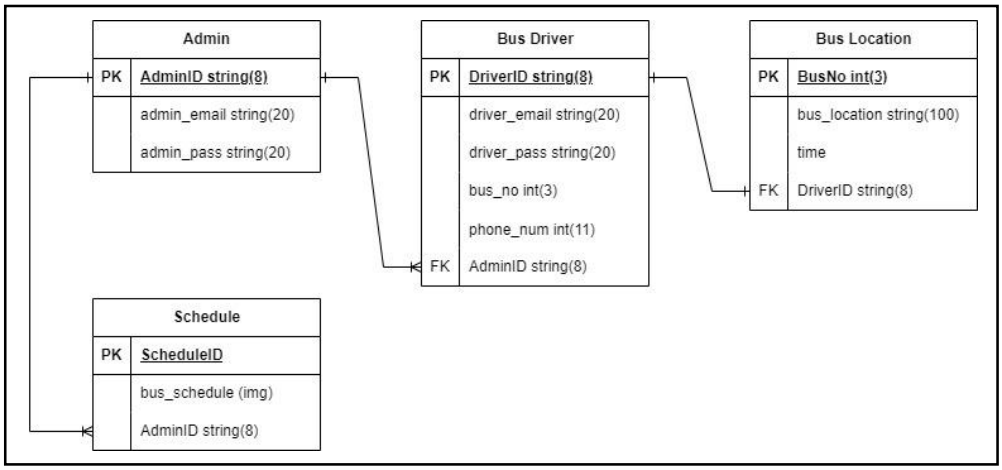

Figure 10: ERD of the Proposed Application

According to Figure 10, there are four entities which are admi , bus driver, bus location, and schedule. The cardinality of admin and bus driver is one bus driver can manage one or many bus driver information. Additionally, one bus driver can only share one bus location at one time because every bus driver has been assigned with one bus only. Lastly, one admin can manage one or more bus schedule.

5. System Implementation

This section discussed the results of module implementation by providing a part of the code and the interface of the module

5.1 Implementation of Security Features

There are four security features that has been implemented in this proposed application which are one-time passwords (OTP), fingerprint biometric, strong password and input validation.

Figure 11 depicts the user experience for requesting an OTP, where they must first provide their preferred phone number, and the other one depicts the user interface for entering the OTP that has been sent to their mobile phone.

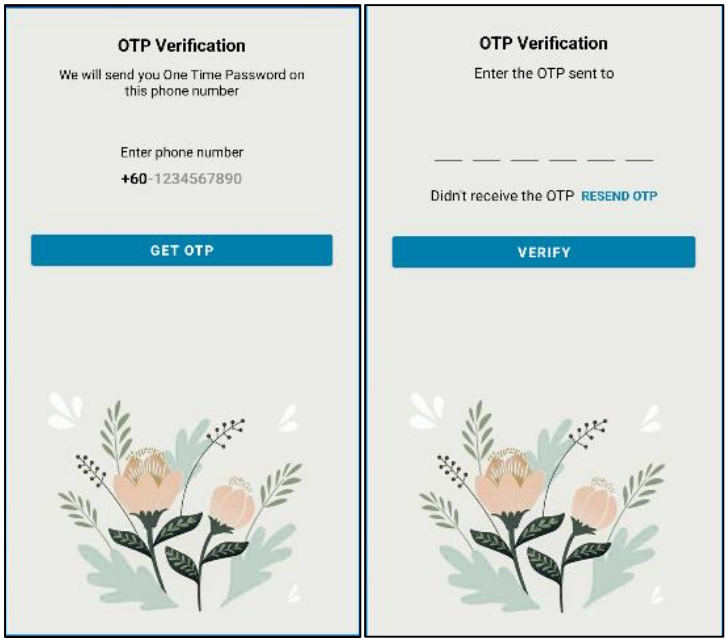

Figure 11: Interface for Getting and Verifying OTP

Figure 12 shows the code for the user to get an OTP number. The user must enter their chosen phone number, which will receive the OTP number and then click on the get OTP button.

```
getOTPbtn.setOnClickListener(new View.OnClickListener() {  //code for user to get OTP
    @Override
    public void onClick(View view) {
        if (inputPhone.getText().toString().trim().isEmpty()) {
            Toast.makeText( context: SendOtp.this,  text: "Enter Phone Number", Toast.LENGTH_SHORT).show();
            return;
        }
        progressBar.setVisibility(view.VISIBLE);
        getOTPbtn.setVisibility(view.INVISIBLE);

        PhoneAuthProvider.getInstance().verifyPhoneNumber(
                phoneNumber: "+60" + inputPhone.getText().toString(),
```

Figure 12: Code of User to Get OTP

Furthermore, Figure 13 shows the code of verify OTP page once the user clicks on the get OTP button. Verify OTP page is where user input their OTP number there.

```
    @Override
    public void onCodeSent(@NonNull String verificationId, @NonNull PhoneAuthProvider.ForceResendingToken forceResendingToken) {
        progressBar.setVisibility(View.GONE);
        getOTPbtn.setVisibility(View.VISIBLE);
        Intent intent = new Intent(getApplicationContext(), VerifyOtp.class); // once they enter the phone number, they will go to verify otp page
        intent.putExtra( name: "phone", inputPhone.getText().toString());
        intent.putExtra( name: "verificationId", verificationId);
        startActivity(intent);
    }
}
```

Figure 13: Verify OTP Page

Next, if the user did not receive any OTP number after they click on the get OTP button, they have to click the resend OTP line and the new OTP number will be send after 60 seconds as shown in Figure 14 the timeout for each OTP is 60 seconds.

```
    PhoneAuthProvider.getInstance().verifyPhoneNumber(
            phoneNumber: "+60" + inputPhone.getText().toString(),
            timeout: 60,
            TimeUnit.SECONDS, //once sent, user can't get new code for the next 60 seconds
            SendOtp.this,
            new PhoneAuthProvider.OnVerificationStateChangedCallbacks(){

                @Override
                public void onVerificationCompleted(@NonNull PhoneAuthCredential phoneAuthCredential) {
                    progressBar.setVisibility(View.GONE);
                    getOTPbtn.setVisibility(View.VISIBLE);
                }
```

Figure 14: Code for Creating New Code After 60 Seconds

Furthermore, if the user has previously obtained the OTP number and put it into the verify OTP page, the application will determine whether the entered OTP is still valid or not because every OTP number has a timeout. If the OTP numbers are no longer valid, the error message "Please enter valid code" will show up as shown in Figure 15.

```
buttonVerify.setOnClickListener(new View.OnClickListener() {  //code for user to validate OTP received
    @Override
    public void onClick(View view) {

        if (inputCode1.getText().toString().trim().isEmpty() // if the code is not enter, then an error message will appear
                || inputCode2.getText().toString().trim().isEmpty()  //after user click on the verify button
                || inputCode3.getText().toString().trim().isEmpty()
                || inputCode4.getText().toString().trim().isEmpty()
                || inputCode5.getText().toString().trim().isEmpty()
                || inputCode6.getText().toString().trim().isEmpty()) {
            Toast.makeText( context: VerifyOtp.this,  text: "Please enter valid code", Toast.LENGTH_SHORT).show();
            return;
        }
```

Figure 15: Code of Validating OTP Received

Aside from that, the main security features for this proposed application are a fingerprint biometric to authenticate and authorize user as well as preventing from unauthorized access and data leaked. Figure 16 shows the fingerprint interface.

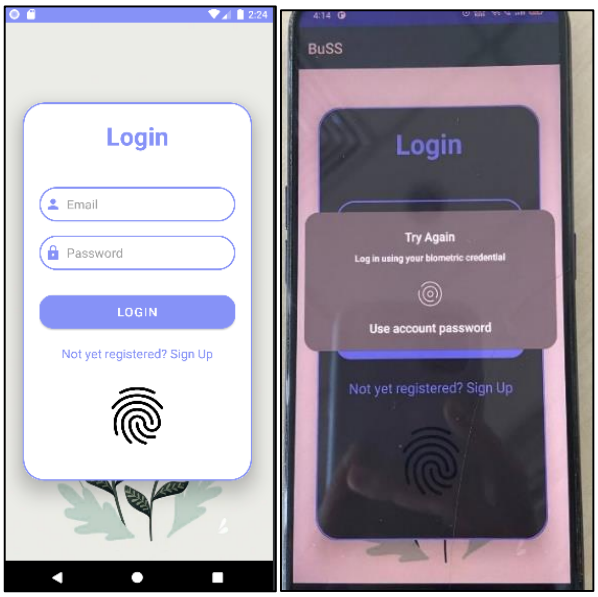

Figure 16: Interface of Fingerprint

Referring to Figure 16, when the bus driver clicks on the fingerprint symbol at the bottom of the login page, then the pop-up message that required them to use the fingerprint is appear. However, if they want to use email and password then they just need to input their correct username or password or else login failed. Figure 17 shows the addition of a Gradle dependency on the android.biometric library into the app module's build.gradle. The code proposes adding biometric authentication to the app.

```
androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
implementation 'androidx.biometric:biometric:1.1.0'
```

Figure 17: Code of Adding Gradle Dependency

After adding the dependency, the required library will be downloaded automatically. The code shown in Figure 18 is to get the user permission to use the biometric authentication and the permission is written in AndroidManifest.xml file.

```
<uses-permission android:name="android.permission.USE_BIOMETRIC" />
```

Figure 18: Code of Permission in AndroidManifest.xml File

Moreover, the application will check to see if the mobile device supports biometric authentication, and the code is shown in Figure 19. If the device does not support biometrics, an error message appears indicating the type of fault, such as the phone lacking a fingerprint sensor or the fingerprint sensor being unable to read the user's fingerprint.

```
BiometricManager biometricManager = BiometricManager.from(this);
switch (biometricManager.canAuthenticate( authenticators: BIOMETRIC_STRONG | DEVICE_CREDENTIAL)) {
    case BiometricManager.BIOMETRIC_SUCCESS:
        Log.d( tag: "MY_APP_TAG",  msg: "App can authenticate using biometrics.");
        break;
    case BiometricManager.BIOMETRIC_ERROR_NO_HARDWARE:
        Toast.makeText( context: this,  text: "Fingerprint Sensor Not Available", Toast.LENGTH_SHORT).show();
        break;
    case BiometricManager.BIOMETRIC_ERROR_HW_UNAVAILABLE:
        Toast.makeText( context: this,  text: "Sensor Not Available or Busy", Toast.LENGTH_SHORT).show();
        break;
    case BiometricManager.BIOMETRIC_ERROR_NONE_ENROLLED:

        final Intent enrollIntent = new Intent(Settings.ACTION_BIOMETRIC_ENROLL);
        enrollIntent.putExtra(Settings.EXTRA_BIOMETRIC_AUTHENTICATORS_ALLOWED,
                value: BIOMETRIC_STRONG | DEVICE_CREDENTIAL);
        startActivityForResult(enrollIntent, REQUEST_CODE);
        break;
}
```

Figure 19: Code Used to Determine Whether the Device Supports Biometric Authentication or Not

Furthermore, Figure 20 shows a part of the process of building an instance biometric prompt, which provides a standardized way to manage biometric authentication across multiple devices and biometric sensors, as well as the process of validating the biometric. The authentication() function is called on the biometricPrompt object to initiate the authentication process, which involves asking the user for their biometric data, which is their fingerprint. The biometric data is then compared to the already enrolled biometric data on the device. If the biometric data matches, the authentication is successful, and the message "Login Successful" is displayed.

```
Executor executor= ContextCompat.getMainExecutor( context: this);
biometricPrompt= new BiometricPrompt( activity: MainActivity.this, executor, new BiometricPrompt.AuthenticationCallback() {
    @Override
    public void onAuthenticationError(int errorCode, @NonNull CharSequence errString) {
        super.onAuthenticationError(errorCode, errString);
    }

    @Override
    public void onAuthenticationSucceeded(@NonNull BiometricPrompt.AuthenticationResult result) {
        super.onAuthenticationSucceeded(result);
        Toast.makeText(getApplicationContext(), text: "Login Successful", Toast.LENGTH_SHORT).show();
        mMainLayout.setVisibility(View.VISIBLE);
    }

    @Override
    public void onAuthenticationFailed() { super.onAuthenticationFailed(); }
});

promptInfo =new BiometricPrompt.PromptInfo.Builder().setTitle("Tech Projects")
        .setDescription("Use Fingerprint To Login").setDeviceCredentialAllowed(true).build();

biometricPrompt.authenticate(promptInfo);
```

Figure 20: Code of Creating an Instance Biometric Prompt and Validate Biometric

Figure 21 shows how to create fingerprint authentication by using an account password as another form of login which are the last part for implementation of fingerprint. The promptInfo object holds the biometric authentication prompt's configuration, including the title, subtitle, and description. The build() method completes the builder and creates an unchangeable promptInfo object that is ready to use.

```
promptInfo = new BiometricPrompt.PromptInfo.Builder()
        .setTitle("Biometric login for my app")
        .setSubtitle("Log in using your biometric credential")
        .setNegativeButtonText("Use account password")
        .build();
```

Figure 21: Code of Giving Another Option than Fingerprint

The next security element included in the proposed application is a strong password. A strong password is essential for a variety of reasons, including security against unauthorized access and defense against popular hacking techniques. In the beginning, the application will validate the user's password, and if the password does not meet the required parameters, an error message "Password must be strong" will be displayed. The coding for the password validation area is shown in Figure 22.

```
}else if (!passwordValidator.isValid(pass)) {
    signupPassword.setError("Password must be strong");
```

Figure 22: Code of Requiring User to Use Strong Password

The user's password should adhere to the strong password requirements, such as requiring a minimum of eight characters (8,20$), digits (?=.*[0-9]), lowercase letters (?=.*[a-z]), uppercase letters (?=.*[A-Z]), and special characters (?=.*[!@#$()-[]:;',?/*$+= >]). The matches() method, as shown in Figure 23, is used to determine whether the password matches the pattern. Return a boolean value confirming the validity of the password.

```
class passwordValidator {
    private static final String PASSWORD_PATTERN = "^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z])(?=.*[!@#&()–[{}]:;',?/*~$^+=<>]).{8,20}$";
    private static final Pattern pattern = Pattern.compile(PASSWORD_PATTERN);

    public static boolean isValid(final String password) {
        Matcher matcher = pattern.matcher(password);
        return matcher.matches();
    }
}
```

Figure 23: Implementation of Strong Passwords

Input validation is the last security features that applied in this proposed application. The input entered by the bus driver will be checked first before they entered the application. In this case, the email address will be checked. The email should meet the requirements that have been established which is the use of the '@' symbol.

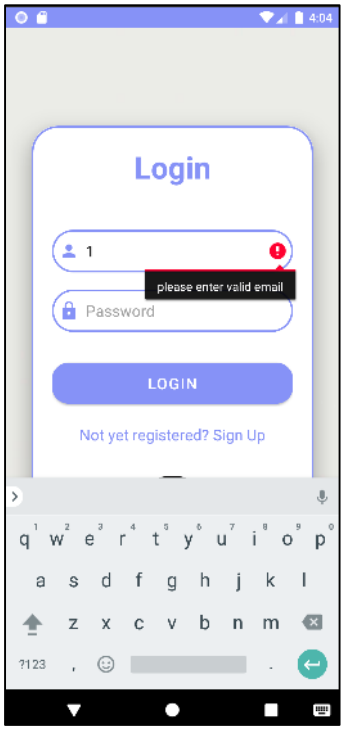

Figure 24: Pop-Up Message to Check the Requirement

Figure 24 shows the error message of input validation. The application checks the user's input in order to verify that only properly formatted data is entered. If the input does not fulfill the requirements, an error message will be displayed. This can assist prevent malformed data from being stored in the database.

5.2 Implementation of Application's Module

Figure 25 is the dashboard that will appear in the bus driver and student interface once the application is open.

Figure 25: Dashboard

Figure 26 shows the interface of the bus schedule. It contains two buttons. The first button is displaying the shuttle bus schedule from Perwira residential college to UTHM main campus. The second button displays the scheduled bus from Tun Fatimah and Tun Dr Ismail residential college to the city campus to UTHM main campus.

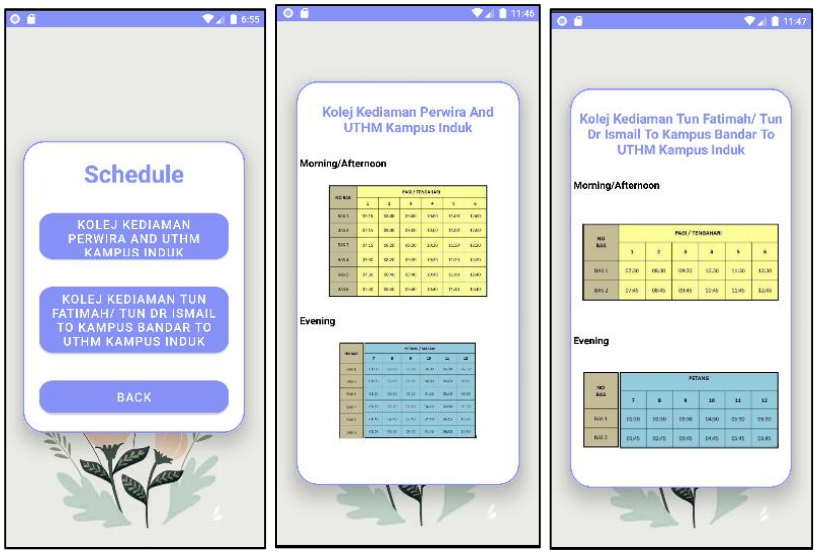

Figure 26: Bus Schedule Homepage

Furthermore, Figure 27 shows the interface of bus driver homepage and the interface of share location once the bus driver clicks on the share location button.

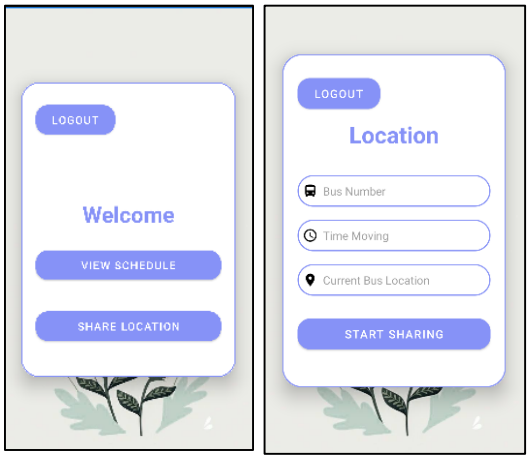

Figure 27: The Interface of Location

The bus driver must provide their bus number, departure time, and current bus position. The data is subsequently stored in Firebase and displayed on the students' location page.

6. Results and Discussions

This chapter discussed the outcome of the proposed application's security and module. This chapter also discusses the result of user approval.

6.1 Test Plan Result

Table 4 displays the results of the application functional testing. The proposed application passes all tests with the desired outcomes.

Table 4: Result of test Plan

Category	Expected result	Actual Result (Pass/Fail)
Sign up	Bus driver can easily sign up to the application	Pass
	The user should be able to create a strong password	Pass
	Bus driver get the OTP number	Pass
Login	Error messages appear such as “Incorrect password” or “incorrect username”	Pass
	The user should be able to input email and password	Pass
	The user should be able to login using the correct email and password	Pass
	The user should not be able to login with an invalid email and password	Pass
	The user should be able to login using a valid fingerprint	Pass
	The user should not be able to login using an invalid fingerprint	Pass
Bus schedule	User should be able to click on the button to view the bus schedule	Pass
	User should be able to view the bus schedule	Pass
Bus location	Bus driver can share their current location	Pass

6.2 User Acceptance Test

Because only bus drivers are required to login to the application, the user testing for the sign up and login module is distributed to three bus drivers.

Table 5: User Testing on Signup and Login Module

No	Question	Actual Result (Yes/No)
1	Bus driver can easily sign up to the application	Yes
2	The user should be able to create a strong password	Yes
3	Bus driver get the OTP number	Yes
4	Error messages appear such as “Incorrect password” or “incorrect username”	Yes
5	The user should be able to input email and password	Yes
6	The user should be able to login using the correct email and password	Yes
7	The user should not be able to login with an invalid email and password	Yes
8	The user should be able to login using a valid fingerprint	Yes
9	The user should not be able to login using an invalid fingerprint	Yes

The results of user testing on the login and signup module are shown in Table 5. All respondents can easily signup to the application and they are required to create a strong password also they successfully get the OTP numbers. Apart from that, the respondents are able to access the application and able to login to the application using the correct fingerprint and password. The user is not able to login using an invalid email or password. There is also an error message that appear for incorrect username, passwords, and the format of input validation. Furthermore, all respondents can login to the application by scanning their fingerprint.

In addition, user testing for the bus schedule and bus location module is shown in Table 6 and 7. The user testing is done by five UTHM students who using shuttle bus service and three bus drivers.

Table 6: User Testing on the Application’s Module – Students

No	Question	Actual Result (Yes/No)
1	User should be able to click on the button to view the bus schedule	Yes
2	User should be able to view the bus schedule	Yes

Table 6 shows user testing on the homepage and bus schedule. All respondents can access the home page and can view the bus schedule. That indicates that the module is functioning properly.

Table 7: User Testing on the Application's Module – Bus Drivers

No	Question	Actual Result (Yes/No)
1	User should be able to click on the button to view the bus schedule	Yes
2	User should be able to view the bus schedule	Yes
3	Bus driver can share their current location	Yes

The results of user testing for bus drivers on the homepage, bus schedule, and bus location are shown in Table 7. All responses can view the bus schedule and access the main page. This signifies that the module is operational.

7. Conclusion

A Bus Schedule Application using Fingerprint authentication allows students to easily access the bus tracking application since the old bus system was a website and is now a mobile application. Furthermore, the use of fingerprints does enable the bus driver to connect to the application more quickly and protects user confidentiality.

The proposed application 's limitations are that it is only available for Android users, and not every student are Android users. Thus, students who use a different operating system than Android are unable to download the app since it is not available in their play store, ignoring the fact that the app should be available for other mobile operating systems. Next, not all devices support fingerprint scanners, so they must login using username and password, which takes some time. Lastly, the bus location that shared by the bus driver does not appear in students' side.

In a nutshell, a Bus Schedule application with Fingerprint authentication is now fully working. The application is designed to address the issue of making it easier for students to find the bus timetable while also reducing the login time for bus drivers by allowing them to use their fingerprint as a login method and increasing security by providing strong passwords, input validation, and OTP. Even though, the application's objectives have been reached, the application still needs improvement in several areas in order to be more user-friendly. In the future, the application should be accessible to all users, since the proposed application only available for android operating system. Furthermore, it is preferable if the application can display the bus's estimated arrival time. For example, the application indicates that the bus will arrive at the next bus stop in one minute. Aside from that, the application should include extra security measures, such as a security log that can monitor the user's activity to enhance the security. Lastly, the application should display live bus movement using Google Maps, making it easier for students to track the movement of the bus rather than the bus driver having to manually put in their current location, which can take some time.

Acknowledgment

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

References

[1] "School Transportation Securing the Best Options." Accessed: Jan. 07, 2023. [Online]. Available: www.TRB.org

[2] "Transportation Policy," *Sunway University*. https://university.sunway.edu.my/sustainability/travel-and-transportation/transportation-policy (accessed Jan. 07, 2023).

[3] KATSANA.org, "KATSANA Shuttle," *University Tun Hussein Onn Malaysia*. https://uthm.katsana.com/ (accessed Jan. 07, 2023).

[4] F. Gazzawe, "Comparison of Websites and Mobile Applications for E-Learning," 2017.

[5] M. A. Hussin, M. F. A. Kadir, S. A. M. Ghazali, S. H. Md Hanafiah, and A. H. Zakaria, "The effectiveness of web systems and mobile applications for their end-users," *International Journal of Engineering Trends and Technology*, no. 1, pp. 148–152, Aug. 2020, doi: 10.14445/22315381/CATI3P224.

[6] A. A. Sheikh, P. T. Ganai, N. A. Malik, and K. A. Dar, "Smartphone: Android Vs IOS," *The SIJ Transactions on Computer Science Engineering & its Applications (CSEA)*, vol. 01, no. 04, pp. 31–38, Oct. 2013, doi: 10.9756/sijcsea/v1i4/0104600401.

[7] D. Bhattacharyya, R. Ranjan, F. Alisherov, C. Minkyu, A. A. Farkhod, and M. Choi, "A review on bio medical image processing View project Biometric Authentication: A Review," 2009. [Online]. Available: https://www.researchgate.net/publication/46189709

[8] "How Fingerprinting Works," *howstuffworks*. https://science.howstuffworks.com/fingerprinting.htm (accessed Jan. 07, 2023).

[9] M. Omar Rayes, *Encyclopedia of Cryptography and Security*, Second. Springer, Boston, MA, 2011.

[10] M. R. Saidin, "RFID Based Campus Bus System Using Internet Of Things (IOT)," 2017.

[11] D. Parkash Chechi, P. J. Kaur C Bose, D. Parkash, T. Kundu, and P. Kaur, "THE RFID TECHNOLOGY AND ITS APPLICATIONS: A REVIEW," 2017. [Online]. Available: https://www.researchgate.net/publication/232575248

[12] C. K. Lip, "IoT Based MRT Feeder Bus Tracking System," 2018.

[13] L. S. Ng, "Delivery Management System," Jul. 2020.

[14] Lumitex.org, "Prototyping Methodology Steps on How to Use It Correctly," 2017.