

---

# **Software Requirements Specification for Student Budget Management System**

**Version 2.0 approved**

**Sufi Nukman Bin Shakimi  
Muhammad Alif Luqman Bin Afandi  
Nurhamizah Bt. Husin  
Eman Hussein**

**15 May 2020**

## Table of Contents

<b>1. Introduction.....</b>	<b>1</b>
1.1 Purpose.....	1
1.2 Scope .....	1
1.3 Definitions, Acronyms, and Abbreviations .....	1
1.4 References .....	1
<b>2. Overall Description .....</b>	<b>2</b>
2.1 Product Perspective.....	2
2.2 Product Functions .....	2
2.3 User Classes and Characteristics .....	4
2.4 Operating Environment .....	4
2.5 Design and Implementation Constraints .....	4
2.6 Assumptions and Dependencies .....	4
<b>3. External Interface Requirements .....</b>	<b>5</b>
3.1 User Interfaces .....	5
3.2 Hardware Interface .....	12
3.3 Software Interface .....	12
<b>4. System Use Cases .....</b>	<b>13</b>
4.1 Login (UC1).....	14
4.2 Register (UC2).....	15
4.3 Record Pocket Money (UC3) .....	16
4.4 View Allocation (UC4).....	17
4.5 Record Expenses and Transactions (UC5) .....	18
4.6 View Summary (UC6).....	19
<b>5. ERD Diagram .....</b>	<b>20</b>
<b>6. Other Non-functional Requirements.....</b>	<b>21</b>
6.1 Performance Requirements .....	21
6.2 Safety Requirements .....	21
6.3 Security Requirements .....	21
6.4 Software Quality Attributes .....	21

## 1. Introduction

### 1.1 Purpose

Student Budget Management (SBM) is a web platform system where it focuses on managing budget and finance specifically for students. This project is related to the finance field among UPM students. Nowadays, students are facing problems in managing their money in a proper way due to lack of management skill and ignorance of importance of management skill. That's the reason for us to choose students as our users. Therefore, we think that through this SBM project, we can help the students to manage their finance. We believe that this project can guide them the way to manage their finance throughout their studies based on their income like loans or scholarships. This project also improves the weaknesses in the current existing systems since most of them are not focusing on students' financial management. Therefore, this project is going to be more focused on students other than existing systems do.

### 1.2 Scope

The SBM focuses on managing UPM students' finances based on their income and pocket money every semester.

### 1.3 Definitions, Acronyms, and Abbreviations

Terms	Definitions
SBM	Student Budget Management
SRS	Software Requirement Specification
UPM	Universiti Putra Malaysia

### 1.4 References

- [1] Lecture notes, Software Requirement Engineering.
- [2] <https://softwarekno.blogspot.my/2016/09/waterfall-model.html>
- [3] <https://gyires.inf.unideb.hu/GyBITT/07/ch01.html>

## 2. Overall Description

### 2.1 Product Perspective

Student Budget Management System (SBMS) is intended to maintain the financial challenges faced by UPM students, and it benefits them effectively by providing features that allows them to record their budget, expenses and can view the effective allocations performed by the system and follow it. SBMS is a web platform management tool that requires a client/web browser, web server and database. Firstly, a web browser is responsible to deliver the requested pages and display to users, a web server is responsible for satisfying user's requests and processes them through the Internet. Lastly, the database is the storage element of user's data. Figure 2.1 illustrates the interaction.

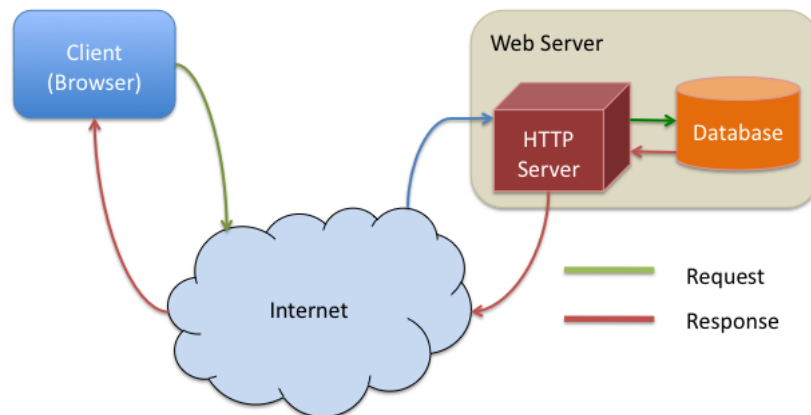


Figure 1: Product Perspective Diagram Illustration

### 2.2 Product Functions

In this section, the use case model is illustrated as well as described briefly. The system's use case diagram is shown in Figure 2.2.

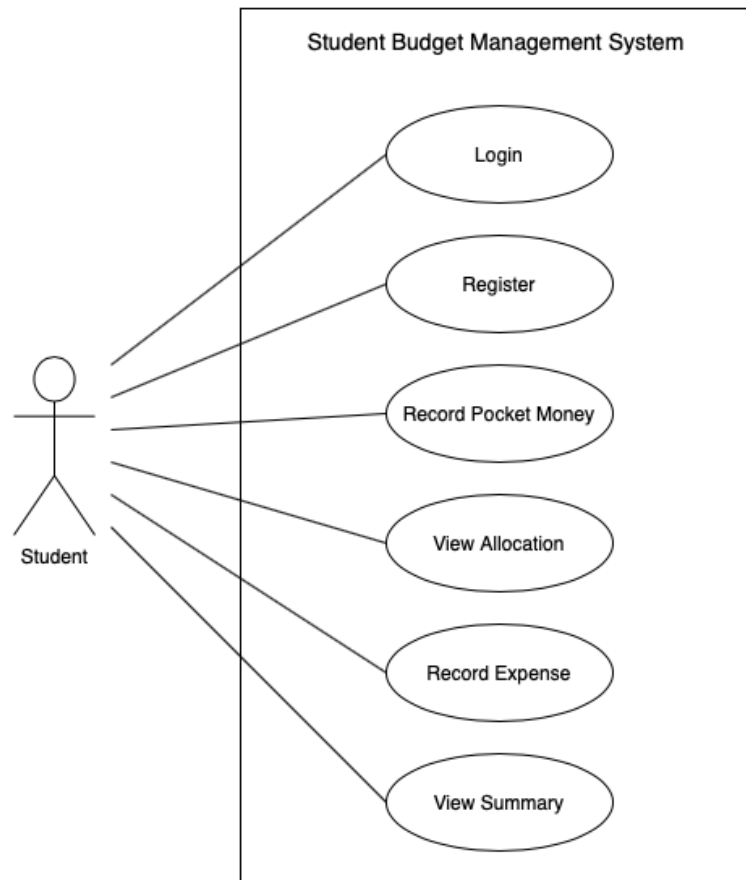


Figure 2: Use Case Diagram

Identifier	Use cases	Description
UC1	Login	This use case allows users to login and access the system features.
UC2	Register	This use case allows new users to register for an account.
UC3	Record Pocket Money	This use case allows students to input their total pocket money.
UC4	View Allocation	This use case allows students to view the allocation of each category calculated by the system.
UC5	Record Expense	This use case allows students to input their daily spent expenses.
UC6	View Summary	This use case allows students to view their summary of spending.

## **2.3 User Classes and Characteristics**

SBM contents only involve one type of user that is student. As a result, SBM will focus only for students. They can manage their budget by inserting their pocket money for a semester. After that, the system will automatically divide their pocket money into some categories so that the user can spend their money by the system's guidance without getting over budget. The system will also tell them how much can be spent on certain things. This application can be used by UPM students.

## **2.4 Operating Environment**

**OE1** - The system shall operate in all modern browsers.

**OE2** - The system shall operate in different operating systems such as Windows and iOS.

## **2.5 Design and Implementation Constraints**

**DC1** - SBM is created for university students since it is a public application.

## **2.6 Assumptions and Dependencies**

**AD1** - It is assumed that users shall have fair Internet connection to access the system.

**AD2** - It is assumed that users have experience in using basic web application features.

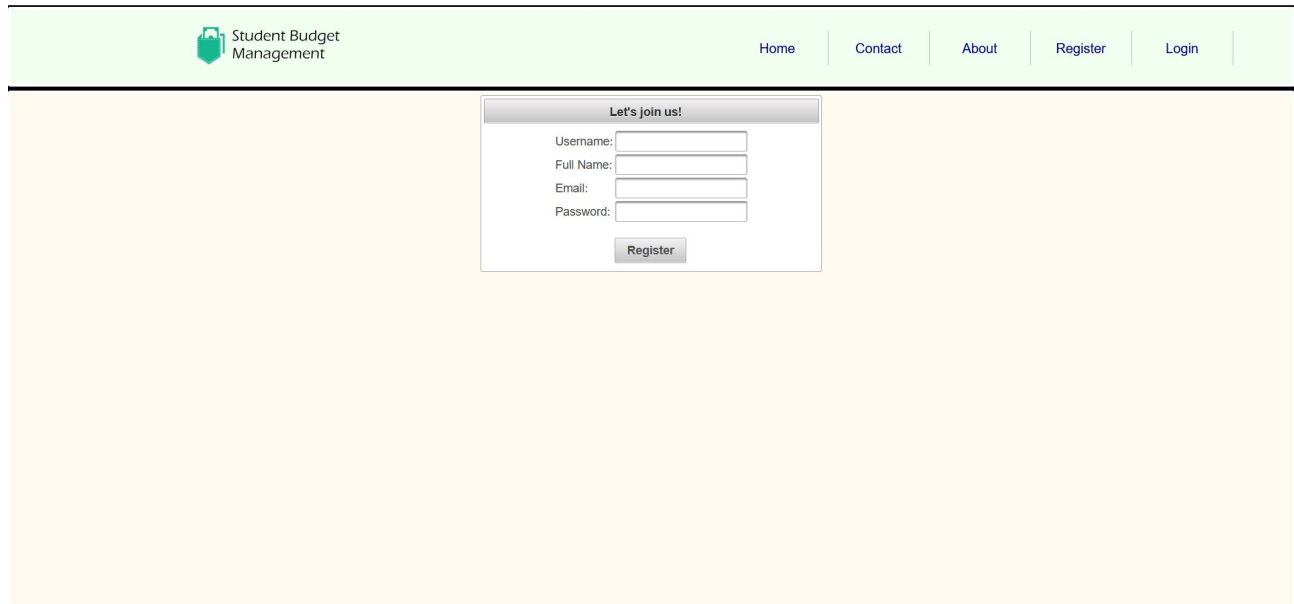
**AD3** - It is assumed that users understand English language.

**AD4** - The date picker is manipulated from a third-party component library named JCalendar.

**AD5** - The summary report provided by the system is manipulated from a third-party component library of graphs and charting.

### 3. External Interface Requirements

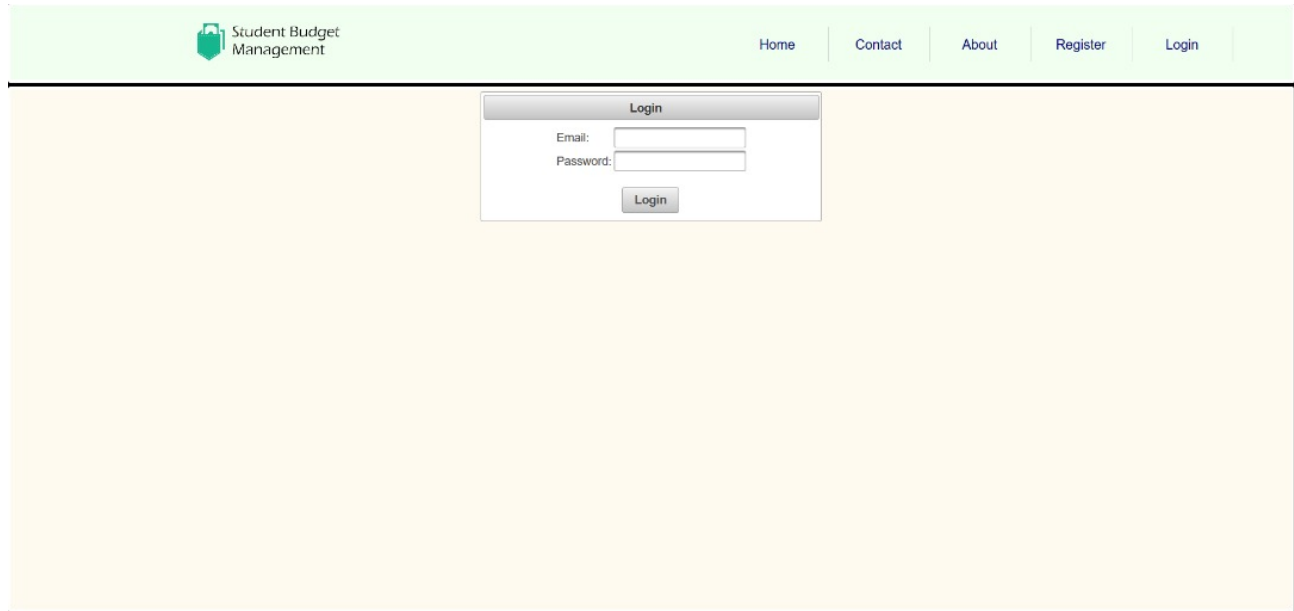
#### 3.1 User Interfaces



The screenshot displays the 'Register' user interface for the 'Student Budget Management' system. The page features a light green header bar with the system logo and name on the left, and a navigation menu with links for 'Home', 'Contact', 'About', 'Register', and 'Login' on the right. The main content area has a light yellow background. Centered on this background is a registration form titled 'Let's join us!'. The form contains four text input fields labeled 'Username:', 'Full Name:', 'Email:', and 'Password:'. Below these fields is a 'Register' button.

Figure 3

Figure 3 shows the interface of the 'Register' use case. It shows six text fields which are available for the user to enter their username, full name, email and password. The 'Register' button will let the system store the user's information.



The screenshot displays the login interface for the Student Budget Management (SBM) system. The header is light green and contains the SBM logo on the left and navigation links (Home, Contact, About, Register, Login) on the right. The main content area is light yellow and features a central login form. The form is titled 'Login' and contains two input fields: 'Email:' and 'Password:'. Below these fields is a 'Login' button.

Figure 4

Figure 4 shows the interface of the 'Login' use case. It shows two text fields which are available for the user to sign in the SBM with email and password. The 'Login' button will let the system verify and allow users to sign in.



The screenshot displays the 'Edit Your Profile Details' page within the 'Student Budget Management' application. The page features a light green header with the application logo and title on the left, and navigation links for 'Home', 'Contact', and 'About' on the right. A left sidebar contains a menu with categories: 'Home' (with a sub-link 'Summary'), 'Pocket Money' (with sub-links 'Record Pocket Money', 'View Allocation', and 'Record Expense'), 'Profile' (with a sub-link 'Profile'), and 'Setting' (with a sub-link 'Logout'). The main content area has a breadcrumb trail 'Profile > Edit Profile' and the title 'Edit Your Profile Details'. It contains four text input fields labeled 'Username:', 'Name:', 'Email:', and 'Password:'. Below these fields is a 'Save Changes' button.

Figure 5

Figure 5 shows the interface for editing the user's profile. It shows the editable user's profile information. Users can update his/her profile information by editing the text fields. The 'Save Changes' will allow the system to update the user's profile information.

The screenshot displays the 'Record Pocket Money' page of a web application. The header is light green with the 'Student Budget Management' logo and navigation links for 'Home', 'Contact', and 'About'. A left sidebar contains menu items: 'Home' (with a sub-item 'Summary'), 'Pocket Money' (with sub-items 'Record Pocket Money', 'View Allocation', and 'Record Expense'), 'Profile' (with a sub-item 'Profile'), and 'Setting' (with a sub-item 'Logout'). The main content area has a breadcrumb trail 'Pocket Money > Record' and a title 'Record Pocket Money' with the instruction 'Please enter your pocket money for us to calculate your budget allocation'. Below this is a form titled 'Pocket Money Record' containing a text input for 'Pocket Money', a dropdown menu for 'Month' (currently showing 'January'), and a 'Submit' button. To the right of the form is a large illustration of a pink piggy bank with gold coins and plus signs floating around it.

Figure 6

Figure 6 shows the interface of 'Record pocket money' use case. There is one text field allowing the user to enter his/her pocket money and a drop down list for the user to choose the month. The 'Submit' button triggers the system to store the amount of the pocket money.

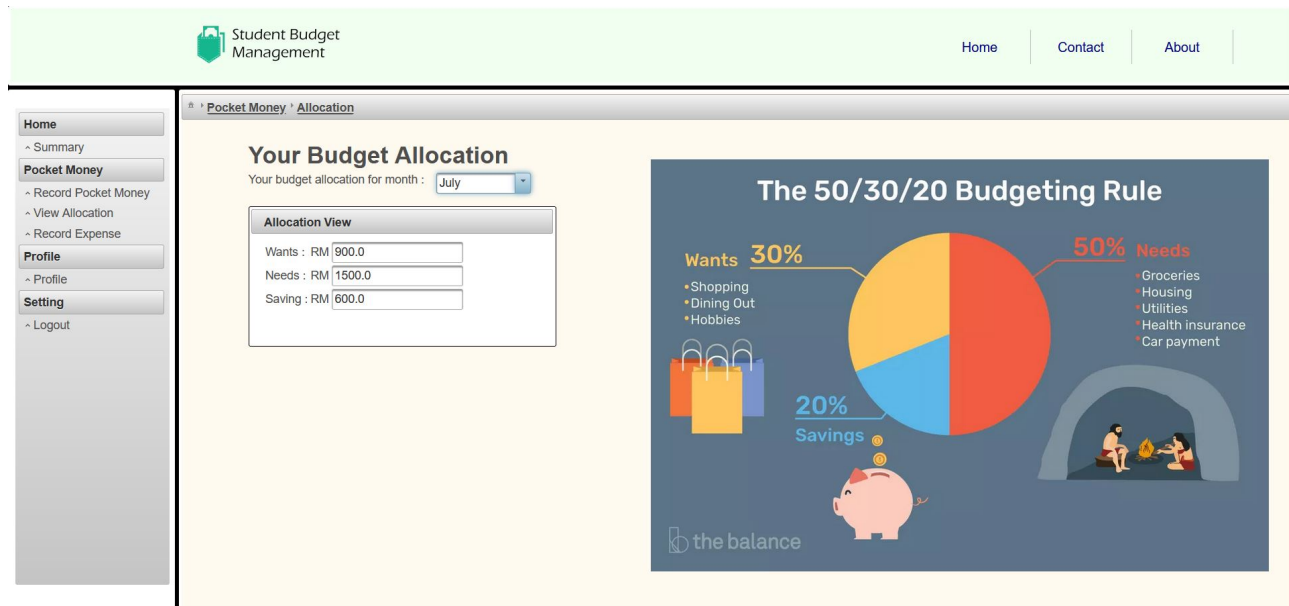


Figure 7

Figure 7 shows the interface of the 'View Allocation' use case. It shows the allocations of the pocket money into 3 main categories: wants, needs and savings.

The screenshot displays the 'Record Expense' interface within the 'Student Budget Management' application. The top navigation bar includes the application logo and name, and links for 'Home', 'Contact', and 'About'. A left sidebar menu contains sections for 'Home' (Summary), 'Pocket Money' (Record Pocket Money, View Allocation, Record Expense), 'Profile' (Profile), and 'Setting' (Logout). The main content area is titled 'Pocket Money > Record Expense' and features a form titled 'Record How Much You've Spent!'. This form contains two text input fields for 'Date' and 'Amount', followed by three radio button options for 'Expensed On': 'Wants (Shopping , Food , Hobbies , etc...)', 'Needs (Groceries, Utilities, Housing , etc...)', and 'Saving'. A 'Record' button is positioned at the bottom right of the form.

Figure 8

Figure 8 shows the interface of 'Record Expense' use case. It shows two text fields for the user to enter the date and amount he/she has spent. Besides, there are radio buttons for the categories (wants, needs and savings) that he/she already spent.

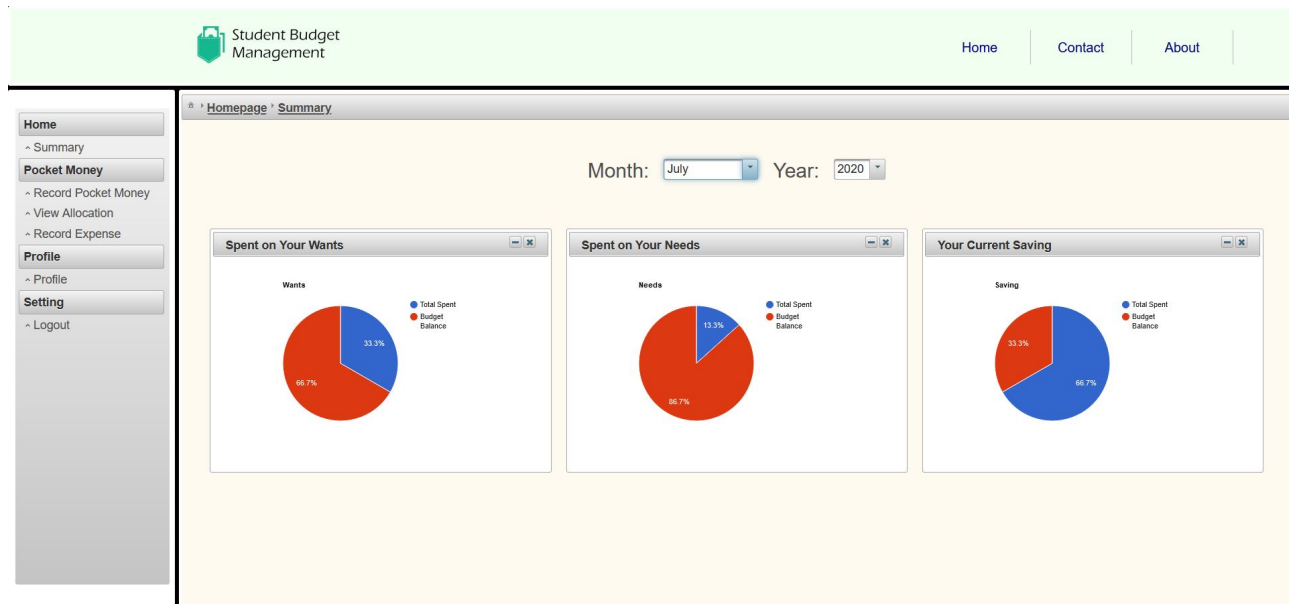


Figure 9

Figure 9 shows the interface of the 'View Summary' use case. It shows the summary of the total money the user has saved, the total amount of money the user has spent and total spent per month.

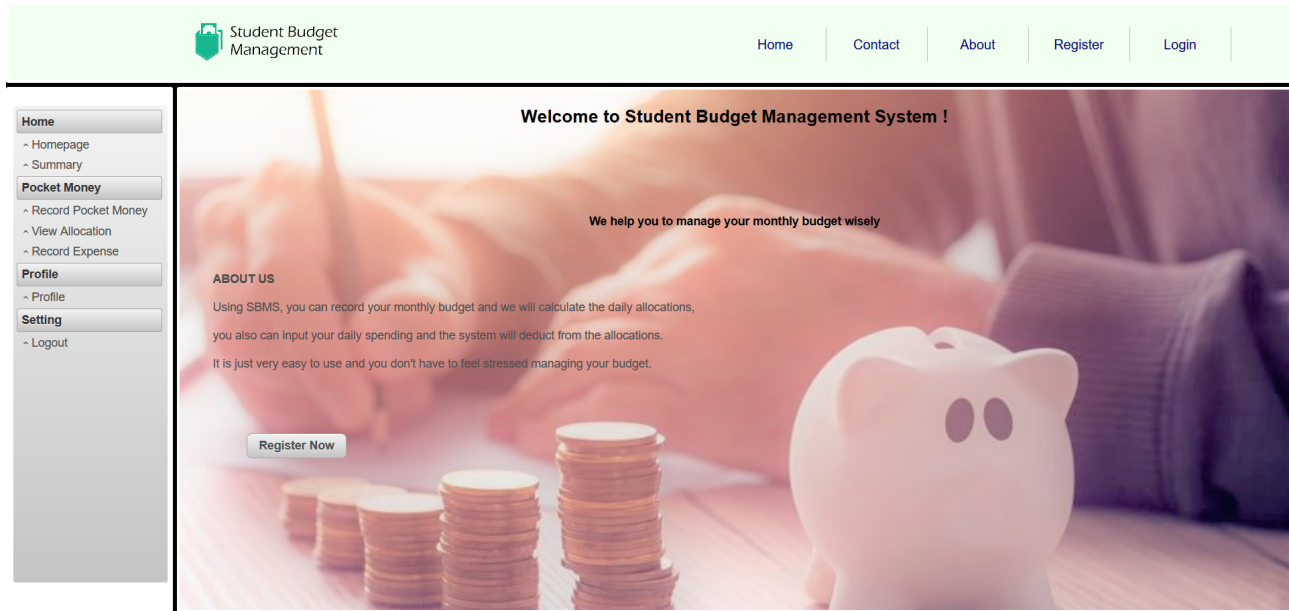


Figure 10

Figure 10 shows the interface of the homepage of SBM. It gives a brief description of SBM to the new user.

## 3.2 Hardware Interface

For Student Budget Management to run, you will need a PC (regardless of its operating system, it can be Windows or Linux) or Mac, with an operating system. A functional standard keyboard and mouse is required.

## 3.3 Software Interface

- Browser: Google Chrome, Mozilla Firefox, Internet Explorer, Safari and Opera.
- Server: Java Application Server
- Database: SQL Database
- External component: Google Chart

## 4. System Use Cases

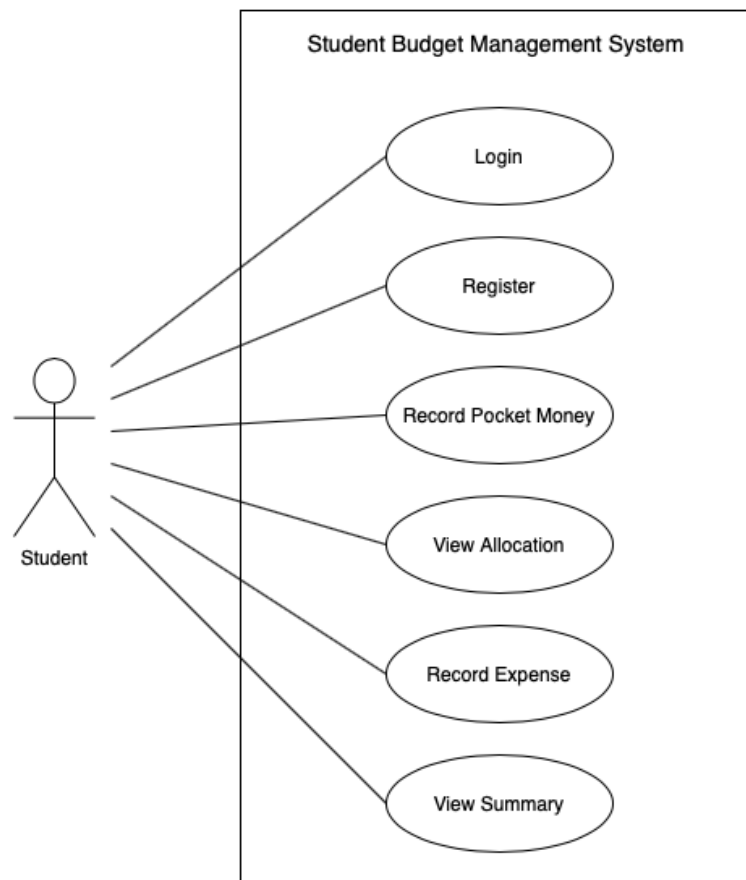


Figure 11

Figure 11 is a use case diagram. There are 6 use cases, they are Login, Register, Record Pocket Money, View Allocation, Record Expenses and View Summary.

## 4.1 Login (UC1)

<b>Description</b>	This use case allows users to login and access the system features.
<b>Objective</b>	To store the login data safely only for users.
<b>Actors</b>	Students
<b>Basic Flow of Events</b>	<ol style="list-style-type: none"><li>1. User has to press login button</li><li>2. System will display text field to insert ID and password</li><li>3. User has to insert ID and password</li><li>4. User has to press Login button</li><li>5. System will validate password</li><li>6. User can login successfully</li></ol>
<b>Alternative Flow of Events</b>	<ol style="list-style-type: none"><li>1. At step 2.1.5 error message is prompt and ask the user to enter again the password</li></ol>
<b>Preconditions</b>	Starts when the user opens the web application.
<b>Postconditions</b>	User is successfully logged into the system and able to access its features



## 4.2 Register (UC2)

<b>Description</b>	This use case allows new users to register for an account.
<b>Objective</b>	To store students personal information.
<b>Actors</b>	Students
<b>Basic Flow of Events</b>	<ol style="list-style-type: none"><li>1. User have to click create profile button</li><li>2. System will display empty form</li><li>3. User have to fill in the information</li><li>4. System will ask the user to confirm</li><li>5. User have to confirm the profile</li><li>6. System will store the profile information</li><li>7. System will display the profile</li></ol>
<b>Alternative Flow of Events</b>	<ol style="list-style-type: none"><li>1. At step 2.2.5 error message is prompt and asks the user to input valid information.</li></ol>
<b>Preconditions</b>	Starts when successfully logged in.
<b>Postconditions</b>	The user is successfully registered to the system.

### 4.3 Record Pocket Money (UC3)

<b>Description</b>	This use case allows students to input their total pocket money.
<b>Objective</b>	To know students pocket money in order to calculate allocation.
<b>Actors</b>	Students
<b>Basic Flow of Events</b>	<ol style="list-style-type: none"><li>1. User have to press Record Pocket Money button</li><li>2. System will display empty form</li><li>3. User have to fill in the information</li><li>4. System will save the pocket money</li><li>5. System will display the confirmation</li></ol>
<b>Alternative Flow of Events</b>	-
<b>Preconditions</b>	Starts when successfully logged in.
<b>Postconditions</b>	Student's pocket money is successfully recorded.

#### 4.4 View Allocation (UC4)

<b>Description</b>	This use case allows students to view the allocation of each category calculated by the system.
<b>Objective</b>	To calculate the allocation according to categories and display them.
<b>Actors</b>	Students
<b>Basic Flow of Events</b>	<ol style="list-style-type: none"><li>1. User have to select the view allocation button</li><li>2. System will calculate the money</li><li>3. System will save the allocation</li><li>4. System will display the allocation</li></ol>
<b>Alternative Flow of Events</b>	-
<b>Preconditions</b>	Starts when the pocket money is recorded.
<b>Postconditions</b>	Student's money allocation is successfully displayed.

#### 4.5 Record Expenses and Transactions (UC5)

<b>Description</b>	This use case allows students to input their daily spent expenses.
<b>Objective</b>	To record daily expenses and transactions to deduct it from the allocation.
<b>Actors</b>	Students
<b>Basic Flow of Events</b>	<ol style="list-style-type: none"><li>1. User have to press the record transaction and Expense button</li><li>2. User have to choose the type of record</li><li>3. User have to fill in the details</li><li>4. User have to choose the category</li><li>5. User have to submit</li><li>6. System will deduct the amount of money</li><li>7. System will save the allocation</li></ol>
<b>Alternative Flow of Events</b>	<ol style="list-style-type: none"><li>1. At 2.5.7 reminder message if the expense exceeds allocation after saved.</li></ol>
<b>Preconditions</b>	Starts when the user has successfully logged in.
<b>Postconditions</b>	The balance amount is successfully saved.

#### 4.6 View Summary (UC6)

<b>Description</b>	This use case allows students to view their summary of spending .
<b>Objective</b>	To show the total money that the user has spent over a month.
<b>Actors</b>	Students
<b>Basic Flow of Events</b>	1. User clicks on View Summary button 2. System will display the monthly summary
<b>Alternative Flow of Events</b>	-
<b>Preconditions</b>	Starts when the user has successfully logged in and one month of expenses have been recorded.
<b>Postconditions</b>	The summary is successfully displayed.

## 5. ERD Diagram

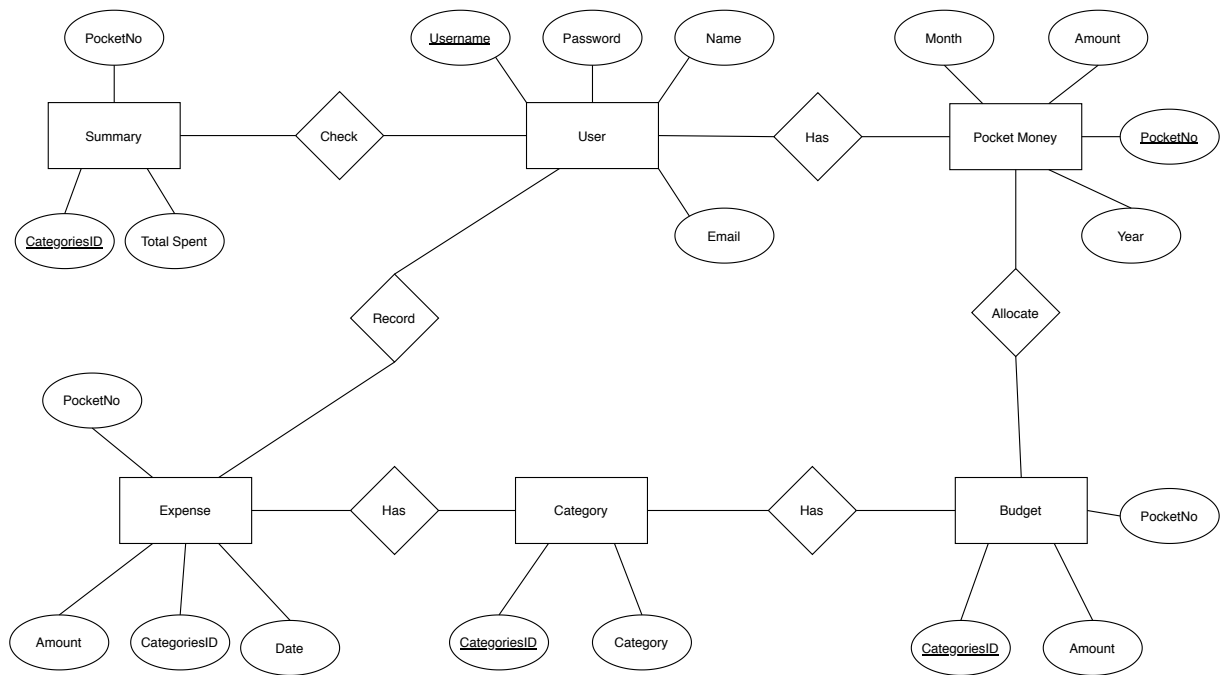


Figure 12

Figure 12 shows the ERD diagram which consists of six entities: User, Expense, Pocket Money, Category, Budget, and Summary with their own attributes. For the User entity, it has a primary key which is username and other three attributes which is name, email, password. Also for the Pocket Money entity, it contains one primary key which is pocketNo and another attribute are amount, month , and year. For the Expense entity, it has four attributes which are amount, categoriesID, pocketNo and date. This entity does not have a primary key. For a Category entity, it has two attributes which is categoriesID and category where categoriesID is a primary key for this entity. Another entity is Budget. For this entity, it contains three attributes with one primary key which is amount, pocketNo and categoriesID respectively. The last one is Summary entity that contains also three attributes with one primary key which is total spent, pocketNo and categoriesID respectively. The user that has pocket money can allocate their budget by having or choosing which category that they will expense their money. Users also can record their expenses and check for the summary of their pocket money.

## **6. Other Non-functional Requirements**

### **6.1 Performance Requirements**

The system will display the Main page of the SBM application after 2 seconds when the app is opened. It requires 2 seconds when the user presses “Start” button in order to display the login page. To display the login page, it requires 10 seconds. When the user key-in their username and password. When the login is successful, the app will go to the Menu page after 2 seconds. In the Menu page, when the user clicks one of the options, the system will display the interface of the options that the user chose in 2 seconds.

### **6.2 Safety Requirements**

The user’s data is stored in an efficiently designed database which is only accessible by developers which eliminates any data loss.

### **6.3 Security Requirements**

The system provides an authentication facility which allows only authorized users to access the system using their login credentials. Accordingly, the login feature provided mitigates the probability of the system to be attacked.

### **6.4 Software Quality Attributes**

#### **6.4.1 Availability**

The system is available for UPM students 24/7. In cases of the system being out of service due to any error or crash, the system will be recovered at most within half a day.

#### **6.4.2 Maintainability**

The system will be developed with high modularity which promotes low coupling between modules. As a result, this design decision enhances maintainability of the system in the future whereby modules are maintained individually when necessary.

#### **6.4.3 Correctness**

Calculation algorithms are correctly applied in the system as well as producing correct results that are efficiently interpreted by users. To illustrate that, the alert must be appeared if the user uses the money which exceeds the budget.

#### **6.4.4 Usability**

The system provides a user-friendly and simple user interface for users to effectively interact with. It is expected that users do not need any training or tutorials to use the system; moreover, the system is easily learned from the first time used and the flow of interaction and tasks can be remembered in further visits.

#### **6.4.5 Efficiency**

Any given task in the system shall be completed by a short and fast procedure that requires few button clicks and second to accomplish the task. For example, three button clicks per task and a total of 5 seconds to get the task done efficiently.