



**TK2023**

## **Object-Oriented Software Engineering**

### **UKM e-Kewangan**

#### **Group Members:**

1. Muhammad Haikal Iman Bin Osman (A179376)
2. Mohamad Saiful Nizam bin Abd Aziz (A179830)
3. Sofea Balqis binti Sri Emirlee (A180892)
4. Muhammad Nur Akmal bin Mohamad Razif (A181765)

# Table of Contents

## CHAPTER 1 INTRODUCTION

1.1	Introduction	3
1.2	Problem Statement	3
1.3	Proposed Solution	3
1.4	Objectives	4
1.5	Scope	4
1.6	Constraints	4
1.7	Methodology	5 - 6
1.8	Project Schedule	7

## CHAPTER 2 SOFTWARE REQUIREMENTS

2.1	Introduction	8
2.2	Functional Requirements	
2.2.1	User Requirements	8 - 9
2.2.2	System Requirements	9 - 10
2.3	Quality Requirements	10 - 11
2.4	Constraints	11
2.5	Domain Model	12
2.6	Use Case	
2.6.1	Use Case Diagram	13 - 14
2.6.2	Use Case Specifications	14 - 23

2.6.3	System Sequence Diagrams
-------	--------------------------

24 - 30
---------

## **CHAPTER 3 DESIGN AND IMPLEMENTATION**

3.1	Introduction	31
3.2	Class Diagram	31
3.3	Sample Output	32 - 38

# **CHAPTER 1**

## **1.1 INTRODUCTION**

Universiti Kebangsaan Malaysia created UKM e-Kewangan, a web-based system. The major role of e-Kewangan is to pay study fees, which makes fee management easier by allowing students to pay all fees via MOLPay's secure online payment system. Students/users can choose from a variety of options in MOLPay, including debit/credit card payments, as well as cash payments at 7Eleven Store and bank counters.

## **1.2 PROBLEM STATEMENT**

A study fee is mandatory for every student because the fee is used for study purposes. Before the final examination begins each semester, the student must pay all fees. If a student is unable to pay their fees, they may be banned from taking the final exam. Students had to go to the office to pay their study fees using the manual payment technique, which had been employed previously. This old technique is inefficient since students must bring a large sum of cash to the counter and queue in line with other students. Aside from that, students and university management have lost a lot of time, energy, and made a mess to record the system.

## **1.3 PROPOSED SOLUTION**

E-Kewangan was created to provide a more efficient method of recording and paying study fees. Students can pay all fees online via the e-Kewangan system, eliminating the need to queue at the counter. Students and university administration will save time and have a more accurate record of fees. Instead of paying study fees and other fees, the e-kewangan system has a lot of functions, but it also has the receipt and invoice as proof of payment. As a result, there are no difficulties such as cheating or misrepresenting information because everything in the system is secure.

## **1.4 OBJECTIVES**

The primary goal of the research is to identify the flaws in the UKM e-Kewangan system and to upgrade it to a better version. The specific objectives include:

1. To develop a more efficient payment method for all fees at UKM.
2. To make sure all the records are secured in the system.

## **1.5 SCOPE**

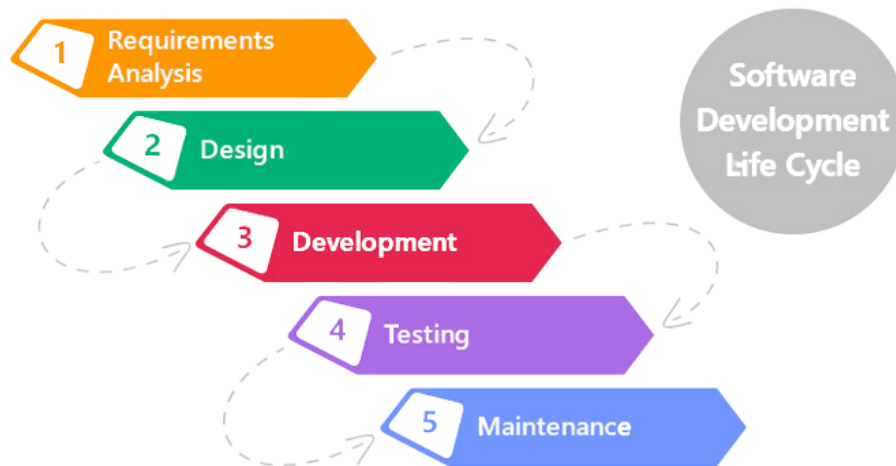
We choose Java Eclipse to code the application for this project. We will improve the present website's UI design and add a few features.

## **1.6 CONSTRAINTS**

The team members are unable to meet in person, which is a constraint in the development of this programme. As a result, holding a group discussion regarding this project is challenging, and it can only be done through online platforms such as Microsoft Team and WhatsApp. It may delay down the development process, and it is possible that e-Kewangan will not be ready for usage by university students and professors in six months.

Aside from that, this app should be written in the Java programming language. This is one of the application's constraints because using multiple types of programming languages can be confusing for the developer and will almost certainly become a major issue in the future.

## 1.7 METHODOLOGY



***Diagram 1.0 Methodology for e-Kewangan UKM application***

After considering all aspects of the program's development, the waterfall model was chosen as the software development life cycle for the e-Kewangan UKM application. This is because the waterfall technique is a linear project management strategy in which stakeholder and user needs are gathered at the start of the project and then a sequential project plan is developed to suit those requirements. The waterfall model gets its name from the fact that each phase of the project flows into the next, descending steadily like a waterfall.

Diagram 1.0 shows a methodology diagram for e-Kewangan UKM application. The first step in waterfall development is requirements analysis which is defined by the fact that all user requirements are gathered at the beginning of the project, allowing for the planning of subsequent phases without additional user communication until the product/system is finished. During the waterfall management phase, all requirements should be collected. Next, the waterfall process' design phase can be divided into two parts: logical design and physical design. Possible solutions are explored and speculated during the logical design subphase. The physical design process transforms theoretical concepts and schemas into concrete specifications. In the implementation phase, programmers use the requirements and specifications from the previous phases to create actual code. For the testing phase, the user

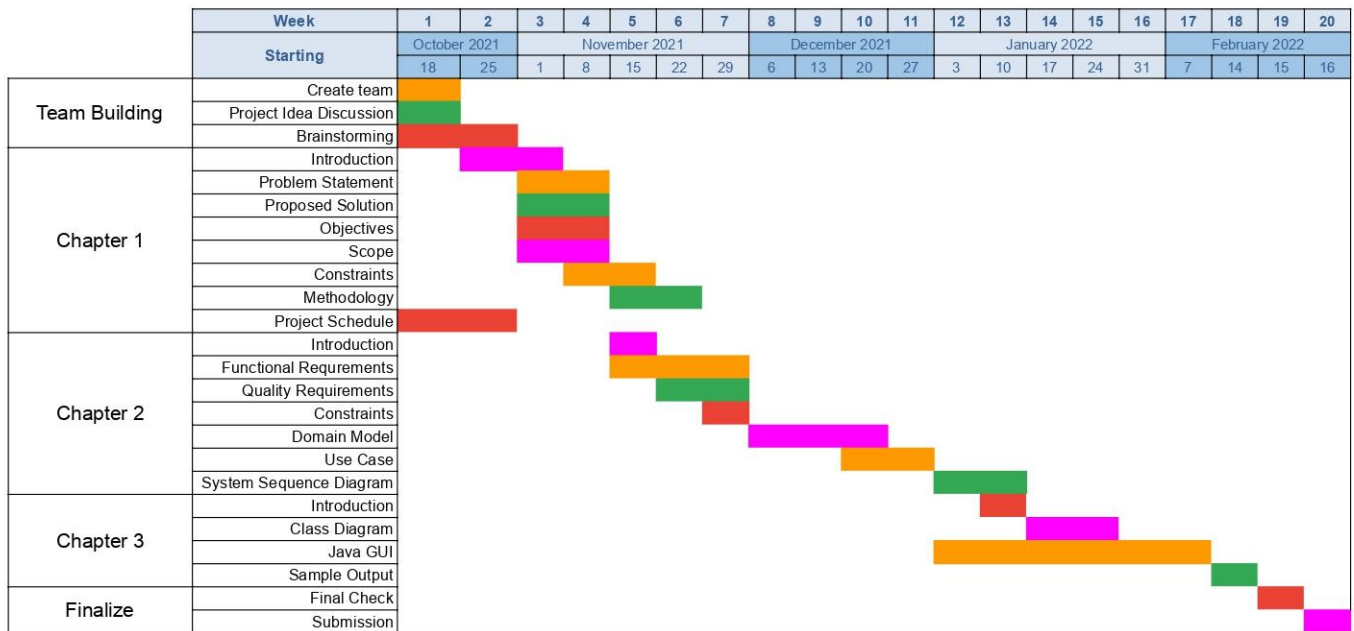
reviews the product/system in this phase to ensure that it fits the requirements established at the start of the waterfall project. This is accomplished by handing over the finished product to the user. The last phase in the waterfall model is maintenance which requires the user to frequently utilize the product/system, detecting defects, insufficient features, and other mistakes that happened. These fixes are applied as needed by the production team until the user is satisfied with the product/system.

Finally, we can say that the waterfall model is simple and straightforward. As a result, it will make a substantial contribution to the app's development while also saving time on the work. Furthermore, because each step is completed before moving on to the next, the waterfall approach is simple to maintain. Finally, the eventual product of application development will be more efficient. We are confident that using the waterfall technique will allow us to develop an application with different phases based on well-planned, structured, and organised stages.

## 1.8 PROJECT SCHEDULE

**UKM e-Kewangan Project Task Distribution**

TTTK2023 Object-Oriented Software Engineering



**Diagram 2.0 Project Schedule**



## CHAPTER 2

### REQUIREMENTS SPECIFICATION

#### 2.1 INTRODUCTION

A requirement specification is a list of all the requirements that must be imposed on the product's design and verification. Other information required for the product's design, verification, and maintenance is also included in the specification. However for our project, we will only be discussing further on the functional requirements, quality requirements, constraints, domain model, and use case.

#### 2.2 FUNCTIONAL REQUIREMENTS

Functional requirements specify what the system should accomplish or what services it should deliver. It contains everything a system user would need to know about the system's capabilities. The functional requirements could be categorized into two parts - user functional requirements (user requirements) and system functional requirements (system requirements).

User requirements are statements in natural language about the services provided by the system whereas system requirements are structured documents that describe the system services in detail.

##### 2.2.1 USER REQUIREMENTS

The user requirements of our system include *Login, Home, Profile, Finance Transaction, Payment Without Invoice, Invoice, Receipt, Payment Voucher, Advance Payment, Excess Payment, Logout, and Language*.

##### **Login**

**UFR1.0** The system shall be able to allow the users to login into their account.

### **Home**

**UFR2.0** The system shall be able to allow the users to access the application's home page.

### **Finance Transaction**

**UFR3.0** The system shall be able to allow the users to access the Finance Transaction section on the application's home page.

### **Invoice**

**UFR4.0** The system shall be able to allow the users to access the Invoice section on the application's home page.

### **Receipt**

**UFR5.0** The system shall be able to allow the users to access the Receipt section on the application's home page.

### **Payment Voucher**

**UFR6.0** The system shall be able to allow the users to access the Payment Voucher section on the application's home page.

### **Logout**

**UFR7.0** The system shall be able to allow the users to logout of their account.

## **2.2.2 SYSTEM REQUIREMENTS**

The system requirements of our system include *Login, Home, Finance Transaction, Invoice, Receipt, Payment Voucher, and Logout.*

### **Login**

**SFR1.1** The system shall provide the users with two textboxes to input the user ID and password.

**SFR1.2** The system shall provide the users with a login button to allow the system to validate the user ID and password.

**SFR1.3** The system shall provide the users with a 'Forgot-Password' button to assist them whenever their password is forgotten.

### **Home**

**SFR2.1** The system shall provide the users with a view for the application's home page.

### **Finance Transaction**

**SFR3.1** The system shall provide the users with buttons on Invoice, Receipt, and Payment Voucher, on the Finance Transaction section which will redirect users to the respective services.

### **Invoice**

**SFR4.1** The system shall provide the users with lists of invoices for both paid and unpaid fees.

**SFR4.2** The system shall provide the users to pick and pay on one or many invoices.

### **Receipt**

**SFR5.1** The system shall provide the users with lists of receipts for paid fees.

### **Payment Voucher**

**SFR6.1** The system shall provide the users with lists of payment vouchers received.

### **Logout**

**SFR7.1** The system shall provide the users with a logout button in order to terminate their active session.

## **2.3 QUALITY REQUIREMENTS**

Quality requirements are non-functional requirements created to ensure the usability, efficiency, reliability, maintainability, and reusability of a system. This section will emphasize more on the *Usability*, *Reliability*, *Efficiency*, and *Availability* of our UKM e-Kewangan system.

**Usability**

**QFR1.0** A trained user shall be able to fulfil a request in 4 to 6 minutes on average, while an untrained user should take no more than 20 minutes.

**Reliability**

**QFR2.0** The acceptable down time shall be 2 hours.

**Efficiency**

**QFR3.0** Responses to queries shall take no longer than 7 seconds to load onto the screen after the user submits the query.

**Availability**

**QFR4.0** The system shall be at least 99.9% available between 6am and midnight, and 95% between midnight and 5am local time.

## **2.4 CONSTRAINTS**

Constraints are non-functional requirements that restrict the way that the system shall be developed. In this part, we will be discussing the *Organisational/Project*, *Legal*, and *Culture*.

**Organisational/Project**

**C1.0** The effort for system development shall not exceed 5000 persons per month.

**Legal**

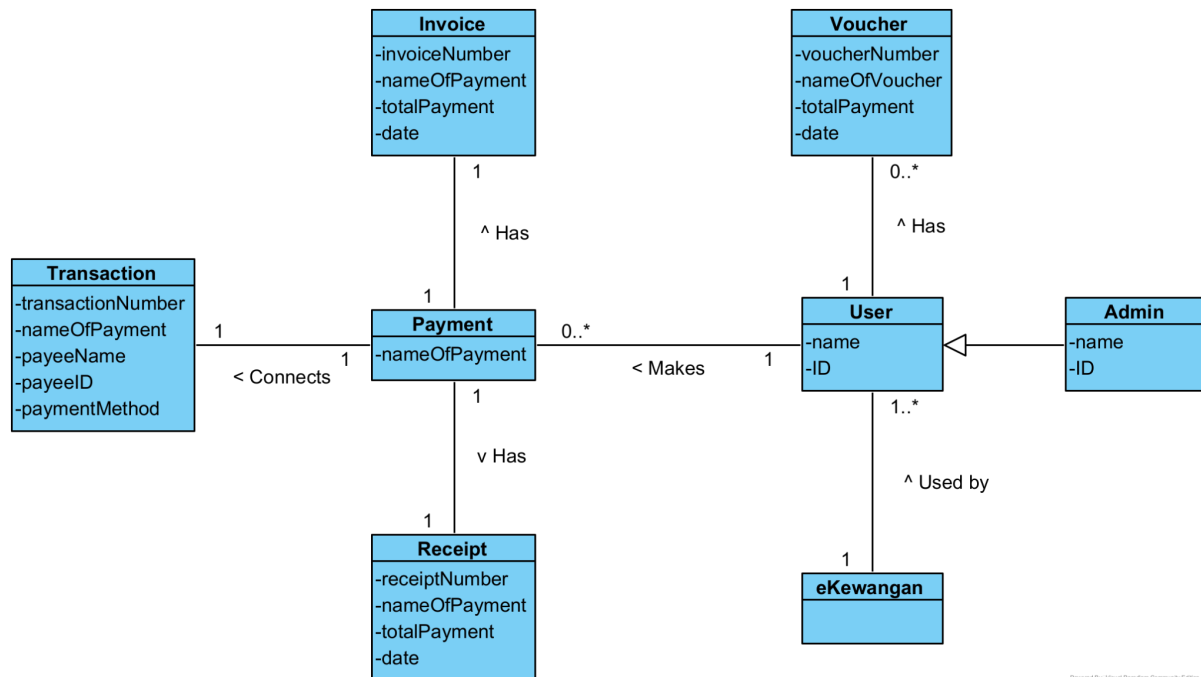
**C2.0** The system shall process personal data in compliance with Malaysian Data Protection Law.

**Culture**

**C3.0** The user interface shall not contain symbols or graphics that are abusive for any Malaysian.

## 1.5 DOMAIN MODEL

A domain model is a visual representation of conceptual classes or real-world objects in a domain of interest. The purpose of creating a domain model is to understand the domain of interest.



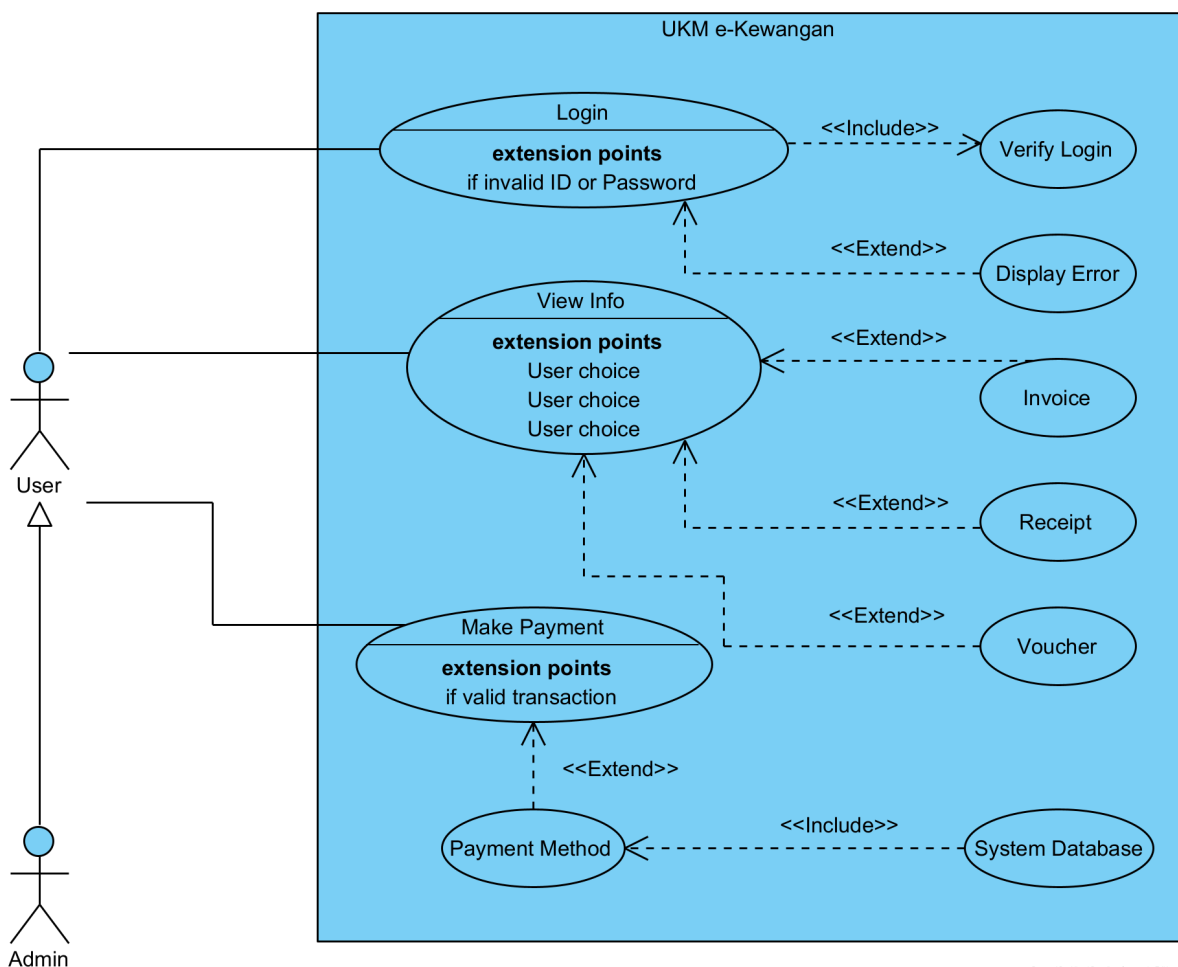
**Diagram 3.0 The domain model of the UKM e-Kewangan system.**

The diagram above shows the domain model of the UKM e-Kewangan system in a simplified manner as compared to the original version of the system available online. There are six entities which are eKewangan, User, Admin, Voucher, Payment, Receipt, Invoice, and Transaction. UKM e-Kewangan (*eKewangan*) is used by one-to-many users (*User*). Admin (*Admin*) is a special kind of user (*User*). Each user (*User*) has zero-or-many vouchers (*Voucher*). User (*User*) can make zero-or-many payments (*Payment*). Each payment (*Payment*) has a receipt (*Receipt*), and invoice (*Invoice*). Upon completing a payment (*Payment*), it will redirect user (*User*) to a transaction page (*Transaction*) which will provide them with lots of options on the payment method (*paymentMethod*) such as online banking or via debit/credit card.

## 1.6 USE CASE

Use Case analysis is a technique used to identify the functional requirements of a system. It is a graphical representation of tasks that will be performed by the actors. In this section, we will be discussing more on the use case diagram, use case specifications, and system sequence diagrams of our UKM e-Kewangan system.

### 2.6.1 USE CASE DIAGRAM



**Diagram 4.0 The Use Case diagram of the UKM e-Kewangan system.**

Based on the diagram above, the UKM e-Kewangan system has ten Use Cases which are *Login*, *Verify Login*, *Display Error*, *View Info*, *Invoice*, *Receipt*, *Voucher*, *Make Payment*, *Payment Method*, and *System Database*. *Login* and *Verify Login* have an *include* association because *Verify Login* only happens to exist if and only if the *Login* is being executed by the

actors and the same goes to the association between *Payment Method* and *System Database*. For all other use cases with *extended* association, they will be executed according to certain conditions. For instance, *Invoice*, *Receipt*, and *Voucher* info will be displayed based on the *User choice*.

## 2.6.2 USE CASE SPECIFICATIONS

ID:	UC01	
Title:	Login	
Description:	Allows users to log into the system.	
Primary Actor:	User	
Precondition:	Users must have a matric number as username and password to log into the system.	
Postcondition:	There must be no errors occurring after the login session.	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. Users must insert their own username and password to log into the system.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system requests the user's username and password.</li> <li>2. The system received the username and password entered by user</li> <li>3. The system verifies the username and password</li> <li>4. The system allows users to access the system.</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. User insert invalid username and password</li> <li>2. User forgot their password</li> </ol>	<ol style="list-style-type: none"> <li>1. The system received invalid username and password.</li> </ol>

**Table 1.0 The use case specification of UC01 Login.**

ID:	UC02	
Title:	Verify Login	
Description:	Allow the system to verify users' username and password	
Primary Actor:	User	
Precondition:	The system request user's username and password	
Postcondition:	The system allow user to access the system	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. Users must insert their own username and password to log into the system.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system requests the user's username and password.</li> <li>2. The system received the username and password entered by user</li> <li>3. The system verifies the username and password.</li> <li>4. The system allows users to access the system..</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. User inserts invalid username and password.</li> <li>2. User forgets their password.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system received invalid username and password.</li> </ol>

***Table 2.0 The use case specification of UC02 User.***



ID:	UC03	
Title:	Display Error	
Description:	Allow the system to display an error message to the user if an invalid username or password occurred.	
Primary Actor:	User	
Precondition:	The user entered a username and password.	
Postcondition:	The system will display an error message if an error occurred.	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. Users must insert their own username and password to log into the system.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system requests the user's username and password.</li> <li>2. The system received the username and password entered by the user.</li> <li>3. The system detects invalid login information.</li> <li>4. The system displays an error message to the user.</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. User inserts invalid username and password.</li> <li>2. User forgets their password.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system received invalid username and password.</li> </ol>

***Table 3.0 The use case specification of UC03 Display Error.***

ID:	UC04	
Title:	View Info	
Description:	Allow users to access profile and other main features of the system such as payment, invoice, receipt and voucher.	
Primary Actor:	User	
Precondition:	Users must be given access by the system through a login session.	
Postcondition:	Users can view the payment history and their own profile.	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. User chooses either invoice, receipt or voucher option.</li> <li>2. User views either list of invoice, receipt or voucher.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system gives invoice, receipt or voucher options for users to choose.</li> <li>2. The system displays either invoice, receipt or voucher selected by the user.</li> </ol>
Alternative Scenarios:	Actor	System

***Table 4.0 The use case specification of UC04 View Info.***

ID:	UC05	
Title:	Invoice	
Description:	Allows user to view payment invoice	
Primary Actor:	User	
Precondition:	User chooses the payment invoice option.	
Postcondition:	User views a list of payment invoices.	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. User chooses the invoice option.</li> <li>2. User views a list of invoices.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system gives invoice, receipt or voucher options for users to choose.</li> <li>2. The system displays a list of invoices to the user.</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. User chooses the receipt or voucher option.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system displays either a list of receipts or vouchers to the user.</li> </ol>

***Table 5.0 The use case specification of UC05 Invoice.***

ID:	UC06	
Title:	Receipt	
Description:	Allow users to view payment receipts.	
Primary Actor:	User	
Precondition:	User chooses payment receipt option	
Postcondition:	User views a list of payment receipt	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. User chooses the receipt option.</li> <li>2. User views a list of receipts.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system gives invoice, receipt or voucher options for users to choose.</li> <li>2. The system displays a list of receipts to the user.</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. User chooses the invoice or voucher option.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system displays either a list of invoices or vouchers to the user.</li> </ol>

***Table 6.0 The use case specification of UC06 Receipt.***

ID:	UC07	
Title:	Voucher	
Description:	Allow users to view payment vouchers.	
Primary Actor:	User	
Precondition:	User chooses payment voucher option	
Postcondition:	User views a list of payment voucher	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. User chooses the voucher option.</li> <li>2. User views a list of vouchers.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system gives either invoice, receipt or voucher options for users to choose.</li> <li>2. The system displays a list of vouchers to the user.</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. User chooses the invoice or receipt option.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system displays either a list of invoices or receipts to the user.</li> </ol>

***Table 7.0 The use case specification of UC07 Voucher.***

ID:	UC08	
Title:	Make Payment	
Description:	Allow users to make payments such as study fees, fines and others.	
Primary Actor:	User	
Precondition:	User's bank account is registered in the UKM database system.	
Postcondition:	The user successfully made the selected payment.	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. The user chooses which payment they want to pay.</li> <li>2. The user views the amount needed to be paid.</li> <li>3. The user enters the payment amount.</li> <li>4. The user views the invoice in the invoice section.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system shows the list of outstanding payments.</li> <li>2. The system displays the amount needed to be paid.</li> <li>3. The system asks the user to enter the amount they want to pay.</li> <li>4. The system accepts the payment details and payment amount made by the user.</li> <li>5. Once the payment is successful, the system records the payment information into the invoice.</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. The user enters invalid payment information.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system receives invalid payment information.</li> </ol>

***Table 8.0 The use case specification of UC08 Make Payment.***

ID:	UC09	
Title:	Payment Method	
Description:	Allow users to choose bank type such as Bank Islam, CIMB and other bank to make payment.	
Primary Actor:	User	
Precondition:	User's account bank is registered in the UKM system.	
Postcondition:	User views the successful payment that has been made.	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. The user chooses the payment method requested by the system.</li> <li>2. The user makes payment based on the selected payment method.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system asks the user to choose a payment method in terms of bank name.</li> <li>2. The system accepts the payment method chosen by the user.</li> <li>3. The system received payment details made by the user.</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. The user's account bank is not registered in the UKM system.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system is unable to identify the payment method selected by the user.</li> </ol>

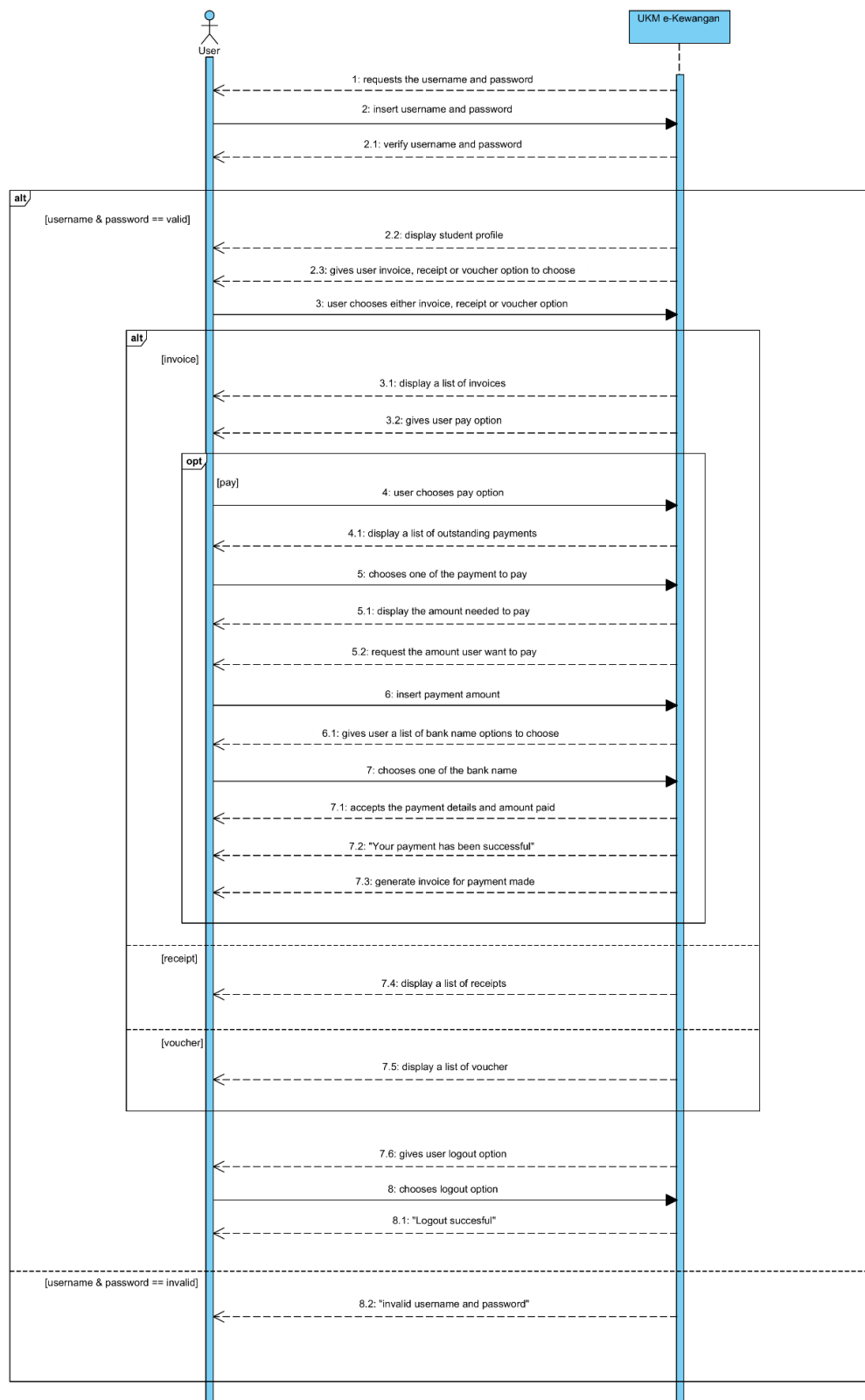
***Table 9.0 The use case specification of UC09 Payment Method.***

ID:	UC10	
Title:	System Database	
Description:	<ol style="list-style-type: none"> <li>1. Allow the system to record all user details and the transactions made by the user.</li> <li>2. Allow admin to access all records in the system database.</li> </ol>	
Primary Actor:	Admin	
Precondition:	All details related to UKM are recorded in the database system.	
Postcondition:	Admin can view and manage all data recorded in the database system.	
Main Success Scenario:	Actor	System
	<ol style="list-style-type: none"> <li>1. Admin log in into the system using their own username and password.</li> <li>2. Admin have permission to access and manage data in the database system.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system recorded all transactions made by users.</li> <li>2. The system validates the login information entered by the admin.</li> <li>3. The system gives permission to the admin to access and manage all data in the system.</li> </ol>
Alternative Scenarios:	Actor	System
	<ol style="list-style-type: none"> <li>1. Admin enters invalid username and password.</li> </ol>	<ol style="list-style-type: none"> <li>1. The system receives invalid username and password.</li> </ol>

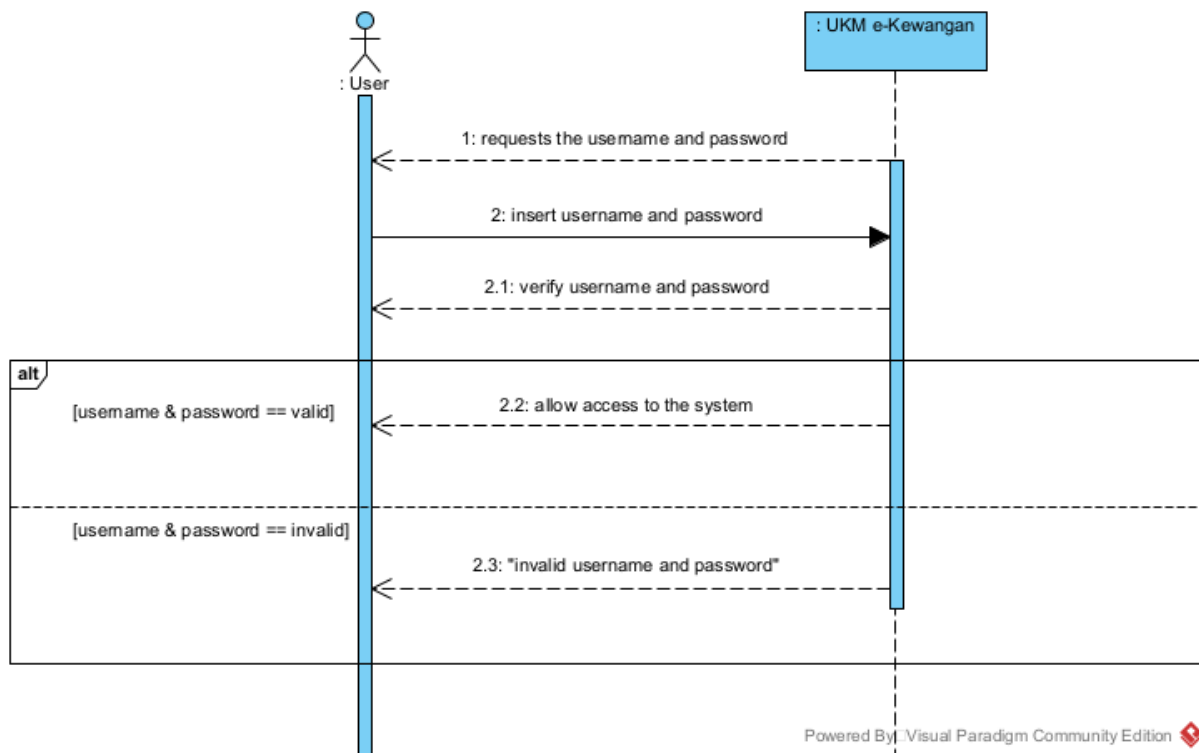
***Table 10.0 The use case specification of UC10 System Database.***



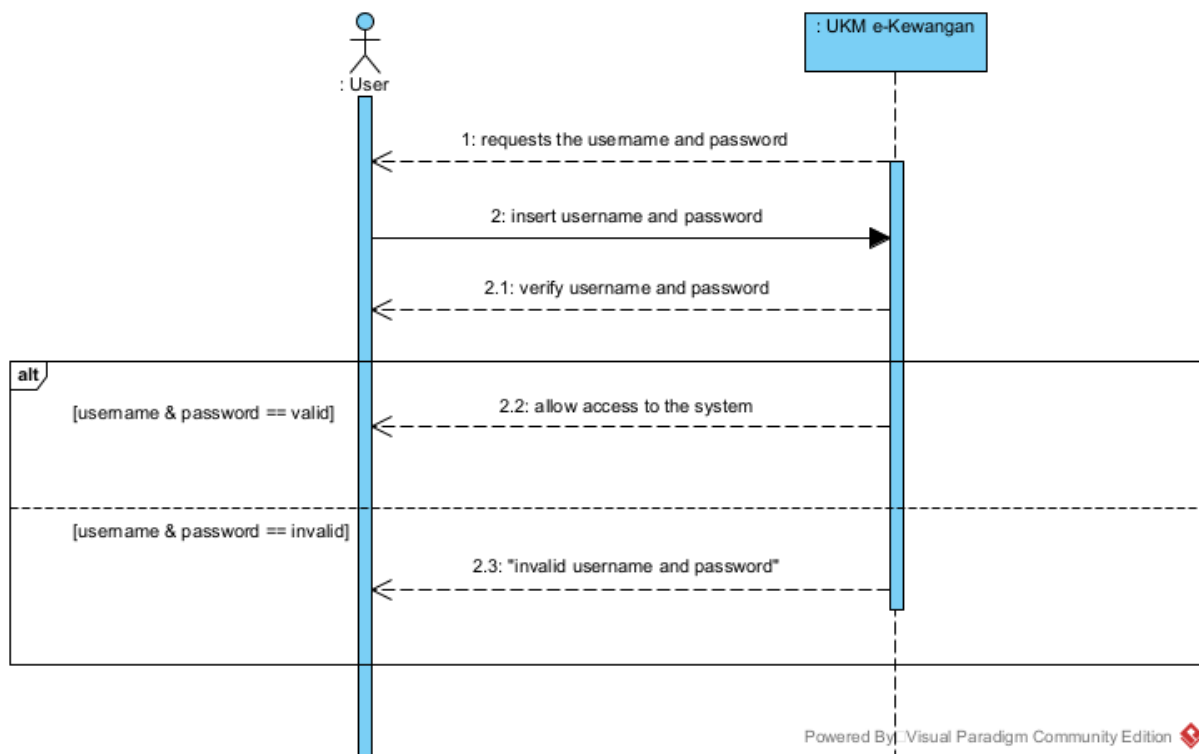
## 2.6.3 SYSTEM SEQUENCE DIAGRAMS



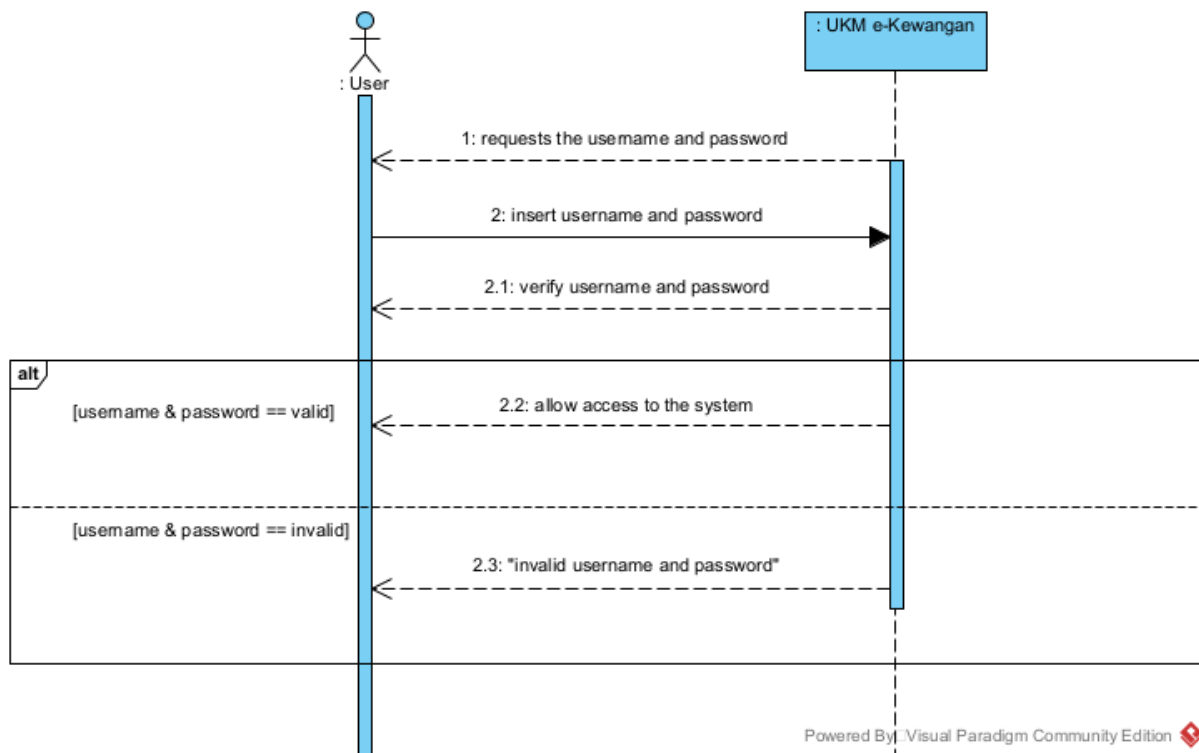
**Diagram 5.0** The Overall System Sequence diagram of the UKM e-Kewangan system.



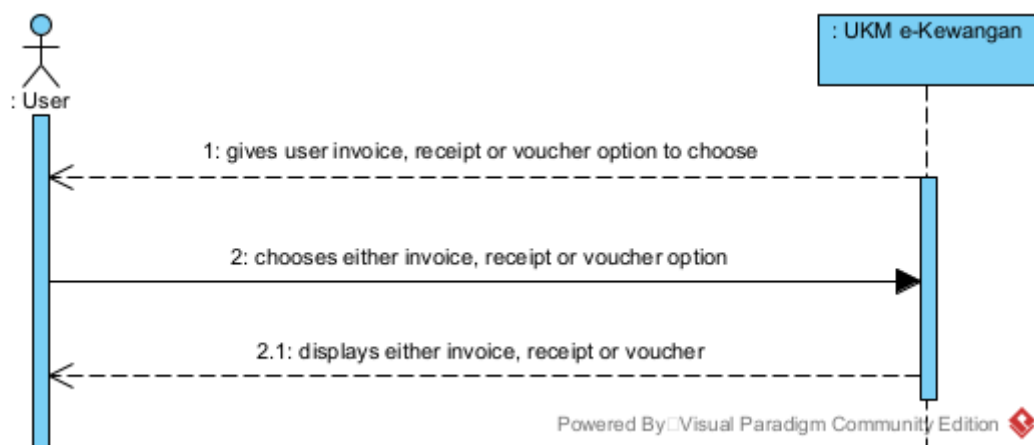
**Diagram 6.0** The System Sequence diagram for UC01 Login.



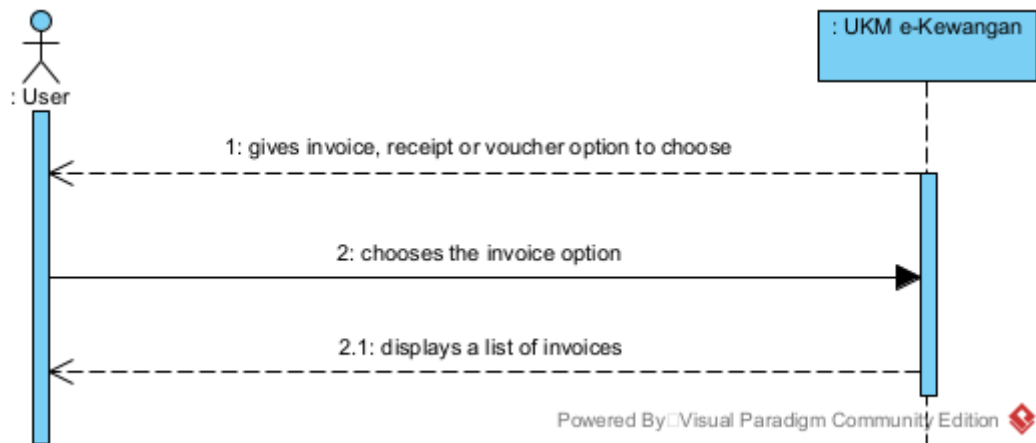
**Diagram 7.0** The System Sequence diagram for UC02 Verify Login.



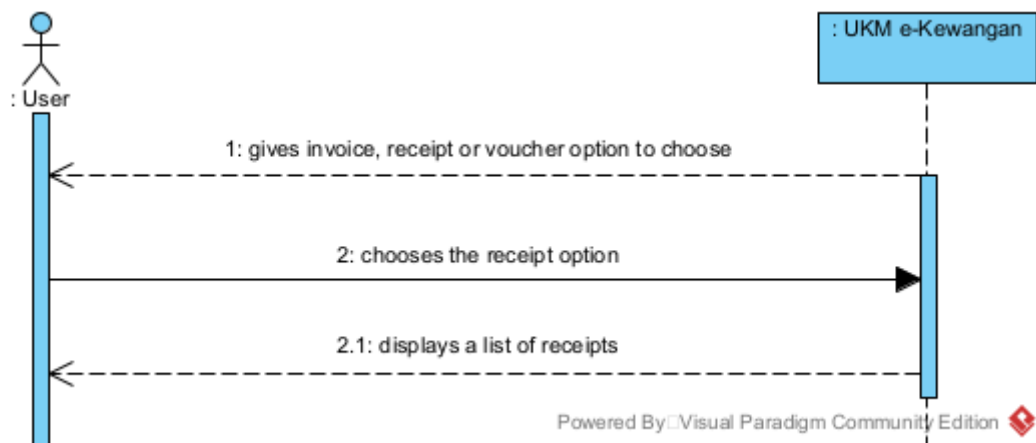
**Diagram 8.0 The System Sequence diagram for UC03 Display Error.**



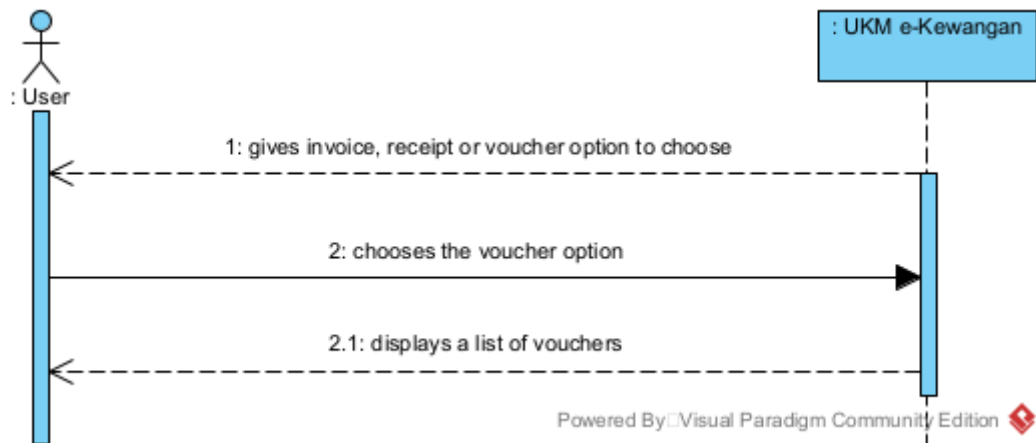
**Diagram 9.0 The System Sequence diagram for UC04 View Info.**



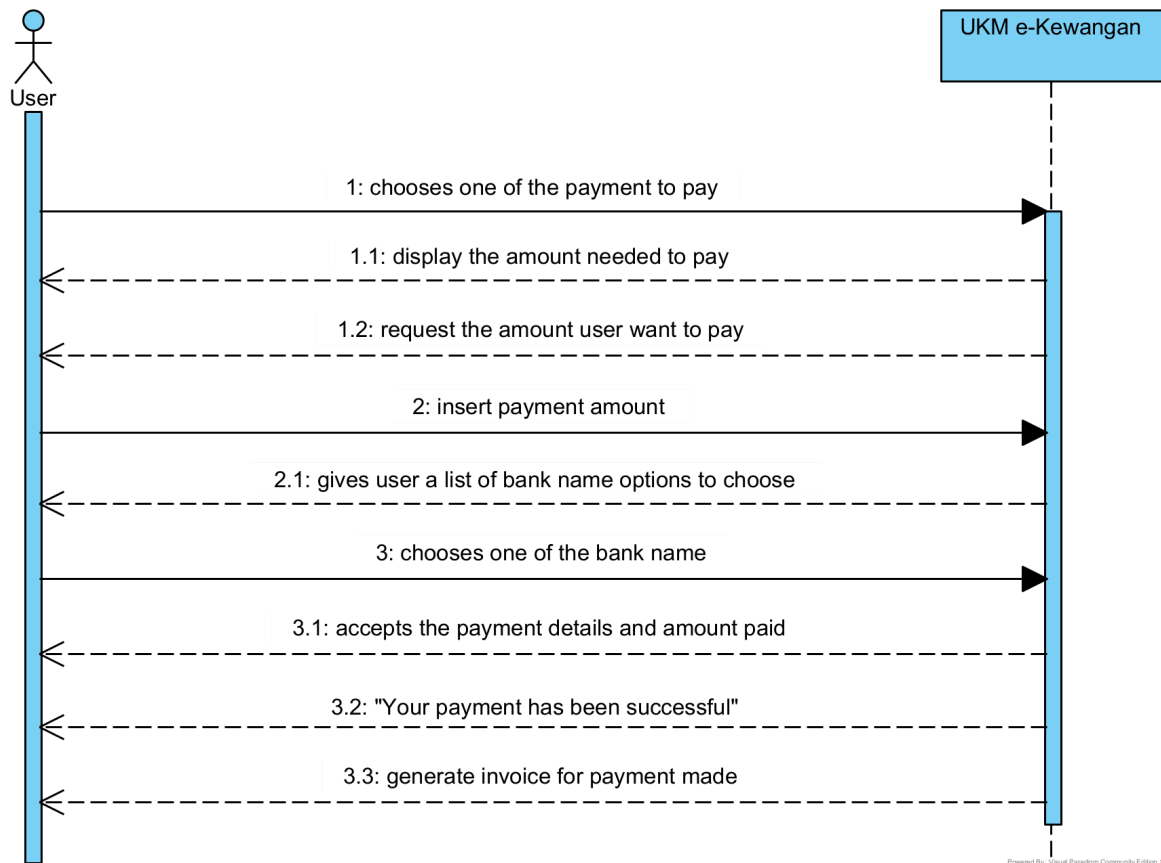
***Diagram 10.0 The System Sequence diagram for UC05 Invoice.***



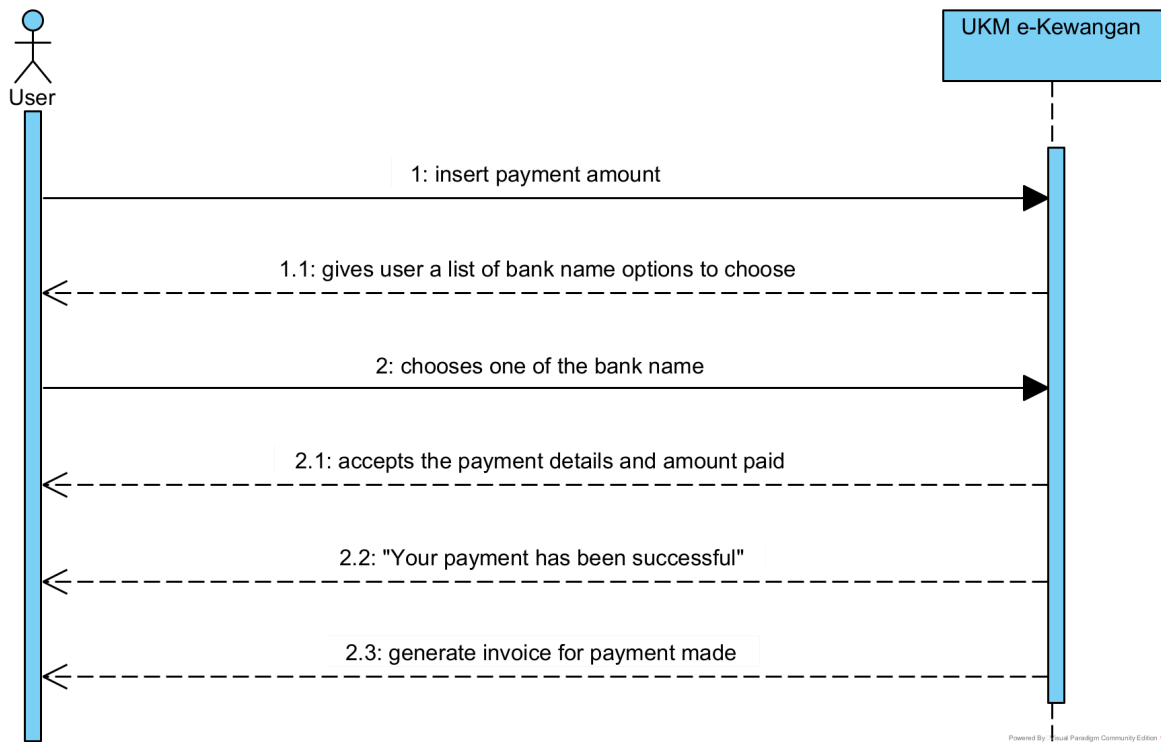
***Diagram 11.0 The System Sequence diagram for UC06 Receipt.***



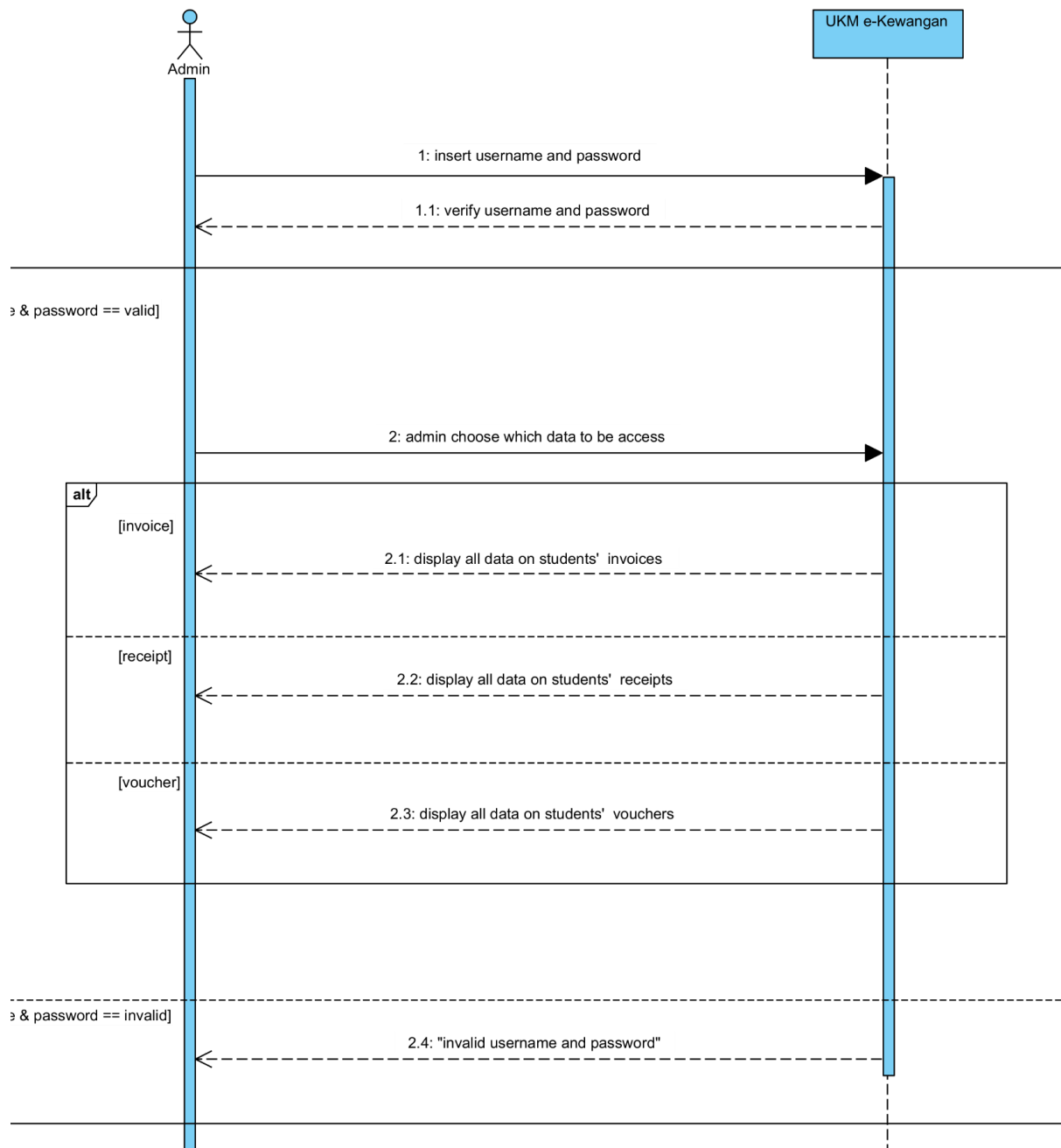
**Diagram 12.0 The System Sequence diagram for UC07 Voucher.**



**Diagram 13.0 The System Sequence diagram for UC08 Make Payment.**



***Diagram 14.0 The System Sequence diagram for UC09 Payment Method.***



**Diagram 15.0 The System Sequence diagram for UC10 System Database.**

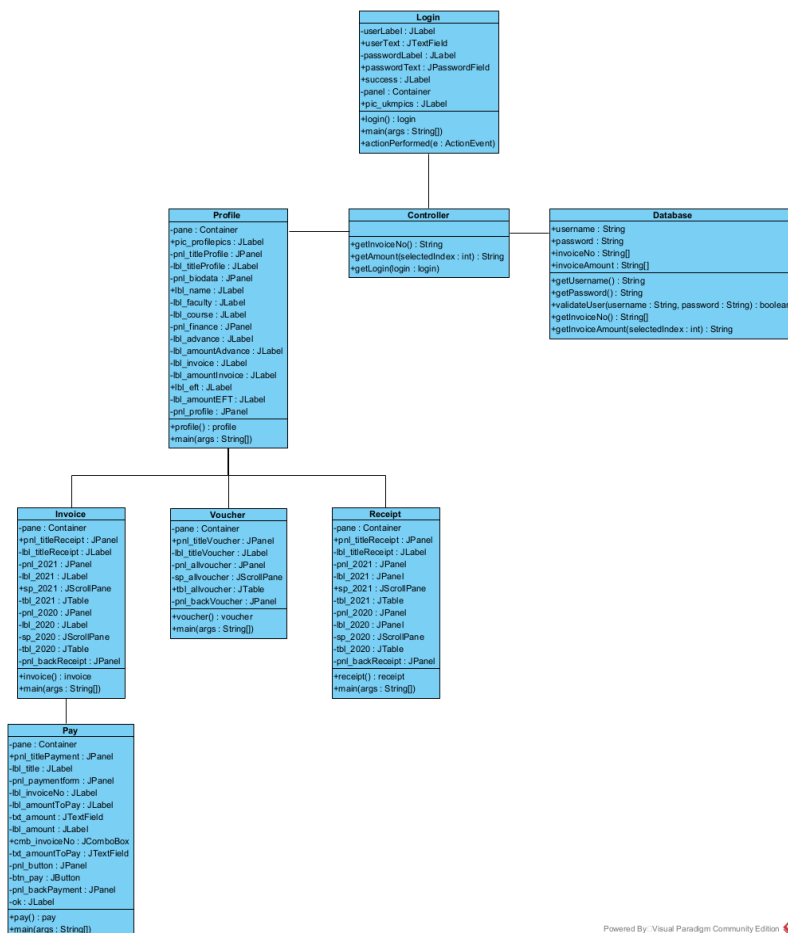
## CHAPTER 3

### DESIGN AND IMPLEMENTATION

#### 3.1 INTRODUCTION

The *Design* and *Implementation* of our UKM e-Kewangan system will be the subject of this chapter. The *Design* phase is when developers begin high-level and low-level software and system design in order to meet each requirement. The *Implementation* phase, on the other hand, is when the developers begin coding in accordance with the requirements and design specified during the preceding phase (*Software Requirement* and *Design*).

#### 3.2 CLASS DIAGRAM

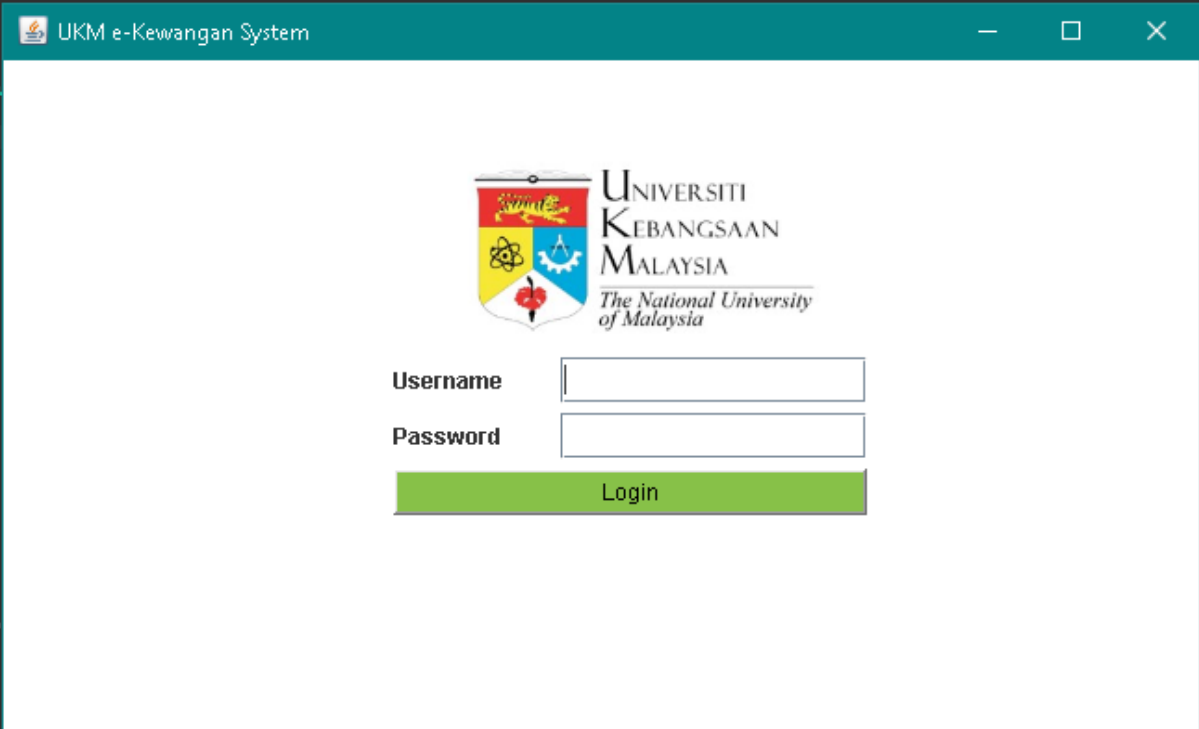


Powered By: Visual Paradigm Community Edition


Diagram 16.0 The Class diagram of the UKM e-Kewangan system.



### 3.3 SAMPLE OUTPUT



UKM e-Kewangan System

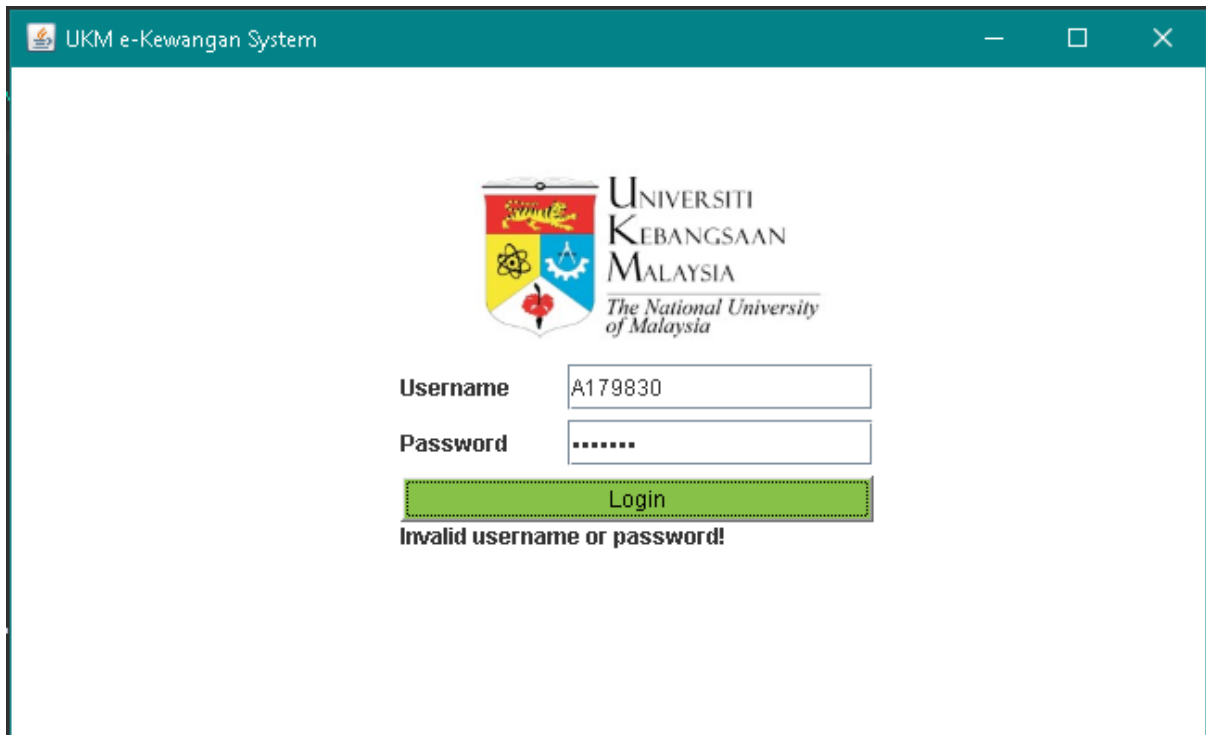
 UNIVERSITI  
KEBANGSAAN  
MALAYSIA  
*The National University  
of Malaysia*

Username

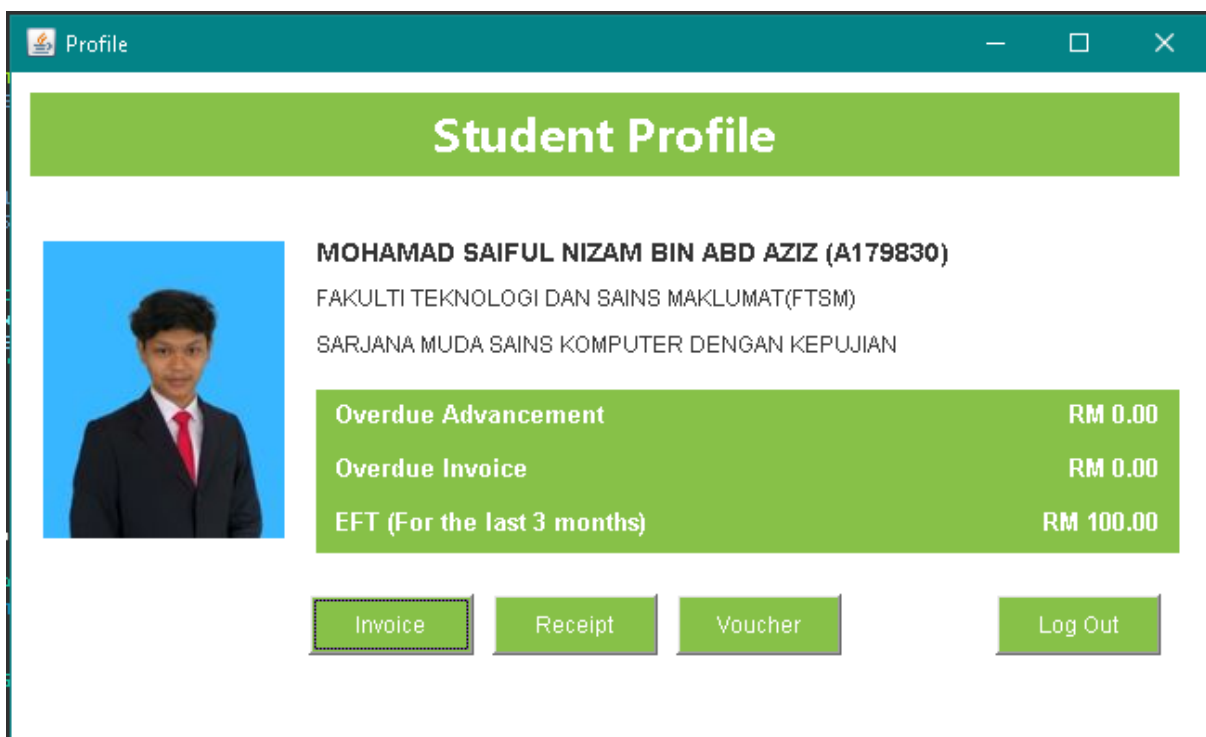
Password

Login

*Figure 1.0 Login page of the UKM e-Kewangan system.*



*Figure 2.0 An error message is displayed when the username or password is invalid.*



*Figure 3.0 Profile page of the UKM e-Kewangan system.*

**Invoice**

**2021**

No	Date	Invoice No.	Payment Information	Amount (RM)
1	17/10/2021	INV 1284480	Pembayaran hutang melalui Epayment : Bayaran e...	752.00
2	12/04/2021	INV 1198025	Pembayaran hutang melalui Epayment : Bayaran e...	831.00

**2020**

No	Date	Invoice No.	Payment Information	Amount (RM)
1	11/11/2020	INV 1128142	BAYARAN OLEH PENAJA LP15389 - (PTPTN) SP 4...	940.00
2	01/09/2020	INV 1098467	Pembayaran hutang melalui Epayment : [infoline] b...	1,200.00

**Pay** **Back**

*Figure 4.0 Invoice page of the UKM e-Kewangan system.*

**Pay**

Invoice No.

Amount Needed to Pay

Amount to Pay(RM) \*above RM5 only

**Pay** **Back**

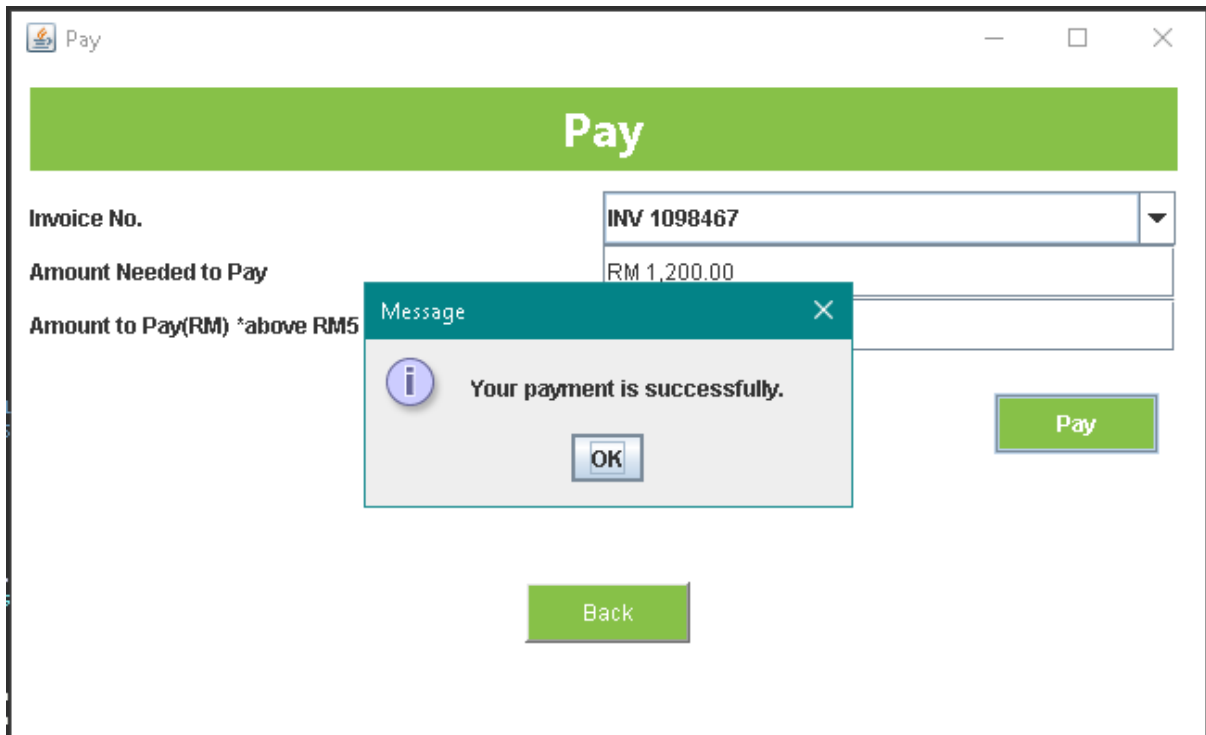
*Figure 5.0 Pay page of the UKM e-Kewangan system.*

The screenshot shows a web application window titled "Pay". It features a green header bar with the word "Pay" in white. Below the header, there are three input fields on the left and their corresponding values on the right: "Invoice No." with the value "INV 1098467", "Amount Needed to Pay" with the value "RM 1,200.00", and "Amount to Pay(RM) \*above RM5 only" with the value "1200.00". To the right of these fields is a green "Pay" button. At the bottom center, there is a green "Back" button.

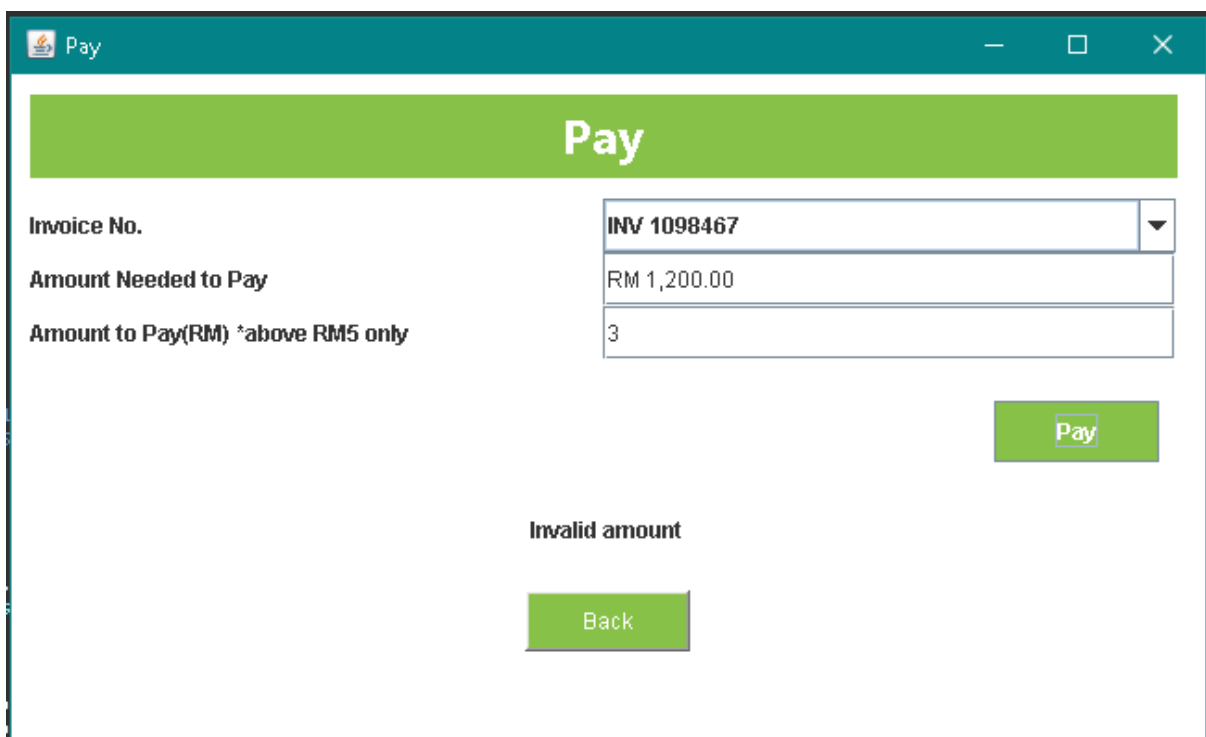
*Figure 6.0 The user selects the invoice number and enters the amount to be paid.*

This screenshot shows the same "Pay" application window as Figure 6.0, but with a confirmation dialog box overlaid in the center. The dialog box has a green header bar with a question mark icon and the text "Confirm Payment". Below this, it asks "Are you sure to continue this transaction?" and provides two buttons: "Yes" and "No". The background application window is partially obscured by the dialog box, but the "Pay" and "Back" buttons are still visible.

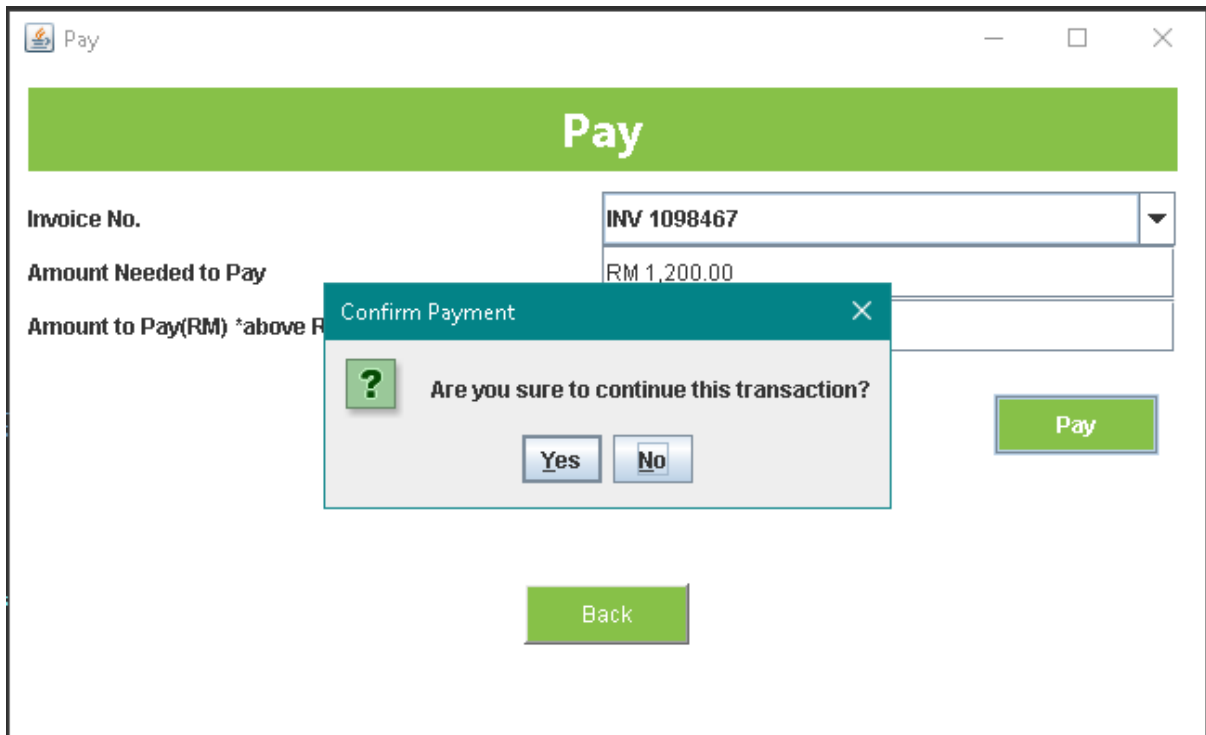
*Figure 7.0 The user chooses Yes when a message appears to confirm the transaction.*



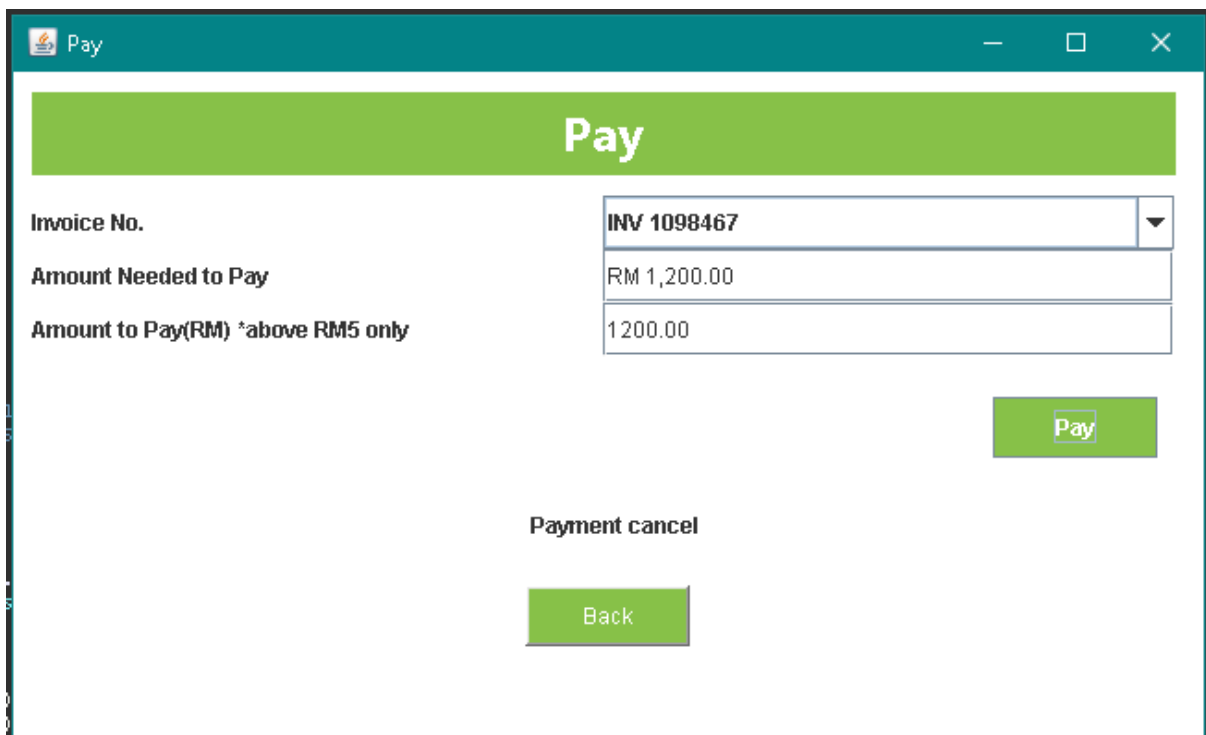
*Figure 8.0 A message appears when the transaction is successful.*



*Figure 9.0 The message displays an invalid amount when the user enters an amount less than RM5.00.*



*Figure 10.0 The user chooses No when a message appears to confirm the transaction.*



*Figure 11.0 A message appears to inform that the transaction has been canceled.*

**Receipt**

**2021**

No	Date	Receipt No.	Payment Information	Amount (RM)
1	17/10/2021	RST 1284480	Pembayaran hutang melalui Epayment : Bayaran e...	752.00
2	12/04/2021	RST 1198025	Pembayaran hutang melalui Epayment : Bayaran e...	831.00

**2020**

No	Date	Receipt No.	Payment Information	Amount (RM)
1	11/11/2020	RST 1128142	BAYARAN OLEH PENAJA LP15389 - (PTPTN) SP 4...	940.00
2	01/09/2020	RST 1098467	Pembayaran hutang melalui Epayment : [infoline] b...	1,200.00

[Back](#)

*Figure 12.0 Receipt page of the UKM e-Kewangan system.*

**Voucher**

No	Date	Receipt No.	Payment Information	Status	Amount(RM)
1	21/07/2021	BCR 1288527	EFT -UKM21072100573 : B...	Telah Dikreditkan	200.00
2	13/11/2020	BCR 1169377	EFT -UKM20111401787 : B...	Telah Dikreditkan	50.00

[Back](#)

*Figure 13.0 Voucher page of the UKM e-Kewangan system.*