

# FACULTY OF ENGINEERING & COMPUTING

**PROGRAMME** : Bachelor of Software Engineering (Hons);

Bachelor of Computer Science (Hons) in Intelligent

Systems

**ACADEMIC YEAR** : 2020

**MODULE** : Visual Programming

**MODULE CODE** : CSWD2010

**MODULE LEADER** : Ms. Tan Siok Yoon **ASSIGNMENT TYPE** : Group Assignment **INTAKE/GROUP** : Jan 2020

**DISTRIBUTION DATE** : Friday, 14 February 2020

**SUBMISSION DATE** : Friday, 24 April 2020 4:00 PM

|  |  |  |
| --- | --- | --- |
| **STUDENT ID** | **STUDENT NAME** | **Remarks** |
| B0633 | Choo Han Yang |  |
| B1472 | Neoh Kim Seng |  |
| B0766 | Tang Hong Yip |  |

Table of Contents

[**Designs** 3](#_Toc40611947)

[**Entity Relation Diagram (ERD)** 3](#_Toc40611948)

[**Table** 4](#_Toc40611949)

[**View** 5](#_Toc40611950)

[**Use Case Diagram** 6](#_Toc40611951)

[**Persistence Storage** 7](#_Toc40611952)

[**Data storage** 7](#_Toc40611953)

[**Data Retrieval** 7](#_Toc40611954)

[**Graphical user interface (GUI) implementation** 10](#_Toc40611955)

[**Navigation Sidebar** 10](#_Toc40611956)

[**GUI Components** 12](#_Toc40611957)

[**Authentication and authorization** 21](#_Toc40611958)

[Authorization 21](#_Toc40611959)

[Authentication 21](#_Toc40611960)

[**User Guidelines** 22](#_Toc40611961)

[User – Login 22](#_Toc40611962)

[User – Register 23](#_Toc40611963)

[User – Add budget 24](#_Toc40611964)

[User – Delete budget 25](#_Toc40611965)

[User – Update budget 26](#_Toc40611966)

[User – Add expenditure 27](#_Toc40611967)

[User – Delete expenditure 28](#_Toc40611968)

[User – Update expenditure 29](#_Toc40611969)

[User – Show graph 30](#_Toc40611970)

[User – Log out 31](#_Toc40611971)

[User – Change password 33](#_Toc40611972)

[Admin – Login 34](#_Toc40611973)

[Admin – Find specific user 35](#_Toc40611974)

[Admin – Show all the user 36](#_Toc40611975)

[Admin – Update user 37](#_Toc40611976)

[Admin – Delete user 38](#_Toc40611977)

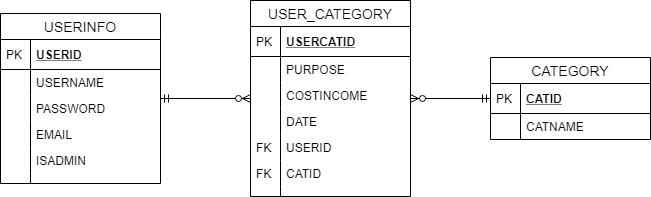
[Admin – Log out 39](#_Toc40611978)

[Admin – Change password 41](#_Toc40611979)

Our project can be seen in the GitHub with this link : <https://github.com/neohks/PersonalFinancialSystem-VP->

## **Designs**

### **Entity Relation Diagram (ERD)**



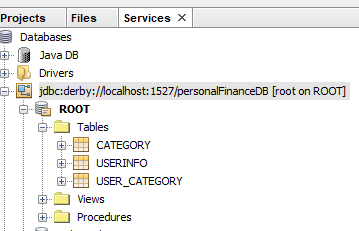
**Figure 1.0** ERD of Personal Finance System

Our table consists of 3 tables which are userinfo, user\_category, and category. The userinfo table is created to store the information about the user, the user ID, username, password, email and identify if the user has permissions of an admin.

Next, the category table stores the category ID and the category name for expenses and income. Lastly, user\_category table stores the transaction ID, purpose of the transaction, the amount, date performed. This table also stores the transaction is performed by which user and the category of money flow.

Below are the codes for creating the tables and view that are used for this project.

### **Table**

****

**Figure 1.1** List of Tables

**create table category (**

**catID varchar(5) not null primary key,**

**catname varchar(100) not null**

**);**

**create table userinfo (**

**userid varchar(5) not null primary key,**

**username varchar(20) not null,**

**password varchar(40) not null,**

**email varchar(50),**

**isadmin boolean not null**

**);**

**create table user\_category (**

**usercatID varchar(6) not null primary key,**

**purpose varchar(100) not null,**

**costincome double precision not null,**

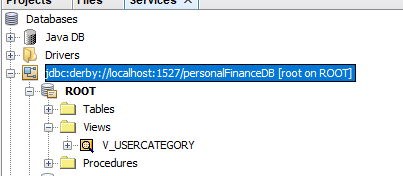
**date date not null,**

**userid varchar(5) not null, foreign key(userid) references userinfo(userid),**

**catid varchar(5) not null, foreign key(catid) references category(catid)**

**);**

### **View**

****

**Figure 1.2** View

**CREATE VIEW V\_USERCATEGORY**

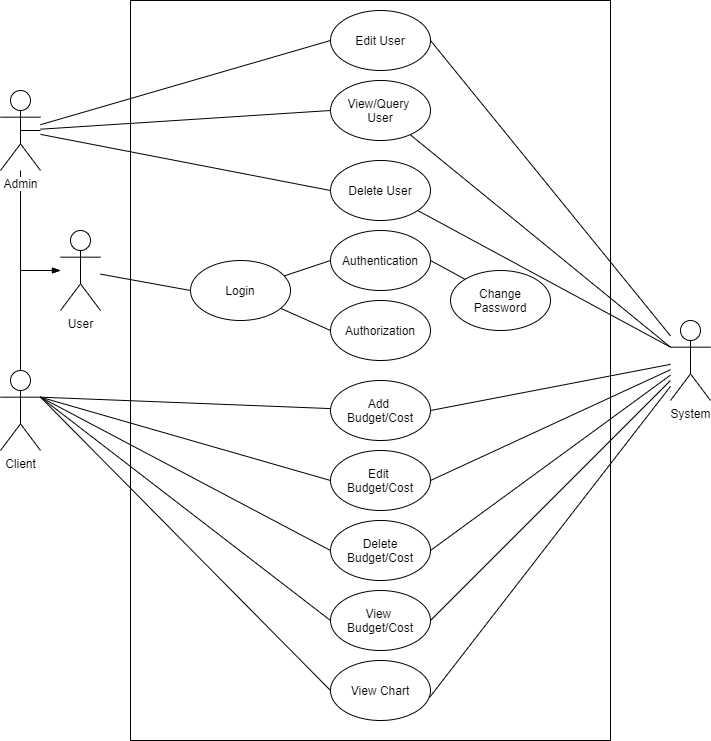
**AS SELECT uc.USERCATID, c.CATNAME, uc.PURPOSE, uc.COSTINCOME, uc."DATE", uc.USERID**

**FROM USER\_CATEGORY uc, CATEGORY c**

**WHERE uc.CATID=c.CATID;**

### **Use Case Diagram**

Based on Figure 1.3 below, each user must login before using the application. There are 2 types of users, which are Admin and Client. Admin can manage the Client accounts. They can either Query specific username, View users, Delete users and Edit users information. Whereby the Clients can Add deposit and expenditure, Edit budget and expenditure, Delete deposit and expenditure, View deposit and expenditure as well as View a monthly expenses chart. Both users can change their desired password.



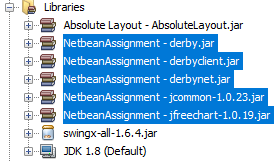
**Figure 1.3** Use Case Diagram

## **Persistence Storage**

### **Data storage**

#### **Derby DB**

For data storage, we chose Derby DB which is an open source relational database that is implemented entirely in Java. Derby DB is based on Java JDBC and SQL standards and provides an embedded JDBC driver that allows users to embed Derby in any Java-based solution.



The figure above shows the libraries that are included in this program.

DBConnection.java (SQL)

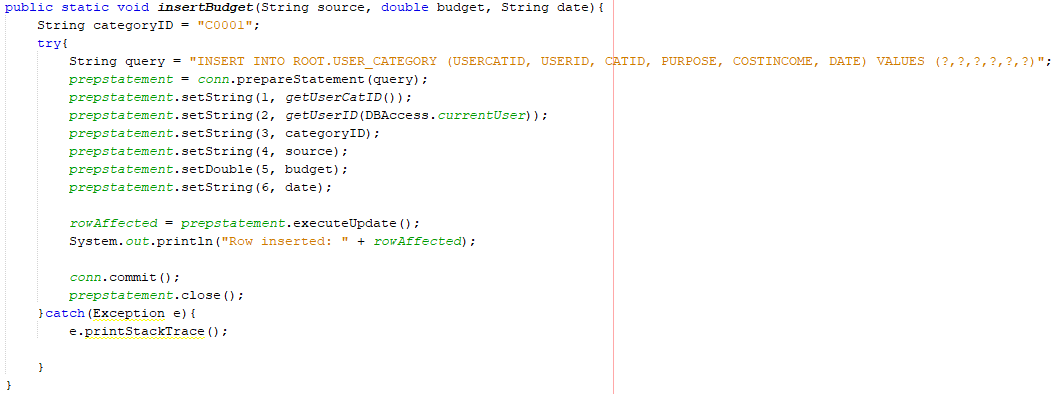
This acts like a bridge between application and Derby DB, allowing the application to access Derby DB.

### **Data Retrieval**

#### **DBAccess (CRUD)**

DBAccess is a java file that contains all the CRUD methods for the database. Below are some of the examples from the .java file.

##### **Create**

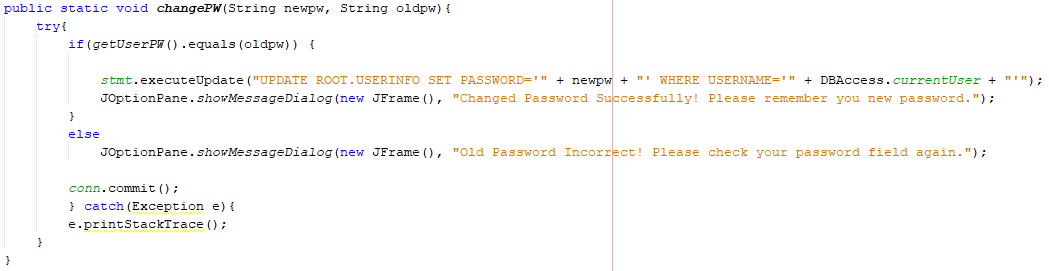


The program can create or insert a new row of data into the database by executing the query as shown above.

##### **Read**

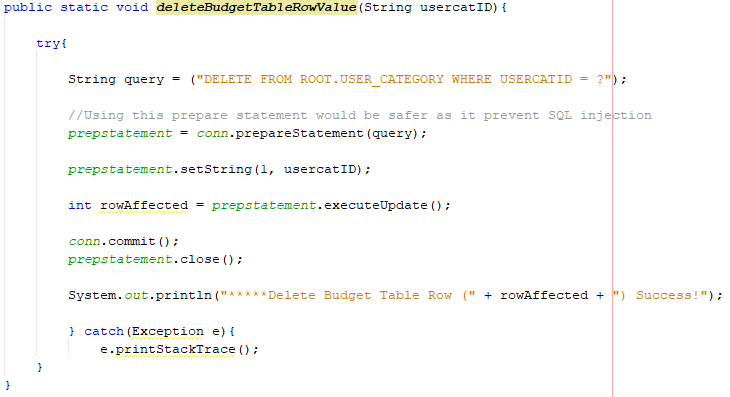
The program will access the database by executing the select query, as shown above the program is retrieving the username, password from table userinfo.

##### **Update**



The database can be updated by executing update queries. As shown above, the program is updating the password of the user.

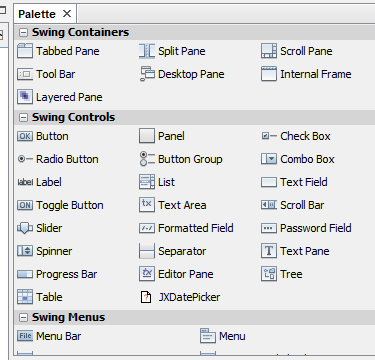
##### **Delete**



Inserted data can be deleted from the database by executing delete statements. Queries are executed by using prepared statements as it prevents any SQL injection.

## **Graphical user interface (GUI) implementation**

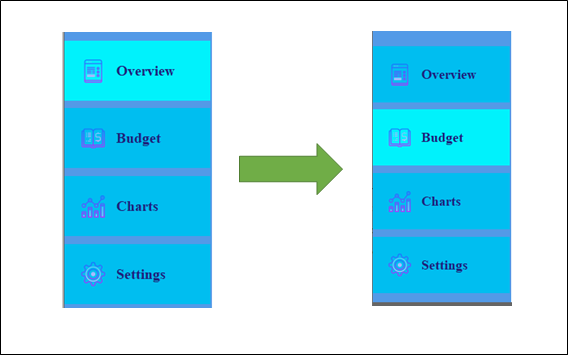
Graphic User Interface (GUI) is important in terms of Visual Programming especially user friendly ones. It can be created in all shapes and sizes using the components that Java has provided. In Netbeans, developers can use drag and drop to design the application’s layout. It is very convenient to design. Inside this Palette as shown below Figure 3.0, it can be dragged out to the components to the Design layer.



**Figure 3.0** Palette

### **Navigation Sidebar**

The navigation sidebar is to allow users to navigate the application when needed. For Admin, it only has 2 navigation panels, one is Overview, another is Settings. Whereby for User, it has 4 navigation panels which are, Overview, Budget, Charts and Settings. When selecting a navigation panel, as Figure 3.1 above shown below, it will have a brighter color compared to others. It uses a Card Layout with a Layered Panel, so it will only switch between the panels.



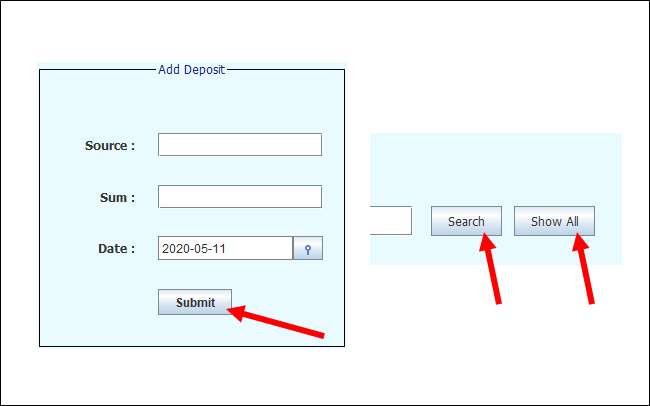
**Figure 3.1** Navigation Sidebar switch color

### 

### **GUI Components**

#### JButton

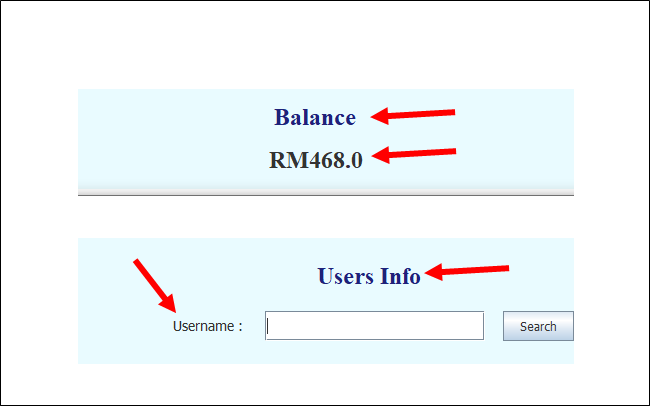
As Figure 3.2 below shows examples of JButton. It is used to create a labeled button that has platform independent implementation. The application results in some action when the button is pressed or clicked.



**Figure 3.2** JButtons

#### JLabel

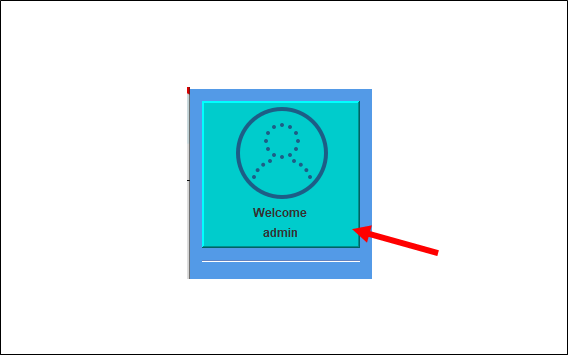
As Figure 3.3 below shows some examples of JLabel. It is used to display a single line of read only text. The text can be changed by an application but a user cannot edit it directly.



**Figure 3.3** JLabel

#### JPanel

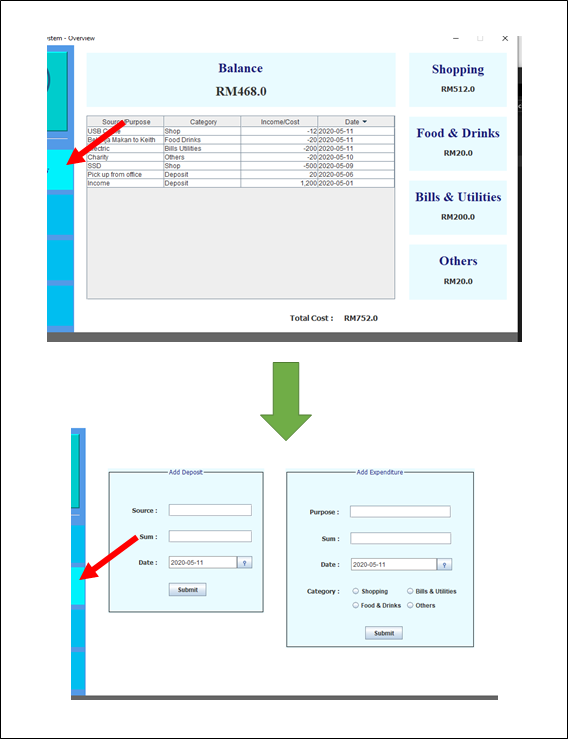
As Figure 3.4 below shows an example of JPanel. It acts as a container so you can customize better. It provides space in which an application can attach any other component. For this instance, it have 3 JLabels contained inside.



**Figure 3.4** JPanel

#### JLayeredPane

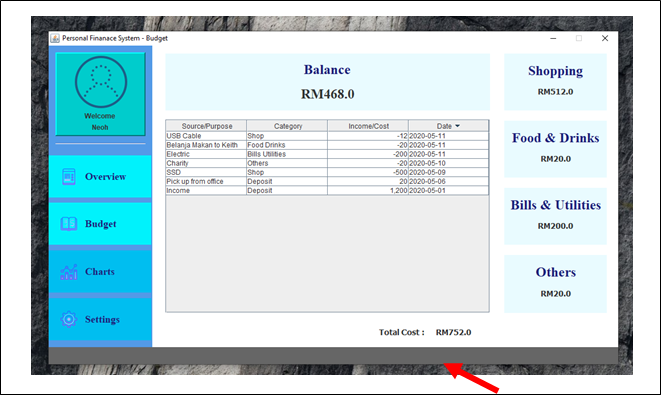
As Figure 3.5 shown below, it is a JLayeredPane. It is used to add depth to the swing container. It is used to provide a third dimension for positioning components and divide the depth-range into several different layers. For example, if the user select another panel, it will be the same frame since it just switches to a different layer of each JPanel.



**Figure 3.5** JLayeredPane

#### JFrame

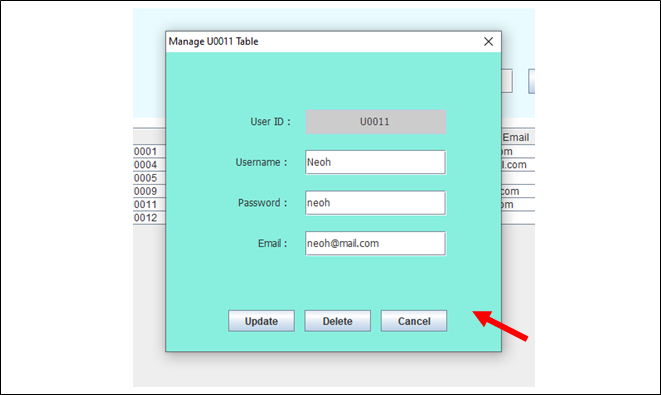
As Figure 3.6 shown below, it is a JFrame, the whole window is the JFrame. It works like the main window where components like labels, buttons, text fields are added to create a GUI.



**Figure 3.6** JFrame

#### JDialog

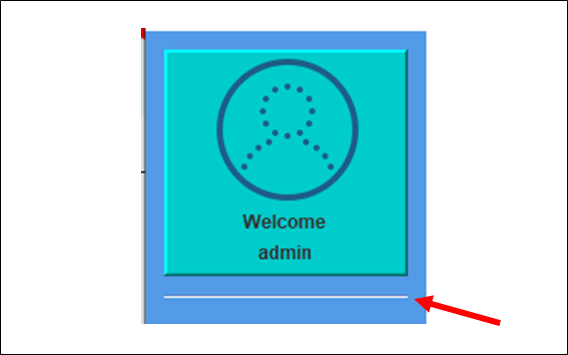
As Figure 3.7 shown below, it is a JDialog. It looks similar to a JFrame, but it only can be shown when a JFrame exists and does not have maximize and minimize buttons. It represents a top level window with a border and a title used to take some form of input from the user.



**Figure 3.7** JDialog

#### JSeparator

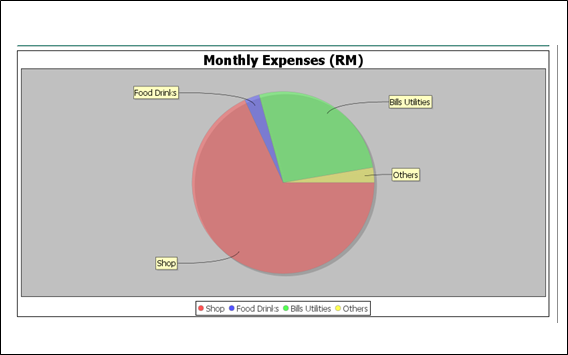
As Figure 3.8 shown below, that white line is a JSeparator. It provides a general purpose component for implementing divider lines. It is used to draw a line to separate widgets in a Layout.



**Figure 3.8** JSeparator

#### JFreeChart

As Figure 3.9 shown below, It is a JFreeChart that uses a pie chart to show monthly expenses. It is an open-source Java library for creating charts and graphs.



**Figure 3.9** JFreeChart (Pie Chart)

#### JXDatePicker

As Figure 3.10 shown below, It is an example for JXDatePicker. It must add swingx-all-1.6.4.jar file into the library in order to use this component. When the user clicks, it will show a calendar to allow the user to select.

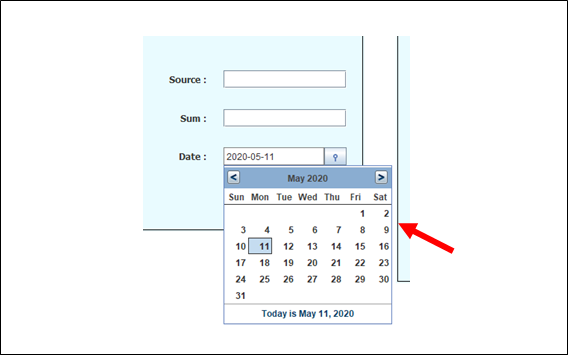
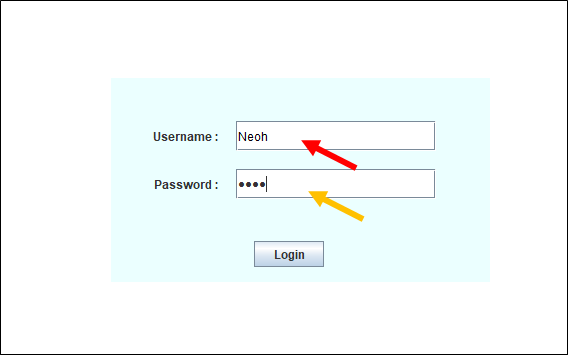


Figure 3.10 JXDatePicker

#### JTextField and JPasswordField

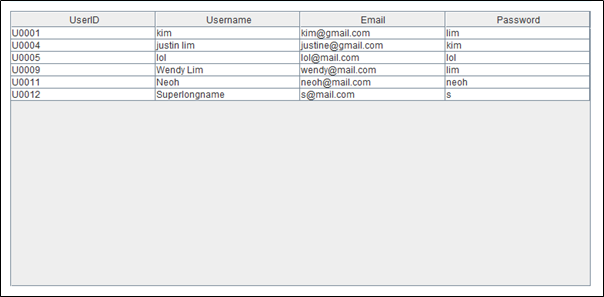
In Figure 3.11 shown below are JTextField with red arrow and JPasswordField with yellow arrow. Both act the same function to let user input. However, a JPasswordField will be hidden as it is specialized with password entry.



**Figure 3.11** JTextField and JPasswordField

#### JTable

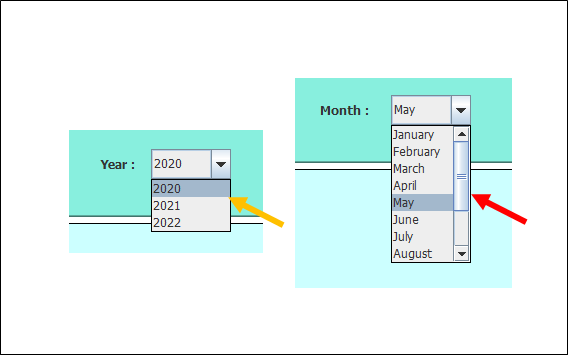
In Figure 3.12 shown below, it is a JTable. It is used to display data in tabular form. It is composed of rows and columns. For example here there are 4 columns with 6 rows of data.



**Figure 3.12** JTable

#### JComboBox and Custom ComBoBox

As shown in Figure 3.13 below, the yellow arrow pointed is JComboBox and while the red arrow pointed is a Customized JCombobox. Both are used to show a popup menu of choices. The choices selected by a user will be highlighted in the popup. The Customized JCombobox’s code can be seen in ComBoBoxCustom.java in the custom package.



**Figure 3.13** JCombobox and Customized JCombobox

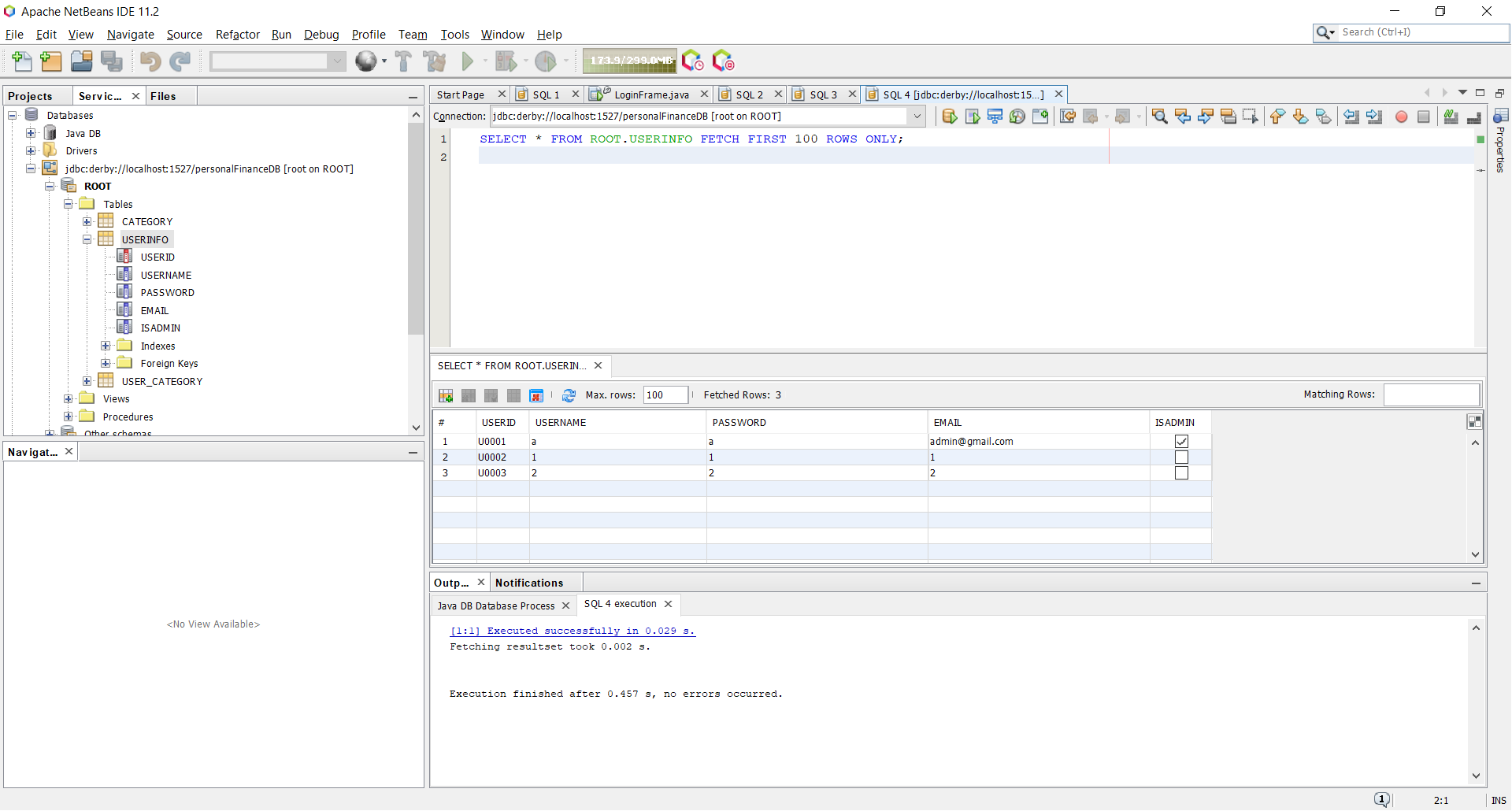
## 

## 

## **Authentication and authorization**

### Authorization

Each user has their details stored in our database. In the userinfo table, the isAdmin column helps us to differentiate each user based on their authority.



### Authentication

User

The user can register their own account using the system. The user can login using the system. The user can add, update and delete their budget. The user can add, update and delete their expenditure. The user can view a chart report of their monthly expenditure. The user can change their new password.

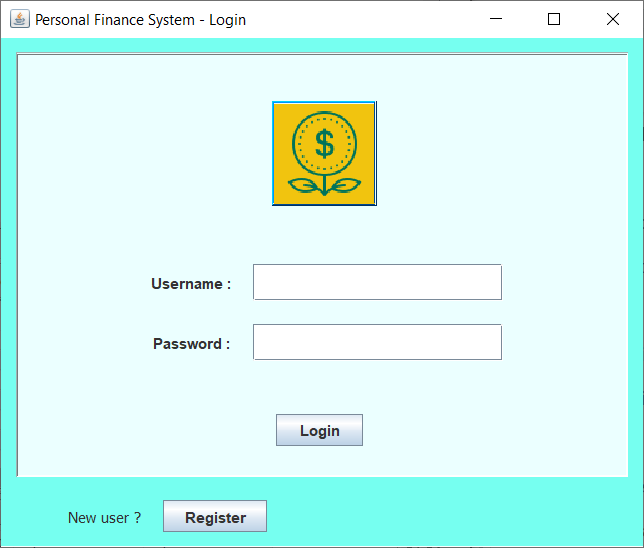
Admin

The admin can login using the system. The admin can update and delete the users. The admin can search for a specific user. The admin can change their new password.

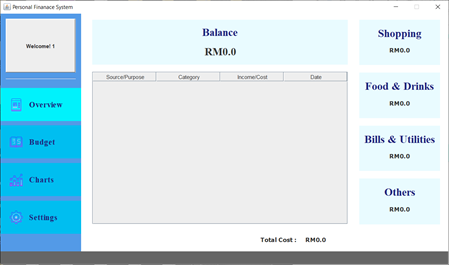
## **User Guidelines**

### User – Login

Step 1: Insert your username and password.

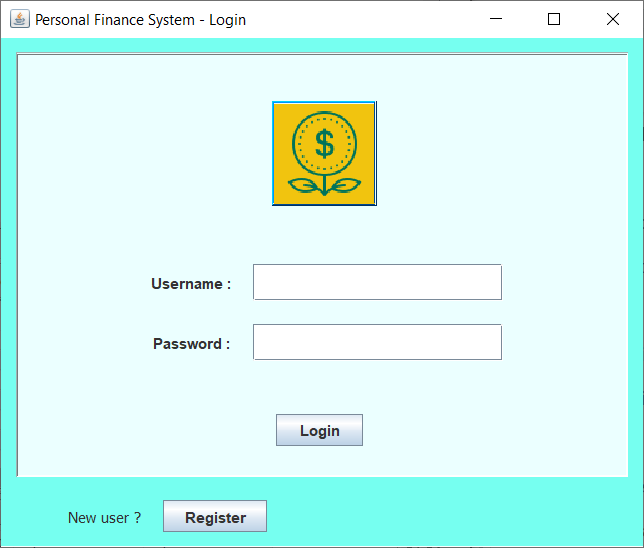


Step 2: Press the Login button and you have logged in the main page.

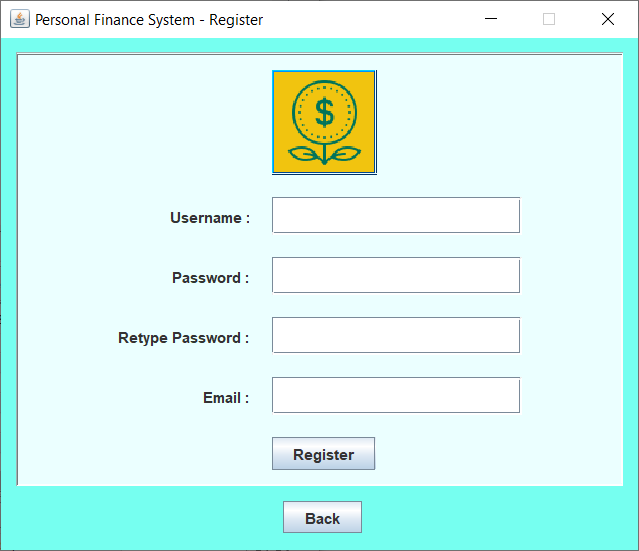


### User – Register

Step 1: Press the Register button.



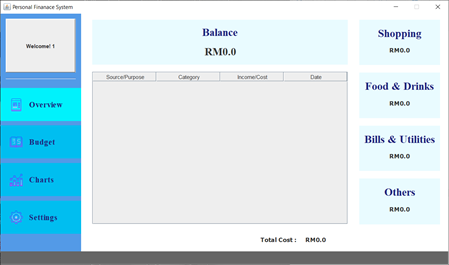
Step 2: Fill in all your personal details.



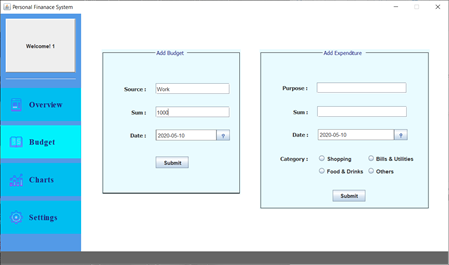
Step 3: Press the Register button when done.

### User – Add budget

Step 1: Click the Budget button on the navigation bar on the left side of the main page.



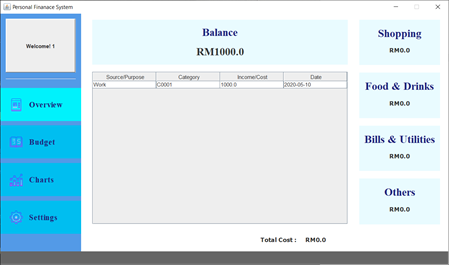
Step 2: Fill in your budget details.



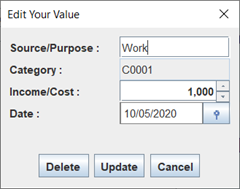
Step 3: Press the Submit button when done.

### User – Delete budget

Step 1: Double click the budget you want to delete.

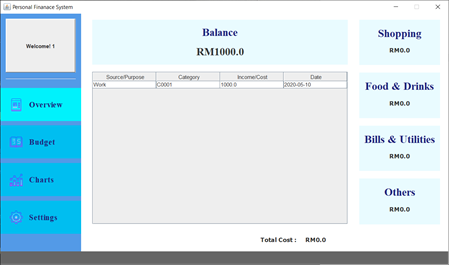


Step 2: Press the Delete button to delete the specific budget.

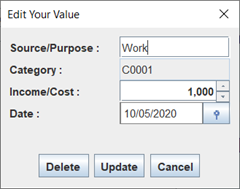


### User – Update budget

Step 1: Double click the budget you want to update.

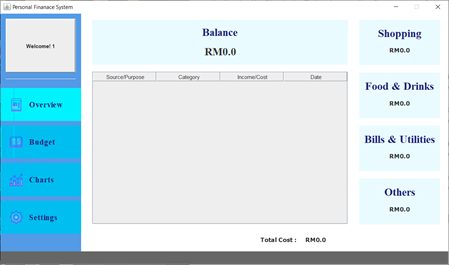


Step 2: After some modification. Press the Update button to update the specific budget.

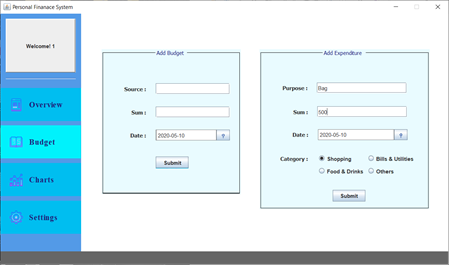


### User – Add expenditure

Step 1: Click the Budget button on the navigation bar on the left side of the main page.



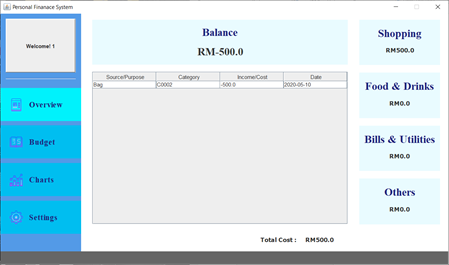
Step 2: Fill in your expenditure details.



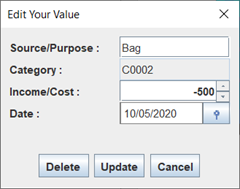
Step 3: Press the Submit button when done.

### User – Delete expenditure

Step 1: Double click the expenditure you want to delete.

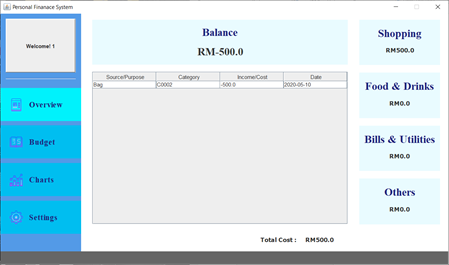


Step 2: Press the Delete button to delete the specific expenditure.

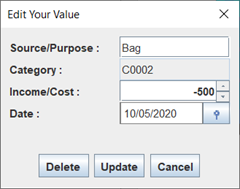


### User – Update expenditure

Step 1: Double click the expenditure you want to update.

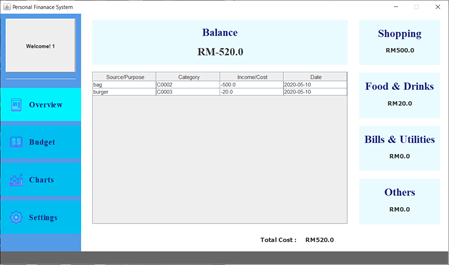


Step 2: Press the Update button to update the specific expenditure.



### User – Show graph

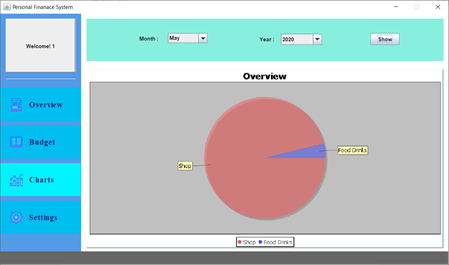
Step 1: Click the Charts button on the navigation bar on the left side of the main page.



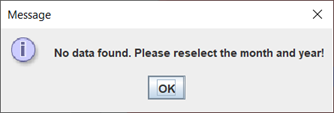
Step 2: Select the specific month and year then press the Show button.



Step 3: Pie Chart is shown.

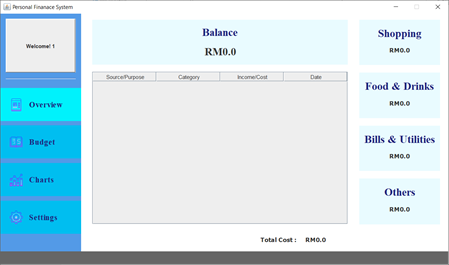


Step 4: If the specific month and year don’t have any data in it. It will prompt error.

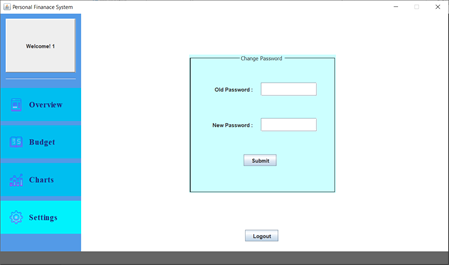


### User – Log out

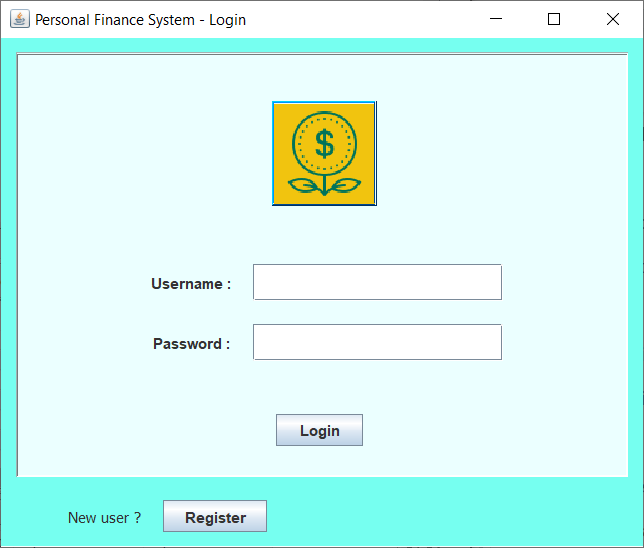
Step 1: Click the Settings button on the navigation bar on the left side of the main page.



Step 2: Press the Logout button.

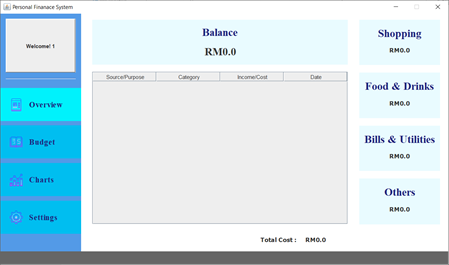


Step 3: The system returns back to the Login page.

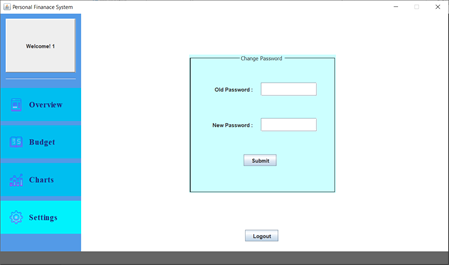


### User – Change password

Step 1: Click the Settings button on the navigation bar on the left side of the main page.



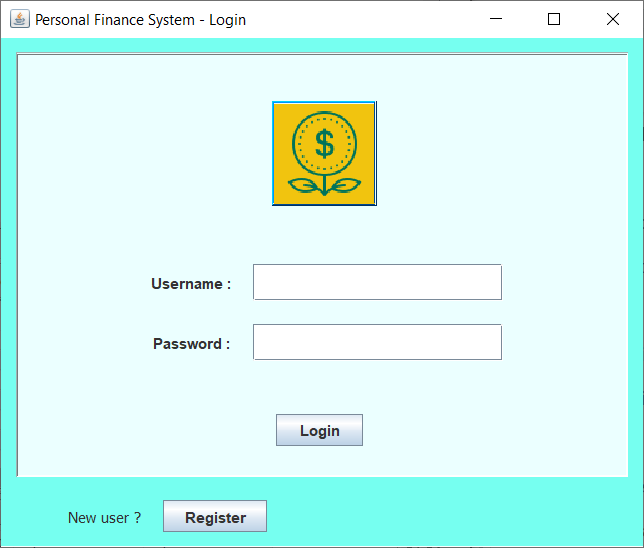
Step 2: Fill in the previous password and the new password.



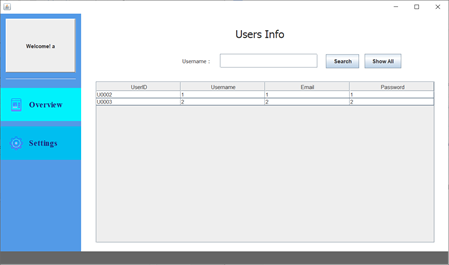
Step 3: Press the Submit button when done.

### Admin – Login

Step 1: Insert your username and password.

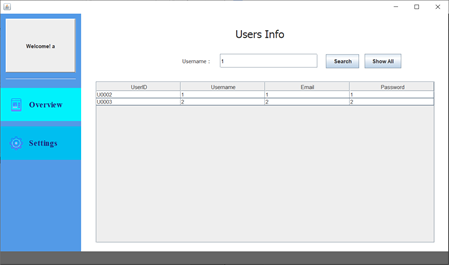


Step 2: Press the Login button and you have logged in the main page.

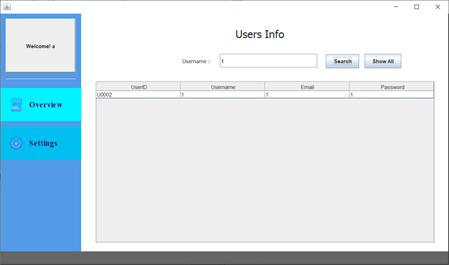


### Admin – Find specific user

Step 1: On the Username search bar, fill in the username of the specific user.

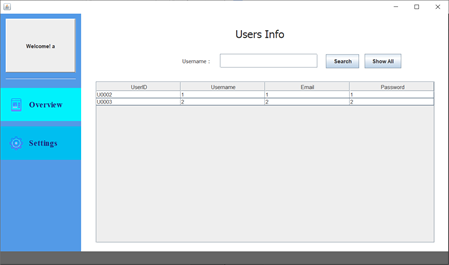


Step 2: Press the Search button to search the specific user.



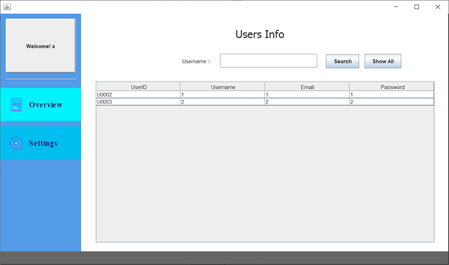
### Admin – Show all the user

Step 1: Press the Show All button to show all the users.

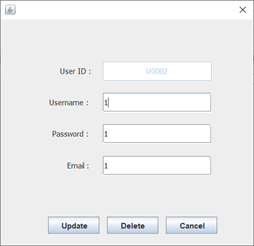


### Admin – Update user

Step 1: Double click the user you want to update.

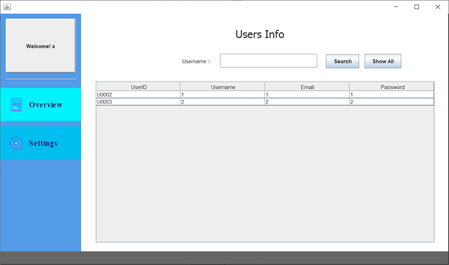


Step 2: After some modification. Press the Update button to update the specific user.

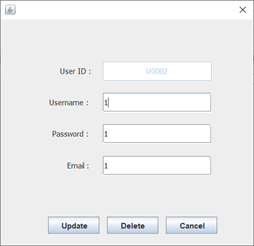


### Admin – Delete user

Step 1: Double click the user you want to delete.

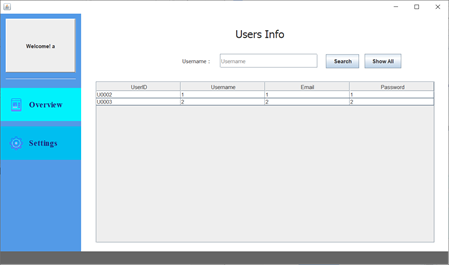


Step 2: After some modification. Press the Delete button to delete the specific user.

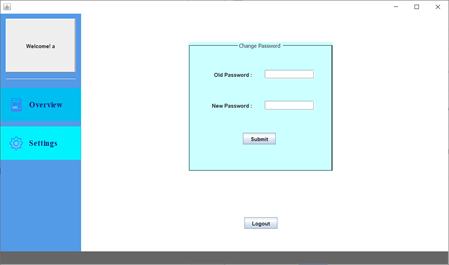


### Admin – Log out

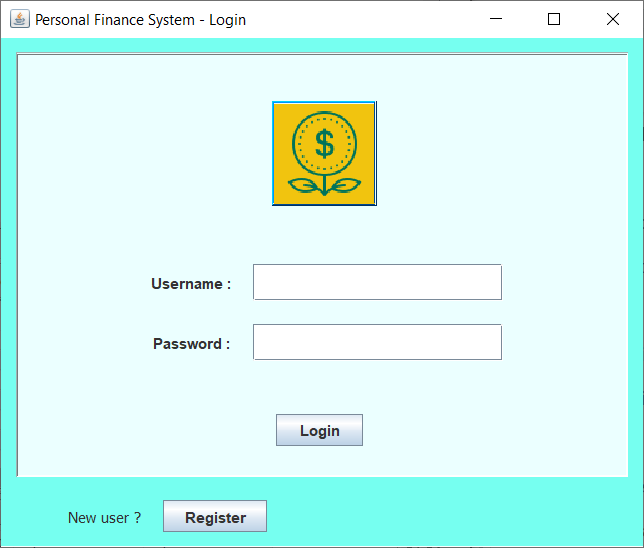
Step 1: Click the Settings button on the navigation bar on the left side of the main page.



Step 2: Press the Logout button.

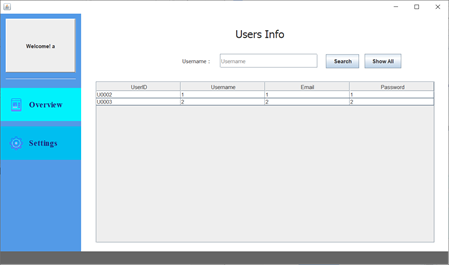


Step 3: The system returns back to the Login page.

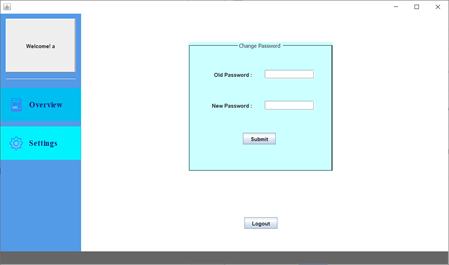


### Admin – Change password

Step 1: Click the Settings button on the navigation bar on the left side of the main page.



Step 2: Fill in the previous password and the new password.



Step 3: Press the Submit button when done.