

# DOCUMENT FOR FINAL YEAR PROJECT IN CS/CE/IS UMM AL QURA UNIVERSITY



## AE BANK

**Team name: AE GROUP**

**Team logo:**



**Team members:**

Name	Email	ID
Eyad Khaled Qbori	S438015859@st.uqu.edu.sa	438015859
Abdulkarim Al-Zahrani	S437003832@st.uqu.edu.sa	437003832
Aseel sami yamani	s437011517@st.uqu.edu.sa	437011517
Ayman Al-zahrani	S437032212@st.uqu.edu.sa	437032212

**Project supervisor**

**DR. Amir Auad AL-Zaidy**

Start date	End date
2021/8/29	2021/12/5

Credits Hrs	Project ID
4	UQU-CS-2021S -03-M

## ACKNOWLEDGMENTS

We dedicate this project to God Almighty our creator, our strong pillar, our source of inspiration, wisdom, knowledge and understanding. He has been the source of our strength throughout this project. We also dedicate this work to our families and friends, who has encouraged us all the way and whose encouragement has made sure that we give it all it takes to finish that which we have started and whom without their support this project would have never been done. We also dedicate this work to our supervisor DR. Amir Auad AL-Zaidy, he has been the first supporter who has never hesitated to help and guide us on the right path. The word thank you will not fulfill his right to what he did with us, he knows every single word we want to say. Thank you. Our love for you all can never be quantified. May Allah bless you.

### Author(s):

Name: Eyad Khaled Qbori

Signature: *Eyad*

Name: Abdulkarim Al-Zahrani

Signature: *Abdulkarim*

Name: Aseel sami yamani

Signature: *Aseel*

Name: Ayman Al-Zahrani

Signature: *Ayman*

### Supervisor(s):

Name: Dr. Amer Awad Al-zaidi

Signature: *Amir*

## Intellectual Property Right Declaration

This is to declare that the work under the supervision **Dr. Amir Al-Zaidy** having the title " **AEBank** " carried out in partial fulfillment of the requirements of Bachelor of Science in Computer Science, is the sole property of the Umm Al-Qura University and the respective supervisor and is protected under the intellectual property right laws and conventions. It can only be considered/ used for purposes like extension for further enhancement, product development, adoption for commercial or organizational usage, etc., with the permission of the University and respective supervisor.

This above statement applies to all students and faculty members.

**Date: 5/12/2021**

### **Author(s):**

Name: Eyad Khaled Qbori

Signature: *Eyad*

Name: Abdulkarim Al-Zahrani

Signature: *Abdulkarim*

Name: Aseel sami yamani

Signature: *Aseel*

Name: Ayman Al-Zahrani

Signature: *Ayman*

### **Supervisor(s):**

Name:

Dr. Amir Aoud Al-Zaydi

Signature: *Amir*

## Anti-Plagiarism Declaration

This is to declare that the above publication produced under the supervision of **Dr. Amir Aoud Al-Zaydi** the title "**AEBank**" is the sole contribution of the author(s) and no part hereof has been reproduced illegally (cut and paste) which can be considered as Plagiarism. All referenced parts have been used to argue the idea and have been cited properly. I/We will be responsible and liable for any consequence if violation of this

**Date: 5/12/2021**

**Author(s):**

Name: Eyad Khaled Qbori

Signature: *Eyad*

Name: Abdulkarim Al-Zahrani

Signature: *Abdulkarim*

Name: Aseel sami yamani

Signature: *Aseel*

Name: Ayman Al-Zahrani

Signature: *Ayman*

## **Abstract**

The open banking system is a third-party application aimed for this people in society who are interested in finance, or savings. The open banking system allows these sectors to manage their money in one software to the maximum extent possible, allowing them to handle all their bank accounts in the Kingdom of Saudi Arabia in one application. When it comes to checking account balances and making bank transfers, the open banking system helps to keep track of all movements and transactions across all accounts at the same time That is, you can check all accounts in one software, by giving the user a new environment in terms of convenience of use that eliminates old techniques that waste a lot of time and effort, such as switching between applications and signing into accounts. Personal financial control is also provided by the system.

## Table of figures

Table name	Page
Figure 2.1 System Data Flow Diagram	17
Figure 2.2 Zero Level Data Flow Diagram	18
Figure 2.3 System Use Case Diagram	19
Figure 2.4 Registration Use Case Diagram	20
Figure 2.5 Login Use Case Diagram	21
Figure 2.6 Recover Account Use Case Diagram	22
Figure 2.7 Change Password Use Case Diagram	23
Figure 2.8 Add / Edit Bank Account Use Case Diagram	24
Figure 2.9 Inquiry Balance Use Case Diagram	25
Figure 2.10 Budget Use Case Diagram	26
Figure 2.11 Login Sequence Diagram	27
Figure 2.12 Register Sequence Diagram	28
Figure 2.13 Recover Account Sequence Diagram	29
Figure 2.14 Change Password Sequence Diagram	30
Figure 2.15 Bank Account Sequence Diagram	31
Figure 2.16 Budget Sequence Diagram	32
Figure 2.17 System Activity Diagram	33
Figure 3.1 Waterfall Model Diagram	46
Figure 4.1: Class Diagram	50
Figure 4.2: Entity Relationship Diagram	51
Figure 4.3: EERD	52

## Table of contents

AEBank .....	8
Chapter #1 Introduction.....	8
1.1 Introduction .....	9
1.2 Product scope.....	9
1.3 Problem Definition .....	9
1.4 Purpose of this Document .....	9
1.5 Overview of this Document.....	10
1.6 Existing Systems .....	11
1.7 Summary .....	15
Chapter 2 Requirement Analysis.....	16
2.1 Data Modeling Diagrams .....	17
2.1.1 Data Flow Diagrams.....	17
2.1.2 Use Case Diagrams.....	19
2.1.3 Sequence Diagrams .....	27
2.1.4 Activity Diagram .....	33
2.2 System Requirements .....	33
2.2.1 Gathering requirements .....	34
2.2.2 Functional and data requirements.....	36
2.2.3 Non-functional requirements.....	40
2.3 Proposed Solutions .....	41
Chapter 3 Design Considerations .....	42
3.1 Design Constraints 3.1.1 Hardware environment:.....	42
3.2 Architectural Strategies .....	44
3.3 Future enhancements.....	45
Chapter 4 System Design .....	46
4.1 System Architecture and Program Flow.....	46
4.1.1 Major modules.....	46
4.2.1 Class diagram .....	47
4.2.2 Data base .....	48
4.2.3 Interface description .....	خطأ! الإشارة المرجعية غير معرفة.

# **AEBank**

## **Chapter #1 Introduction**

### **Contents**

#### **1.1 Introduction**

#### **1.2 Purpose of the Project**

#### **1.3 Problem Definition**

#### **1.4 Purpose of this Document**

#### **1.5 Overview of this Document**

#### **1.6 Existing Systems**

#### **1.7 Summary**



## **1.1 Introduction**

The open banking system is an application that aims to help people who are interested in finance and savings. It allows them to manage their money in one place. By giving them the ability to track all of their bank accounts movements across all accounts at the same time, so they could check all of their accounts in one place, this eliminates the need for them to travel to different banks and manage their accounts in Saudi Arabia.

## **1.2 Product scope**

This document is intended for providing an overview of open banking system and general overview of the entire project. The scope of the document:

- Seeing and checking all your credit's cards accounts in one place.
- Tracking your spending habits by giving a spending analysis.
- You'll get a notification to tell you how far you're from your budget goal.

## **1.3 Problem Definition**

Any one of us wants to manage all his bank accounts from one application, track our spending habits by giving a spending analysis.

## **1.4 Purpose of this Document**

This document gives details of project activities including:

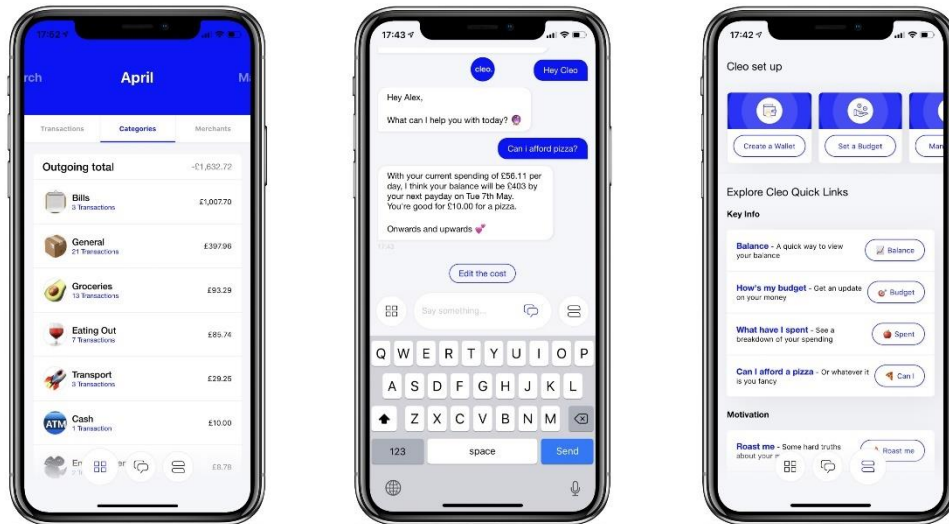
- Requirements of the system.
- Highlight several related works.
- Analysis and design of the system.
- Proposed solution that makes all person accounts in one place.

## **1.5 Overview of this Document**

There are four chapters in this report. Chapter 1 is an introduction that describes the purpose of our project. System analysis and data flow diagrams are given in Chapter 2. Chapter 3 contains project design considerations including requirements of the system and project management details. System design is mentioned in chapter 4

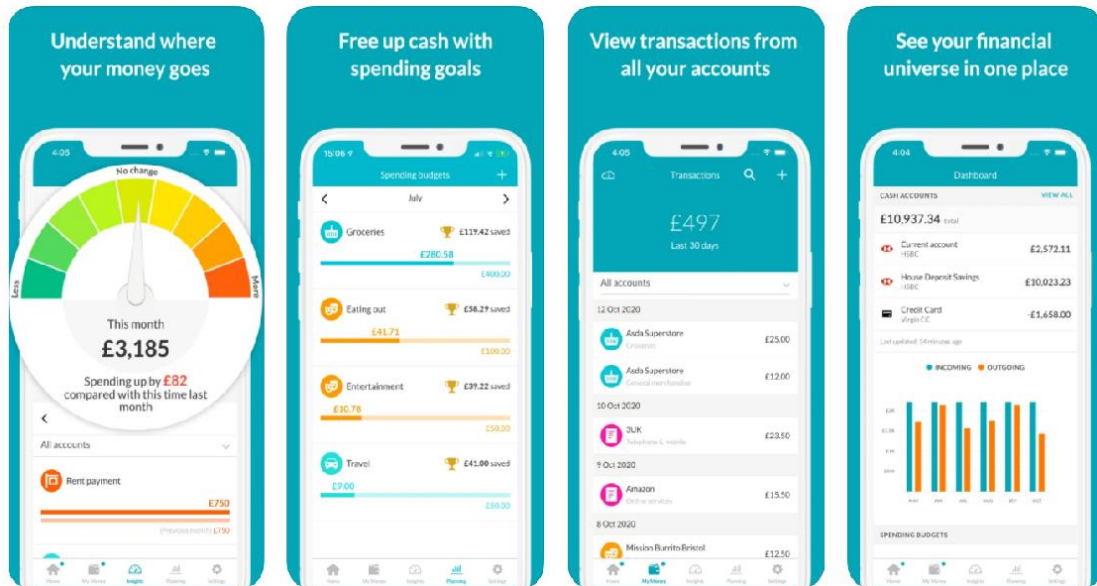
## 1.6 Existing Systems

### 1.6.1 Cleo [1]



Cleo Integration Cloud (CIC) is a cloud-based ecosystem integration platform that brings together end-to-end integration visibility across API, EDI, and non-EDI integrations. With CIC, technical and business users gain the confidence to rapidly onboard trading partners, enable integration between applications, and accelerate revenue-generating business processes. Organizations have the choice of self-service, managed services, or a blended approach – ensuring complete flexibility and control over their B2B integration strategy.

## 1.6.2 Moneyhub [2]



Want a breakdown of the main aspects of Money hub? Here's a few of the features that you'll find in the app to help you on your financial journey:

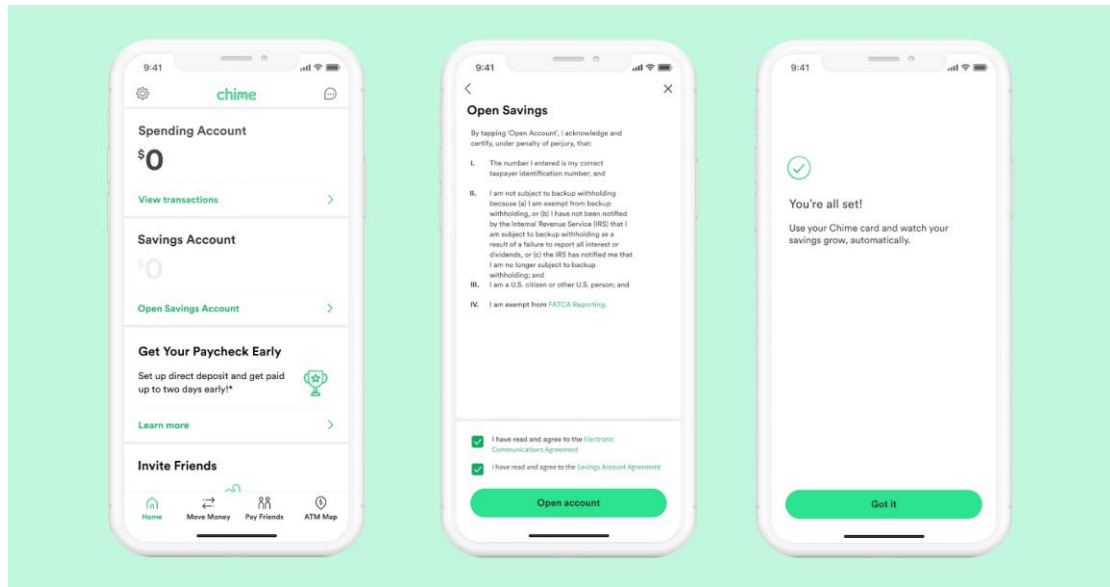
Accounts and assets - All your finances in one place

Through the 'Accounts and assets' page you can connect and view any number of accounts you might have, so that you can see all of your finances in one place.

We currently connect to: current accounts; credit cards; savings accounts; mortgages; pensions; and investments.

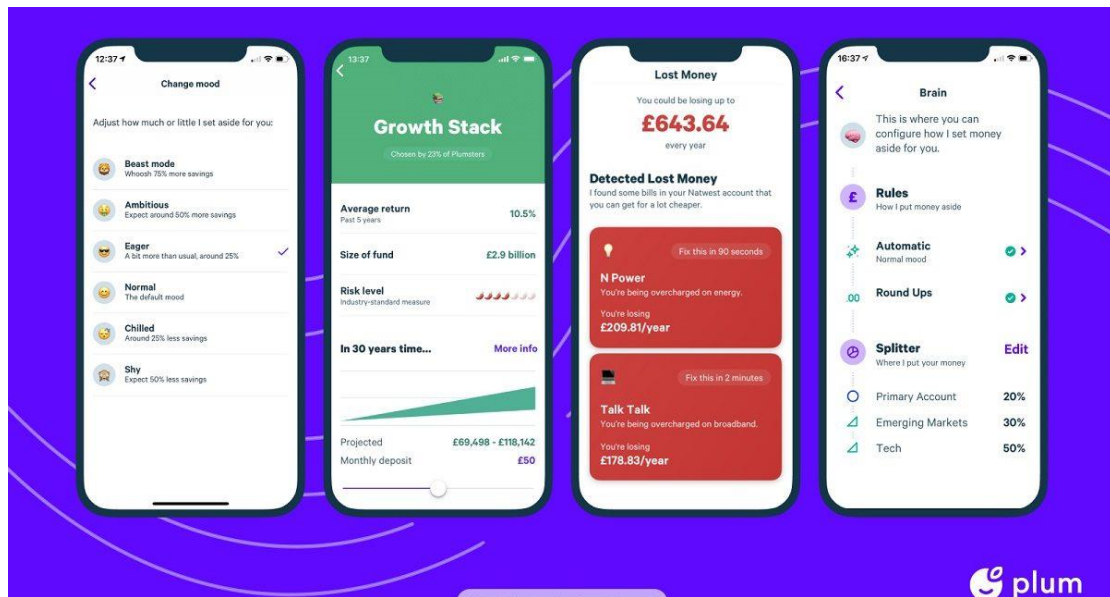
You can also manually add assets that contribute to your net worth, but might not necessarily be part of an online banking institution.

### 1.6.3 Chime[3]



We created Chime because we believe everyone deserves financial peace of mind. We're building a new kind of online bank account that helps members get ahead by making managing money easy. It's your money. It's your life. Chime in.

## 1.6.4 Plum[4]



Plum connects to your current account and analyses your incomings and outgoings. It analyses transactions and then identifies your regular income, rent, bills and daily spend. Using this and other factors like your available balance it calculates daily what amount it can safely put aside without affecting your daily life and moves it to your Plum account via direct debit every 4-5 days.

## 1.7 Summary

In this chapter we provided overview of the project. We give introduction of the proposed solution. We explained objectives and motivation of the project. We give brief details about each chapter.

### References

- [1]<https://www.cleo.com>
- [2]<https://www.moneyhub.com>
- [3]<https://www.chime.com>
- [4]<https://withplum.com>

# **Chapter 2 Requirement Analysis**

## **Contents**

### **2.1 Data Modeling Diagrams**

#### **2.1.1 Data Flow Diagrams**

#### **2.1.2 Use Case Diagrams**

#### **2.1.3 Sequence Diagrams**

#### **2.1.4 Activity Diagram**

### **2.2 System Requirements**

#### **2.2.1 Gathering requirements**

#### **2.2.2 Functional and Data Requirements**

#### **2.2.3 Non-functional Requirements**

##### **2.2.3.1 Look and Feel Requirements**

##### **2.2.3.2 Usability Requirements**

##### **2.2.3.2 Reliability Requirements**

##### **2.2.3.2 Availability Requirements**

##### **2.2.3.5 Security Requirements**

##### **2.2.3.4 Performance Requirement**

##### **2.2.3.5 Portability Requirements**

### **2.3 Proposed Solutions**



## 2.1 Data Modeling Diagrams

### 2.1.1 Data Flow Diagrams

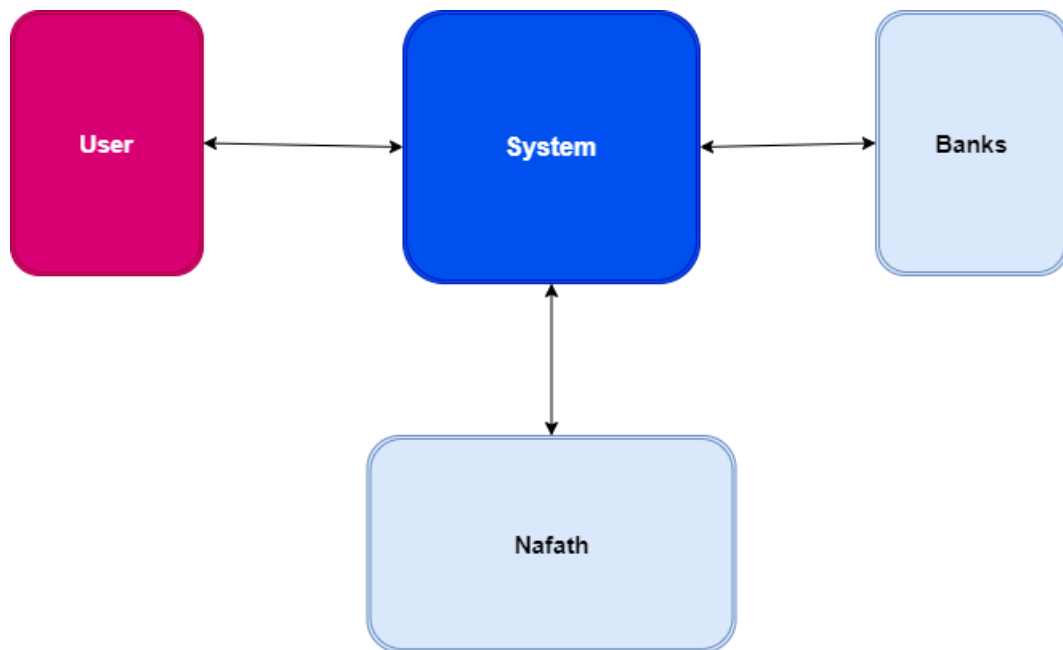


Figure 2.1 System Data Flow Diagram

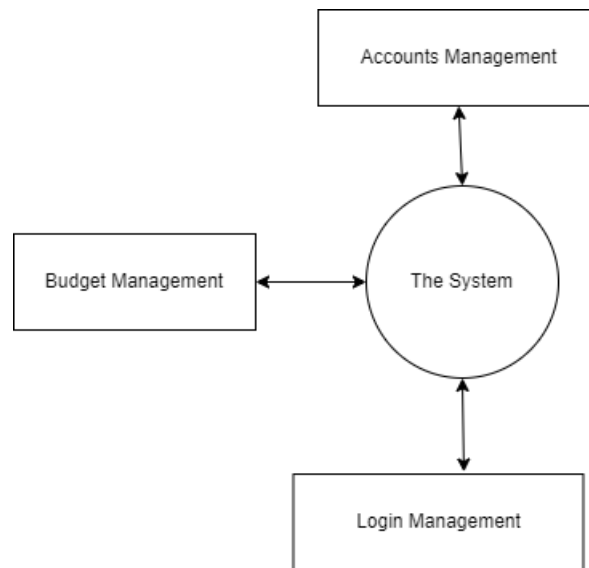


Figure 2.2 Zero Level Data Flow Diagram

**Description:**

A registered user can supply his email and password to login the system, if the user's credentials are correct, he will be redirected to his dashboard page else an error message will be displayed for him.

A user can add a new bank account or edit an existing one after supplying the required information about account .

A user can add or edit budget by adding a new budget item or edit an existing one.

## 2.1.2 Use Case Diagrams

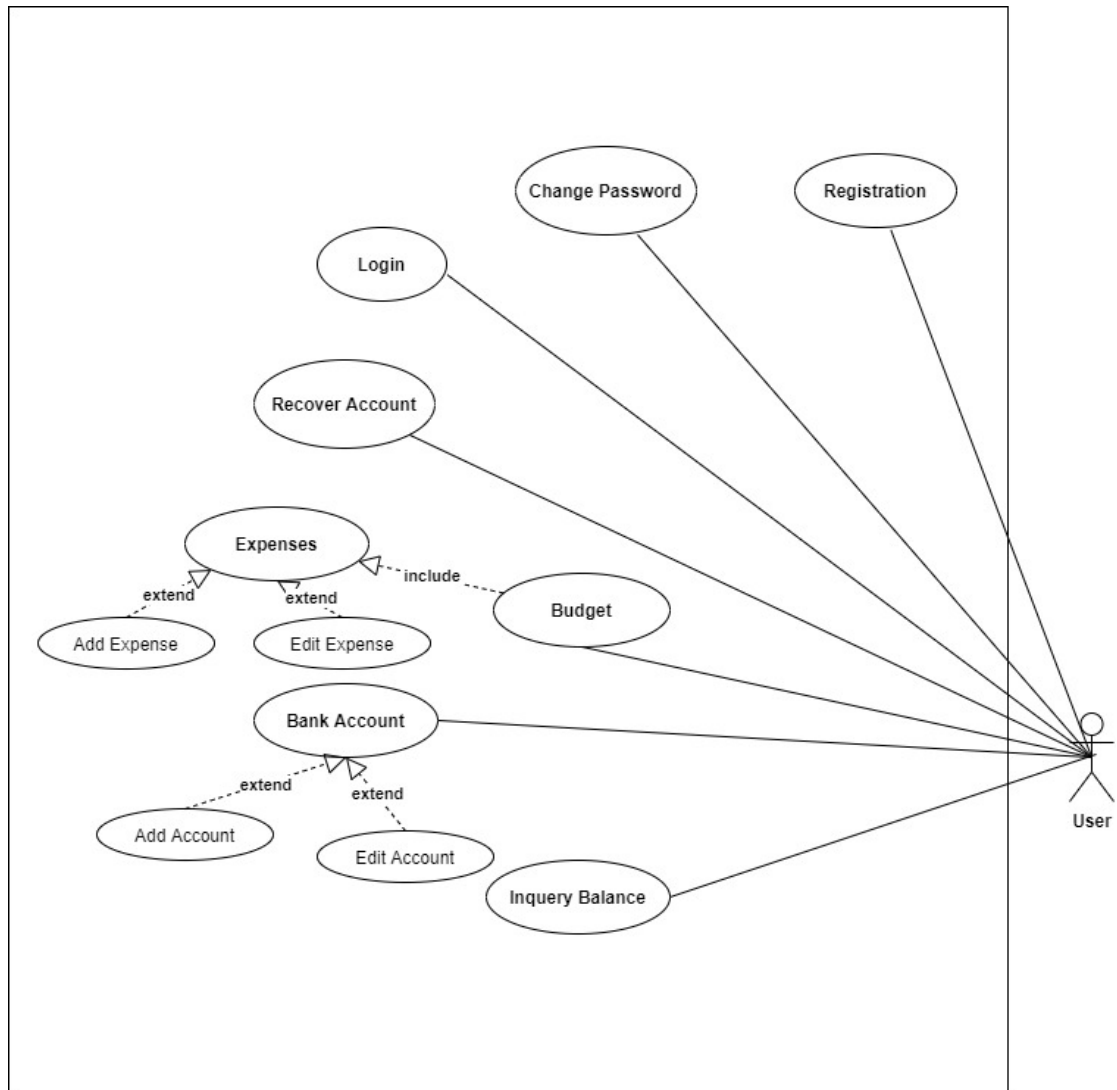


Figure 2.3 System Use Case Diagram

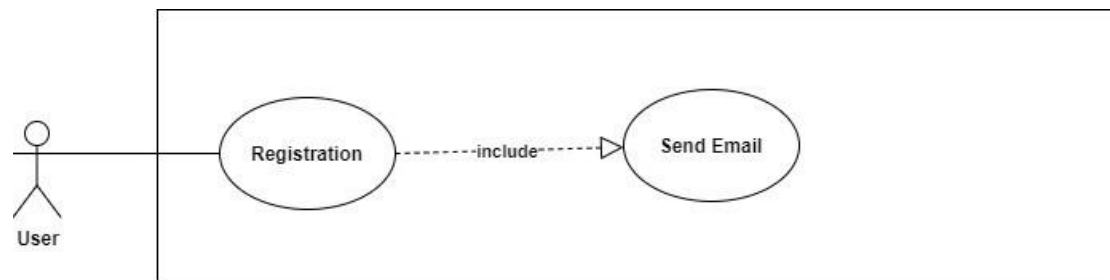


Figure 2.4 Registration Use Case Diagram

<b>Use case name</b>	Registration
<b>Unique</b>	Email, ID
<b>Actor(s)</b>	user
<b>Description</b>	The user should open the application to get to registration screen, Then the system will ask the user to enter his information: ("First name", "Last name", "Email", "Password", "ID").
<b>Steps Performed (main Path)</b>	1- The user will click on registration button 2- The system will ask the user to fill his information 3- The user should enter his information 4- The user will click on registration button 5- The system will take the user to "Nafath Gate" and sent an email to the user to validate his information 6- The user will his national identification. 7- The system takes to user to the main dashboard.
<b>Alternative Path</b>	In step 1,3 and 7 the user can cancel and exit to the home page
<b>Exception Points</b>	In step 4 if the user inputs wrong credentials a pop-up message will show on the screen says what the issue the user put.
<b>Preconditions</b>	1-The user must have internet connection. 2-The user must have a valid registered email.
<b>Post Condition</b>	The user will be notified that he was sent an activation email link.

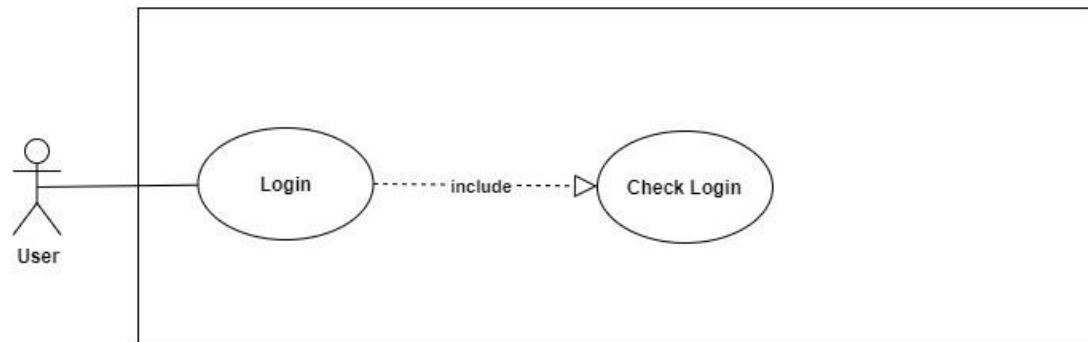


Figure 2.5 Login Use Case Diagram

<b>Use case name</b>	Login
<b>Unique</b>	email
<b>Actor(s)</b>	user
<b>Description</b>	The user chooses sign in from the menu, the system will sign in the user into the system.
<b>Steps Performed (main Path)</b>	1- The system asks the user to enter his information 2- The user enters his email and password 3- The system checks if the user already registered 4- The user choose sign in button to login 5- The system shall sign in the user to the dashboard
<b>Alternative Path</b>	In step 2,4 the user can cancel and exit to the menu
<b>Exception Points</b>	If the user enters input wrong credentials, the system will pop-up an error message to the user.
<b>Preconditions</b>	1-The user must have internet connection. 2-The user must have a valid registered email.
<b>Post Condition</b>	The user will be redirect to his dashboard.

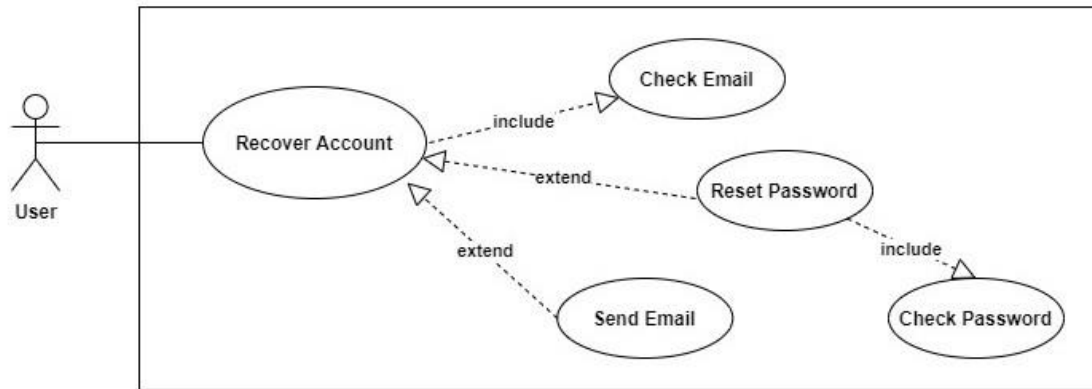


Figure 2.6 Recover Account Use Case Diagram

<b>Use case name</b>	Recover Account
<b>Unique</b>	Email
<b>Actor(s)</b>	user
<b>Description</b>	The user could recover his account password by this feature.
<b>Steps Performed (main Path)</b>	<ol style="list-style-type: none"> <li>1- The user will open the application and click on login button</li> <li>2- The system would give an option to choose, says "Forget your password?"</li> <li>3- The user will click on "Forget your password?" option in the login page then he must supply his email and press on "Reset my account" button.</li> <li>4- The user will open his email and click on link in the email the application sent.</li> <li>5- The system will redirect the user to put the new password with the code sent to him.</li> <li>6- The user will supply the new password.</li> </ol>
<b>Alternative Path</b>	In step 1,3 and 6 the user can cancel or exit the system.
<b>Exception Points</b>	In step 3 if his email is not in our database an error message will be displayed.
<b>Preconditions</b>	<ol style="list-style-type: none"> <li>1-The user must have internet connection.</li> <li>2-The user must have a valid registered email.</li> </ol>
<b>Post Condition</b>	The user will be notified that an email sent to him.

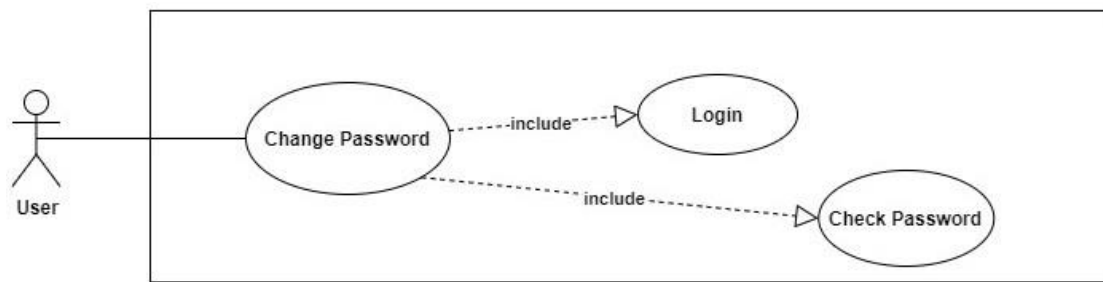


Figure 2.7 Change Password Use Case Diagram

<b>Use case name</b>	Change Password
<b>Actor(s)</b>	user
<b>Description</b>	Enables user to change his password.
<b>Steps Performed (main Path)</b>	<ol style="list-style-type: none"> <li>1- The user will sign in into the system.</li> <li>2- The system will show the dashboard screen.</li> <li>3- Then the user would choose "Profile" tab then choose "Account" and click on "Change password" tab.</li> <li>4- The system asks the user to put the current password, new password and the password conformation.</li> <li>5- The user will supply information and click on submit button.</li> </ol>
<b>Alternative Path</b>	In step 1,3 and 5 the user could cancel or exit from the system.
<b>Exception Points</b>	<p>In step 1, if the user supplies a wrong credentials the system would pop-up an error message to the user.</p> <p>In step 5, if the user supplies same old password an error message will show on the screen.</p> <p>In step 5, if the user supplies a different new password and password conformation an error message will show on the screen.</p>
<b>Preconditions</b>	<ol style="list-style-type: none"> <li>1-The user must have internet connection.</li> <li>2-The user must be logged to the system.</li> </ol>
<b>Post Condition</b>	The user will be notified that his password changed successfully.

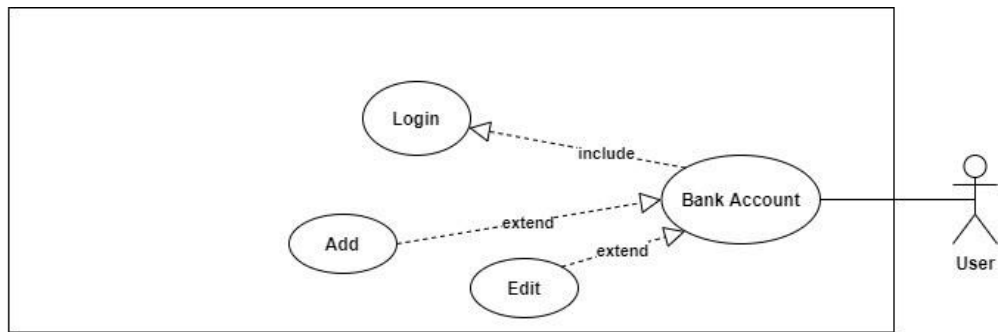


Figure 2.8 Add / Edit Bank Account Use Case Diagram

<b>Use case name</b>	Add /Edit Bank Account
<b>Actor(s)</b>	user
<b>Description</b>	Enables a user add a new bank account or edit an existing one.
<b>Steps Performed (main Path)</b>	1- The user would sign in into the application 2- The system would take the user to dashboard 3- The user will click on "Accounts" button 4- The system will show two options ("Add" & "Edit") 5- The user could choose between add or edit account 6- The system will show to the user the result.
<b>Alternative Path</b>	In step 1,3 and 5 the user can cancel or exit from the system.
<b>Exception Points</b>	In step 1 and 5 if the user supplies a wrong credentials the system would pop-up an error message to the user.
<b>Preconditions</b>	1-The user must have internet connection. 2-The user must be logged to the system.
<b>Post Condition</b>	The user will be notified that the account will be added or edited successfully.



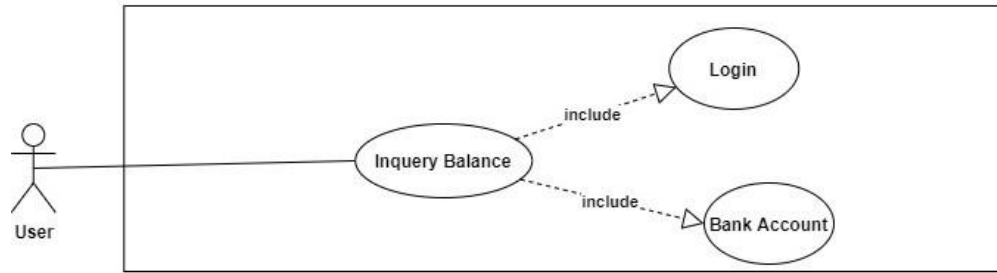


Figure 2.9 Inquiry Balance Use Case Diagram

<b>Use case name</b>	Inquiry Balance
<b>Actor(s)</b>	user
<b>Description</b>	Enables a user from inquiry his total balance from all his accounts
<b>Steps Performed (main Path)</b>	1- The user would sign in into the application 2- The system would take the user to dashboard 3- The user will choose "Balances" from the dashboard 4- The system will give the user the results
<b>Alternative Path</b>	None
<b>Exception Points</b>	None
<b>Preconditions</b>	1-The user must have internet connection. 2-The user must be logged to the system.
<b>Post Condition</b>	The user balance will be shown.

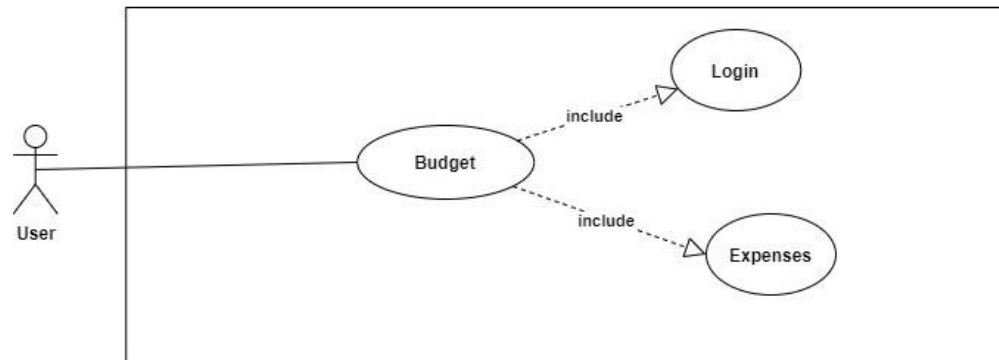


Figure 2.10 Budget Use Case Diagram

<b>Use case name</b>	Budget
<b>Actor(s)</b>	User
<b>Description</b>	The user can add a new budget item or editing an existing one.
<b>Steps Performed (main Path)</b>	1- The user would sign in into the application 2- The system would take the user to dashboard screen 3- The user will choose budget button from the main screen 4- The system will show a budget screen to the user 5- The user will add a new budget or edit an existing one. 6- The system will pop-up a success message to the user.
<b>Alternative Path</b>	In step 1,3 and 5 the user can cancel or exit from the system.
<b>Exception Points</b>	None
<b>Preconditions</b>	1-The user must have internet connection. 2-The user must be logged to the system.
<b>Post Condition</b>	The user will be notified that his operation is successfully done.

### 2.1.3 Sequence Diagrams

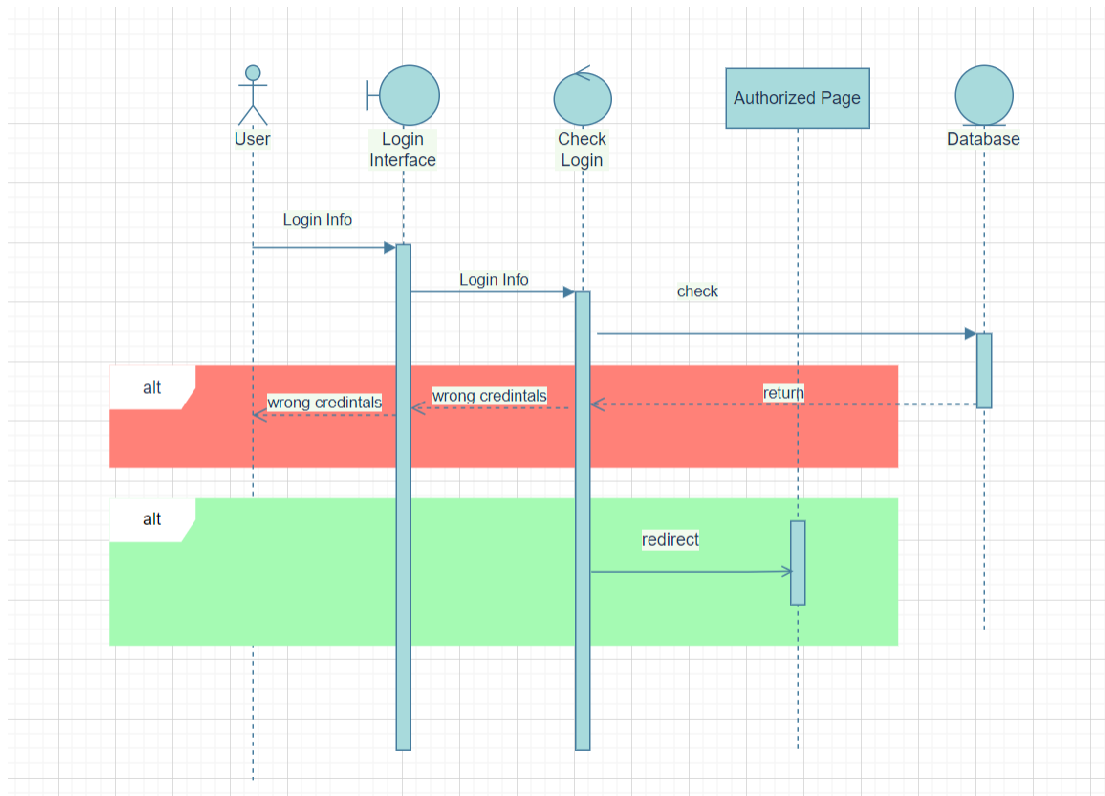


Figure 2.11 Login Sequence Diagram

#### Description:

A user can go to login in page to log in the system, he provides his email and password if his credentials are correct, he will be logged into the system else an error message will be displayed to him.

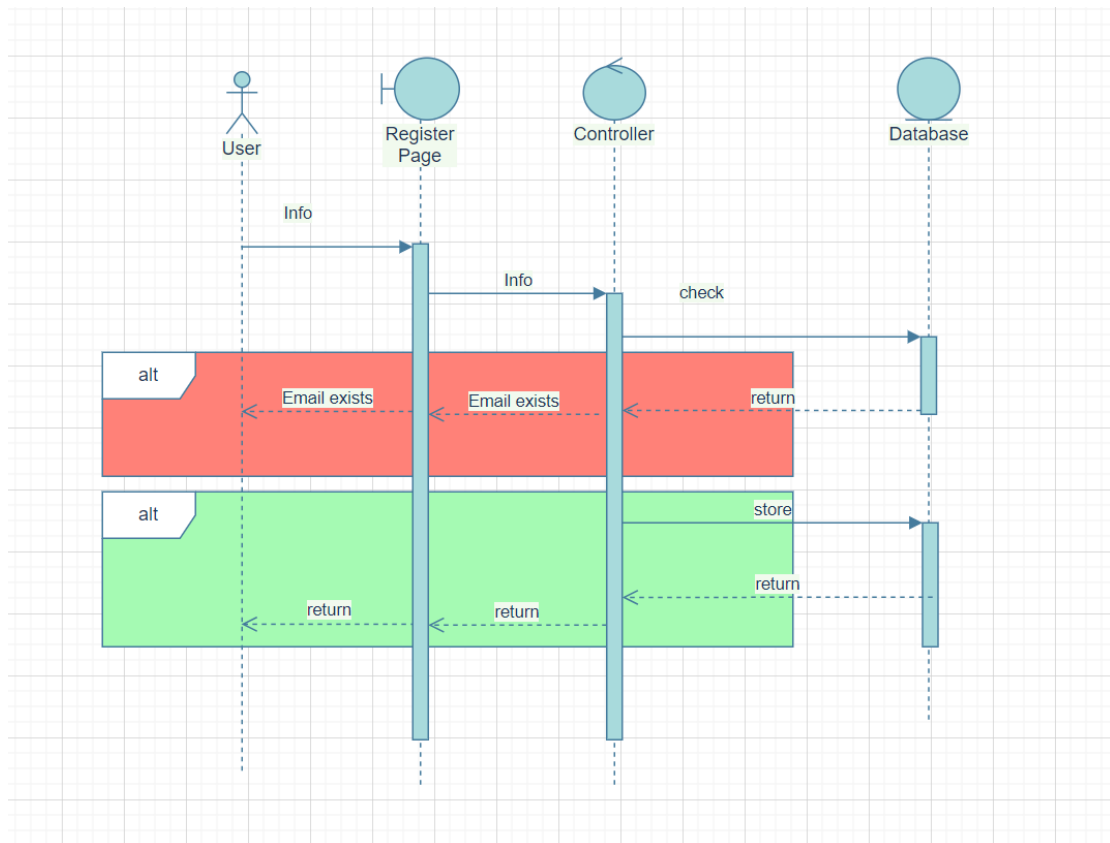


Figure 2.12 Register Sequence Diagram

### Description:

A user can create an account in the system by going to register page and provide the user information, then if the email is existing, an error message will be displayed to him otherwise he will be redirected to login page.

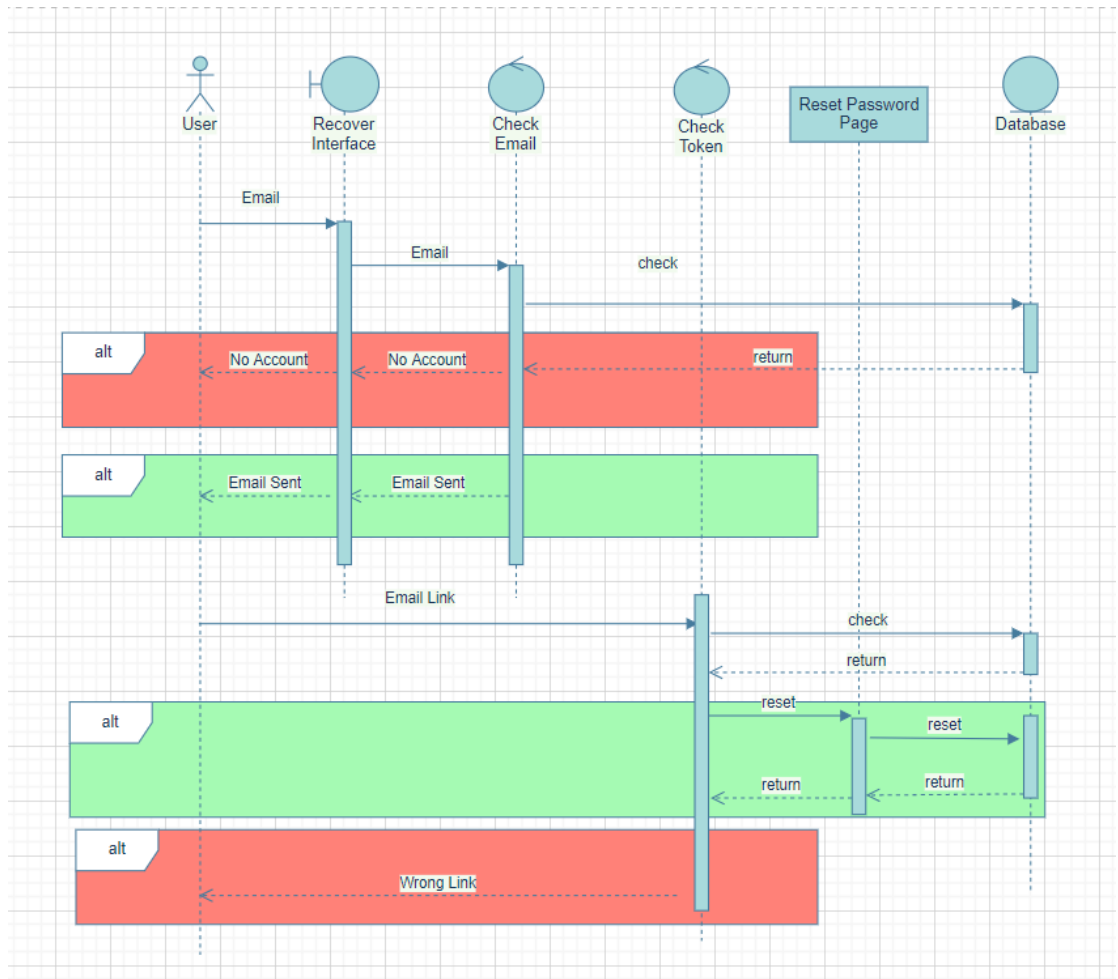


Figure 2.13 Recover Account Sequence Diagram

**Description:**

A user can go to recover page to recover his account, he provides his email and if email is not found he will be reported but if his email exists then a security link will be sent to his email and redirect him to reset password page. After pressing on the security link from email he can provide the new password for his account.

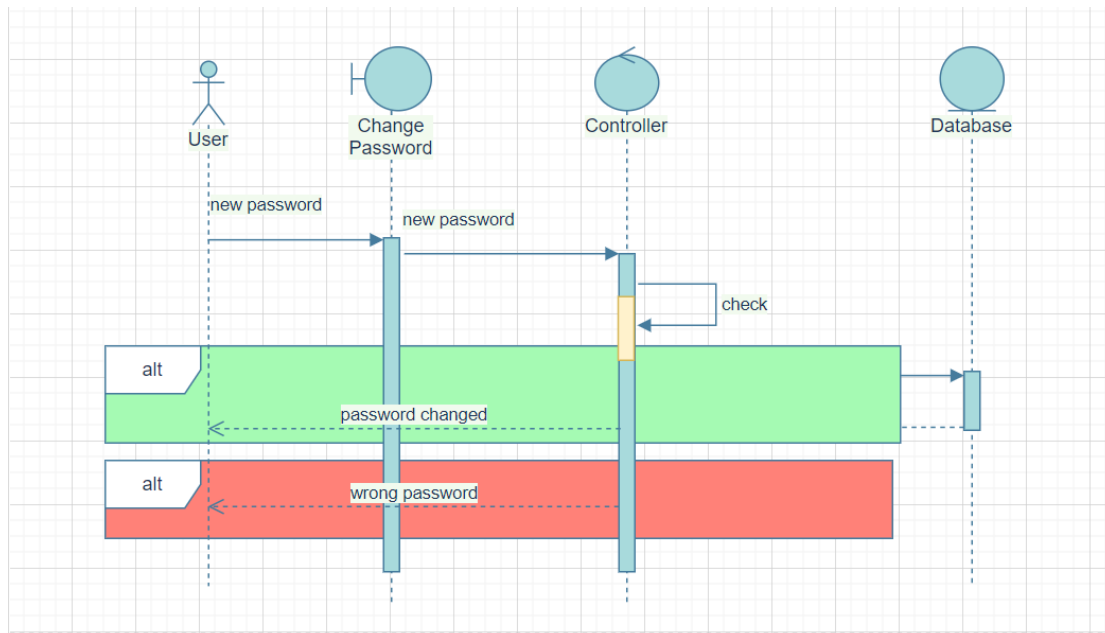


Figure 2.14 Change Password Sequence Diagram

**Description:**

A user can go to change password page to change his password after a successful login to the system. He provides the new password then click on submit button. The user will be informed about the operation result.

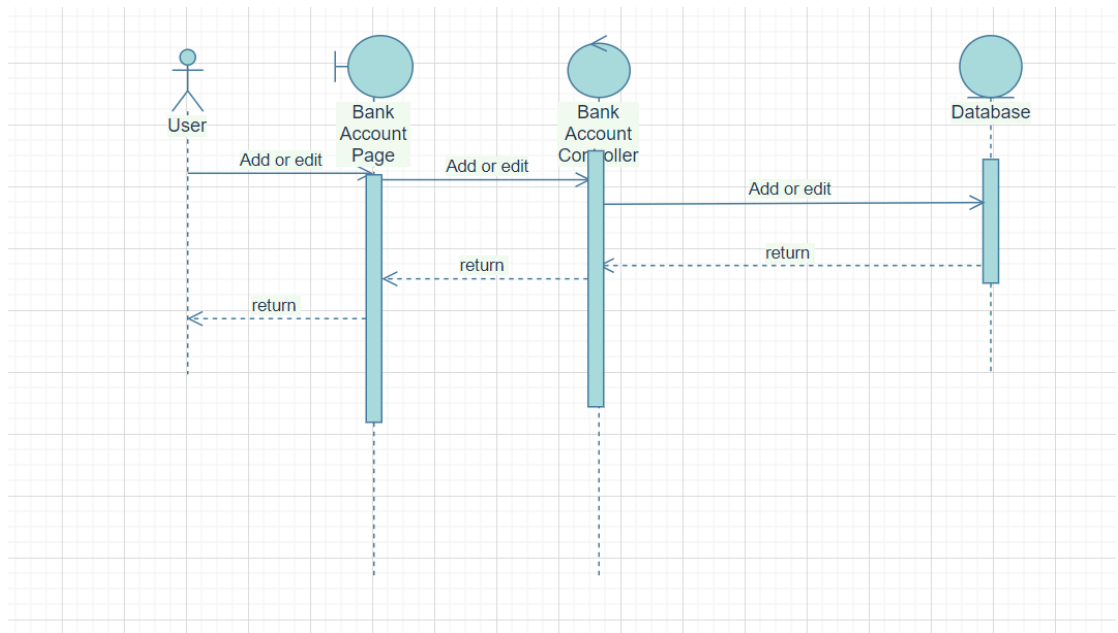


Figure 2.15 Bank Account Sequence Diagram

**Description:**

A user can go to banks page and to add or edit an existing bank , after filling required information, then press save button then he will redirected to banks page or error message will be shown if an error happened.

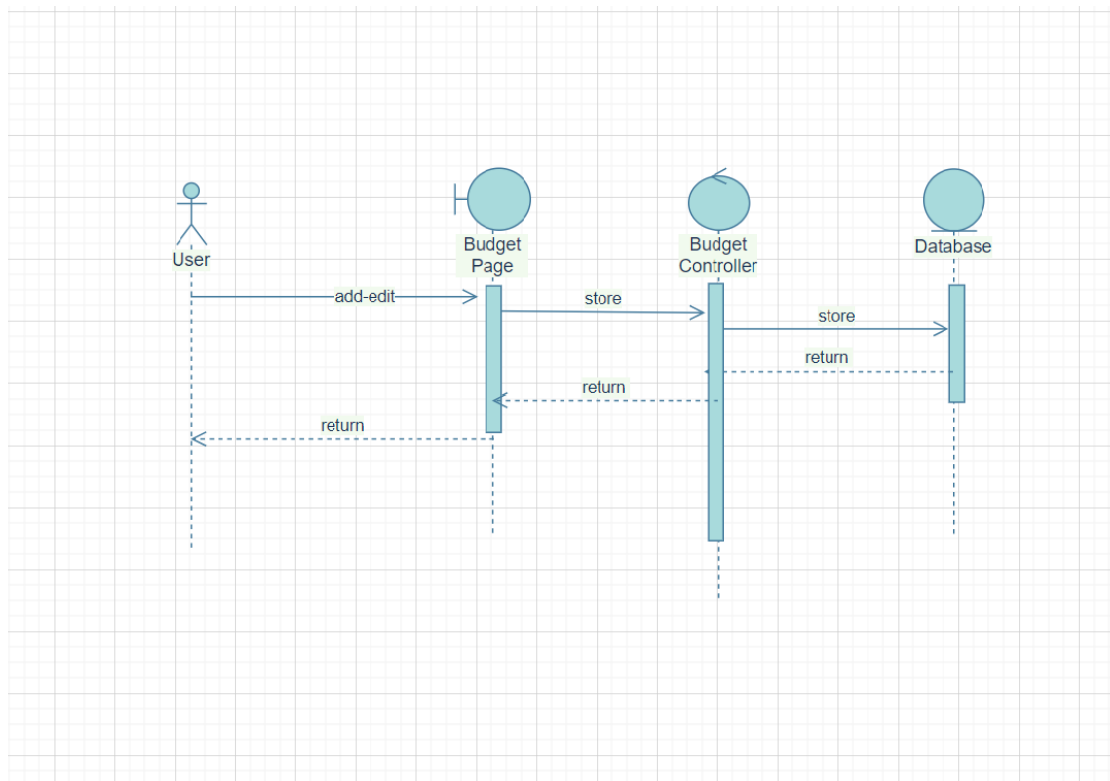


Figure 2.16 Budget Sequence Diagram

**Description:**

A user can go to budget page to add a new budget item or to edit an existing one. Then filling in the required information then press save button then he will be redirected to budget page or error message will be shown if an error happened.



### 2.1.4 Activity Diagram

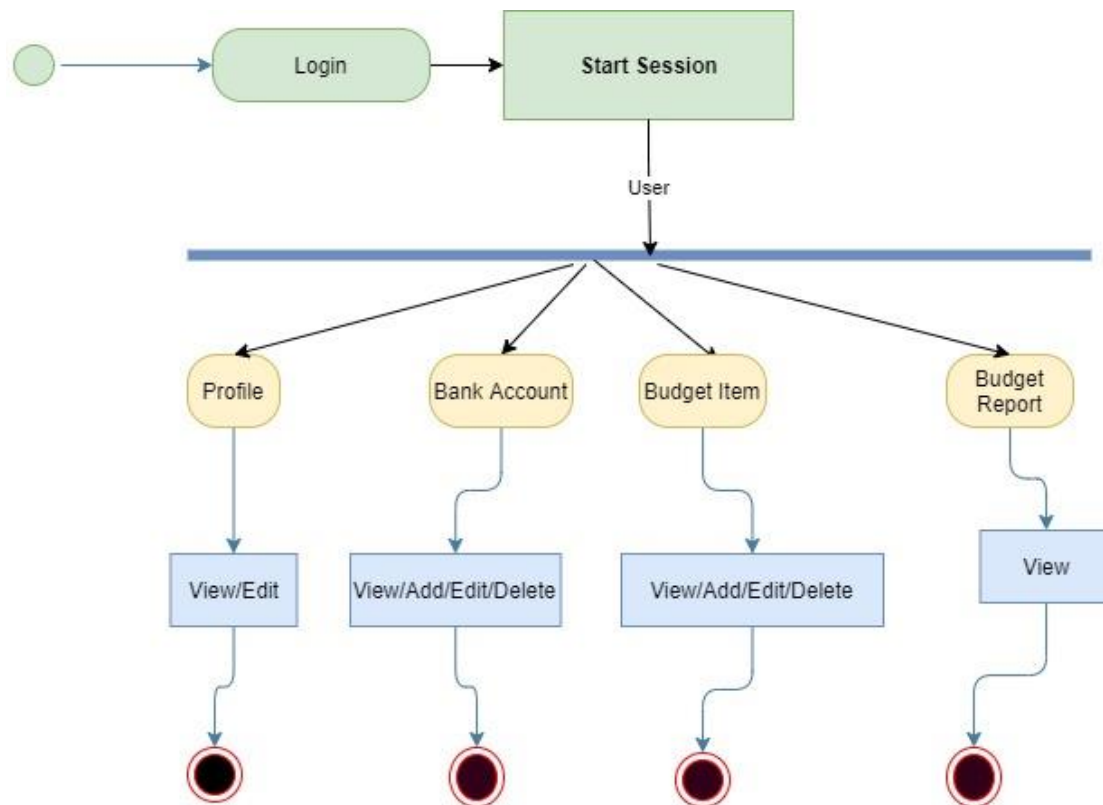


Figure 2.17 System Activity Diagram

#### Description:

A session starts after successful login, actions will be available as described below.

- Can view his profile and change his personal information.
- can view, add or edit bank account.
- can view, add or edit budget item.
- Can view budget report.

## 2.2 System Requirements

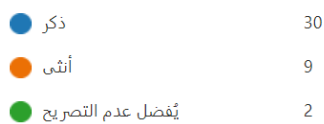
System analysis is the most important phase the purpose of this stage is analyzing all functionalities of system. The aim of this phase is taking a deep look to the system.

## 2.2.1 Gathering requirements

Gathering of the requirements provides a foundation for the success of any project, so when the system analysis process is fairly performed, it gives the correct path regards to the main system features, helps to complete the next stages of the project successfully with minimum errors, ensures a better understanding of client requirements and ensures that the effort of the team is directed perfectly to achieve the proposed requirements, this is the questionnaire that we made so on it we make the functional requirements:

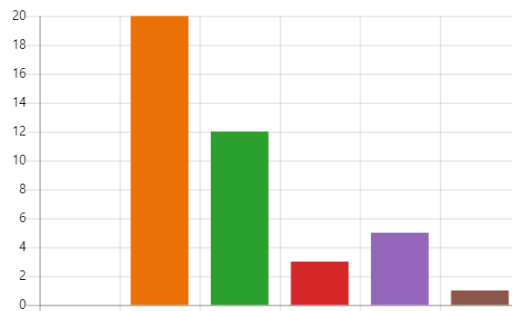
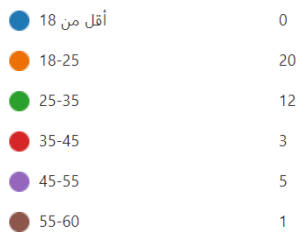
### 1. نوع الجنس

[More Details](#)



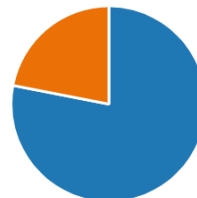
### 2. الفئة العمرية

[More Details](#)



### 3. هل تمتلك أكثر من حساب بنكي ؟

[More Details](#)



#### 4. كيف تفضل ادارة حساباتك البنكية

[More Details](#)

زيارة الفرع	1
عن طريق برنامج او موقع البنك	40



#### 5. هل تواجه صعوبة في ادارة حساباتك البنكية ؟

[More Details](#)

[Insights](#)

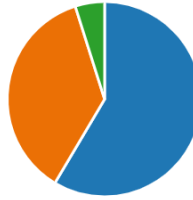
نعم	8
لا	35



#### 6. كيف ترى جودة الخدمات الالكترونية المقدمة من البنوك من حيث : (سهولة الاستخدام , توفير الوقت , السرية و الامان) ؟

[More Details](#)

ممتازة	24
مقبولة	15
أقل من المتوقع	2



#### 7. هل تهتم بالخطط الادخارية ؟

[More Details](#)

نعم	23
لا	5
ربما	13



#### 8. ما معدل استخدامك للتطبيقات البنكية ؟

[More Details](#)

يوميًا	26
أحيانًا	15
نادرًا	0
لا أستخدم التطبيقات البنكية	0



#### 9. ماذا تستخدم عادة من خدمات في التطبيقات البنكية ؟

[More Details](#)

الاستعلامات	16
الحوالات	39
خدمات سداد	26



10. ما المشاكل التي تواجهك عادة في التطبيقات البنكية ؟

[More Details](#)

43

Responses

Latest Responses

...حوليل للاشخاص من دون تفعيل المستفيد و عمولة التحويل من بنك الا اخر"

"صعوبة الاستخدام"

...الخدمات عن طريق البنك او اجهزة الخدمة الذاتية غير متاحة في التطبيقات"

## 2.2.2 Functional and data requirements

### 2.2.2.1 Register

Requirement ID:	R01.0	Use case:	Register
Description:	<p>The system shall allow the user to register in the system, by filling the requested information.</p> <p>-if the user leaves an empty filed, the system shall pop-up an error message to the user.</p> <p>-if the user supplies an already registered email, the system shall pop-up an error message to the user.</p> <p>- If the user supplies a wrong credentials, the system shall pop-up an error message to the user.</p>		
Rationale:	Without sign up into the system, you can't access the system and you can't use any Features		
Inputs:	First name, Last name, Email, Password, ID		
Outputs;	A conformation message		
Persistent change:	A new account will be added to the system		
Related requirement:	None		

### 2.2.2.2 Login

Requirement ID:	R02.0	Use case:	Login
Description:	<p>The system shall allow the user to login in the system by checking his email and his password if he already registered in the system, and his credentials are identical.</p> <p>-if the user leaves an empty field, the system shall pop-up an error message to the user.</p> <p>- If the user supplies a wrong credentials, the system shall pop-up an error message to the user.</p>		
Rationale:	Without login into the system, you can't access the system and you can't use any Features		
Inputs:	Email, password		
Outputs;	Show a main dashboard		
Persistent change:	The system will assess the user to join his profile.		
Related requirement:	R01.0		

### 2.2.2.3 Change Password

Requirement ID:	R03.0	Use case:	Change Password
Description:	<p>The system shall allow the user to change his password after he logged into the system.</p> <p>-if the user leaves an empty field for his request, the system shall decline his request and show an error message.</p> <p>-if the user doesn't provide a suitable password the system will show an error message will show on the screen.</p>		
Rationale:	The user wants to change his password.		
Inputs:	Current password, new password, password conformation		
Outputs;	Conformation message.		
Persistent change:	The user's password will be changed		
Related requirement:	R02.0		

### 2.2.2.4 Recover Account

Requirement ID:	R04.0	Use case:	Recover Account
Description:	<p>The system shall allow the user to recover his account if he forgot his password.</p> <p>-if the user leaves an empty field for his request, the system shall decline his request and show an error message.</p> <p>-if the system doesn't find the user email in the database the system an error message will show on the screen.</p>		
Rationale:	The user could recover his account by this feature.		
Inputs:	Email		
Outputs;	Confirmation message.		
Persistent change:	The account will be accessible to the user after recovering the account.		
Related requirement:	R01.0, R02.0		

### 2.2.2.5 Update Profile

Requirement ID:	R05.0	Use case:	Update Profile
Description:	<p>The system shall allow the user to update his profile.</p> <p>-if the user leaves an empty field for his request, the system shall decline his request and show an error message.</p>		
Rationale:	The user wants to change his profile information		
Inputs:	Profile information		
Outputs;	Display the success message.		
Persistent change:	The user profile will be updated		
Related requirement:	R02.0		

### 2.2.2.6 Bank Account

Requirement ID:	R06.0	Use case:	Bank Account
Description:	<p>The system shall allow the user to add a new bank account or edit an existing one.</p> <p>-if the user leaves an empty field, the system shall pop-up an error message to the user.</p> <p>- If the user supplies a wrong credentials, the system shall pop-up an error message to the user.</p> <p>-if the user supplies an already registered bank account, an error message will be shown.</p>		
Rationale:	The user wants to add or edit a bank account.		
Inputs:	Bank account details.		
Outputs;	Display the success message.		
Persistent change:	A new bank account added or edited in the system		
Related requirement:	R02.0		

### 2.2.2.7 Budget item

Requirement ID:	R07.0	Use case:	Budget item
Description:	<p>The system shall allow the user to add a new budget item or edit an existing one.</p> <p>-if the user leaves an empty field for his request, the system shall decline his request and show an error message.</p> <p>-if the user supplies a wrong percentage in the expenses, an error message will show on the screen.</p>		
Rationale:	The user wants to add or edit a Budget item.		
Inputs:	Budget item details.		
Outputs;	Display the success message.		
Persistent change:	Adding new item will affect the whole budget.		
Related requirement:	R02.0		

### 2.2.2.8 Budget Report

Requirement ID:	R08.0	Use case:	Budget report
Description:	The system shall display report for budget items for the user.		
Rationale:	The user wants to display report of his budget.		
Inputs:	None		
Outputs;	Display report about budget items.		
Persistent change:	None		
Related requirement:	R02.0		

## 2.2.3 Non-functional requirements

### 2.2.3.1 Look and Feel Requirements

In our application we will use Neilson usability heuristic, this includes:

- 1- Visibility of system status.
- 2- User control and freedom.
- 3- Consistency and standards.
- 4- Error prevention.
- 5- Flexibility and efficiency of use.
- 6- Help and documentation.
- 7- Recognize, diagnose, and recover from errors.
- 8- Aesthetic and minimalist design.
- 9- Match between system and the real world.
- 10- Recognition rather than recall.

### 2.2.3.2 Usability Requirements

Our application will be designed to ensure that it is easy to use product. We will provide representative icons and make sure there is no ambiguity in system tasks.



### **2.2.3.3 Reliability Requirements**

Our application will be designed for work on as long as required hardware and software is provided. During the test process, any possible hardware and software combinations will be tested.

### **2.2.3.4 Availability Requirements**

Our application will be available all the time, except for backing up data process or server maintenance time, the system will be available any time as long as required hardware and software is provided.

### **2.2.3.5 Security Requirements**

All possible hacker attacks such as SQL injection or Cross Site Scripting will not be available.

### **2.2.3.6 Maintainability Requirements**

Our application will be easily maintainable if it has high-quality code that is readable and well-documented, so we will keep good coding practices in mind while our software is still in development.

### **2.2.3.7 Portability Requirements**

Our application is intended to be multi-platform such as Android and IOS but at first it will be released in Android.

## **2.3 Proposed Solutions**

Our project aims are:

- The user can view all banks account available in one place.
- The user can monitor his expenses.
- The application is intended to be multi-platform application so it will run on Android and IOS devices.

## Chapter 3 Design Considerations

### Contents

#### 3.1 Design Constraints

##### 3.1.1 Hardware environment

##### 3.1.2 Software environment

##### 3.1.3 Reuse of existing software components

#### 3.2 Architectural Strategies

##### 3.2.1 Project management strategies

##### 3.2.2 Development method

#### 3.3 Future enhancements/plans

### 3.1 Design Constraints

#### 3.1.1 Hardware environment:

Our system is a mobile application, so an internet connection is required. Our application is supposed to run on any device with android system.

#### 3.1.2 Software environment:

##### Operating System:

Operating system is a special type of software, which manage the hardware and other resources.

Our mobile application is a hybrid application so it is supposed to run on IOS and android operating systems. But at first, we will build it to run android system version  $\geq 8$ .

##### Database system:

**MySQL [1]:** MySQL is the world's most popular open source database. With its proven performance, reliability and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties including Facebook, Twitter, YouTube, Yahoo! and many more.

Oracle drives MySQL innovation, delivering new capabilities to power next generation web, cloud, mobile and embedded applications.

## Other tools Software:

### **draw.io (version 15.8.3) [2]**

It a free tool for drawing all UML drawings: data flow diagram, Class diagram, Sequence diagram, use cases. etc.

### **WPS Office (version 2020) [3]**

WPS Office (an acronym for Writer, Presentation and Spreadsheets, previously known as Kingsoft Office) is an office suite for Microsoft Windows, macOS, Linux, iOS, Android, and HarmonyOS developed by Zhuhai-based Chinese software developer Kingsoft. It also comes pre-installed on Fire tablets. WPS Office is made up of three primary components: WPS Writer, WPS Presentation, and WPS Spreadsheet.

We will use it for presentation and word.

### **Visual Studio Code IDE (version 1.45.1) [4]**

Visual Studio Code is an IDE developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent code completion, snippets, and code refactoring.

We will use it for Angular.js, CSS and HTML.

## **3.1.3 Reuse of existing software components**

We will use many free open-source libraries and frameworks like:

Software
<p>AngularJS [5]</p> <p>AngularJS is a JavaScript-based open-source front-end web framework for developing single-page applications. It is maintained mainly by Google and a community of individuals and corporations. It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–view model (MVVM) architectures, along with com.</p>
<p>Cordova [6]</p> <p>Apache Cordova is a mobile application development framework originally created by Nitobi. Adobe Systems purchased Nitobi in 2011, rebranded it as PhoneGap, and later released an open source version of the software called Apache Cordova.</p>
<p>Bootstrap [7]</p> <p>Bootstrap is the most popular CSS Framework for developing responsive and mobile-first websites.</p>

## 3.2 Architectural Strategies

### 3.2.1 Project management strategies

In this part, a good management and motivation among different task in the project will be presented.

<b>Initiating</b>	<p>This phase includes discussing different ideas with the supervisor and choosing one of them.</p> <p>It also includes writing a complete project proposal and delivering it on time to keep the project on.</p>
<b>Planning</b>	<ul style="list-style-type: none"><li>● Studying the proposed idea and writing the literature review.</li><li>● Setting up the time and all activities (schedule the project).</li><li>● Writing the general system idea and compare it with the previous system.</li><li>● Studying the advantage and disadvantage of all previous systems in order to build what is missing in these systems.</li></ul>
<b>Analysis</b>	<ul style="list-style-type: none"><li>● Defining different system requirements.</li><li>● Drawing system models.</li><li>● Writing different solutions (proposed and alternative solutions) of the system problem.</li></ul>
<b>Design</b>	<ul style="list-style-type: none"><li>● Defining the software and the hardware environment. Searching for other existing components which can be used in the application.</li></ul>

## 3.2.2 Development method

### Waterfall Method [7]

Waterfall Model is a sequential model that divides software development into predefined phases. Each phase must be completed before the next phase can begin with no overlap between the phases. Each phase is designed for performing specific activity during the SDLC phase.

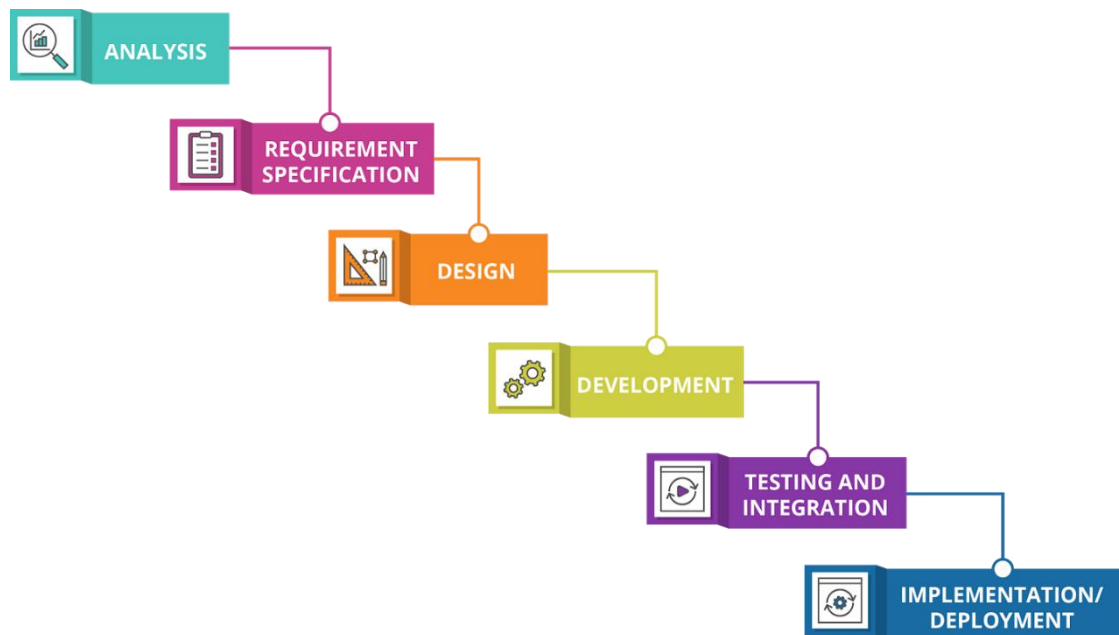


Figure 3.1 Waterfall Model Diagram

## 3.3 Future enhancements.

1-Adding more services for system.

2-Building against more platforms like windows phone.

References:

[1] <https://www.mysql.com/about/>

[2] <https://www.draw.io/index.html>

[3] <https://www.wps.com/>

[4] <https://code.visualstudio.com/>

- [5] <https://angularjs.org/>
- [6] <https://cordova.apache.org/>
- [7] [https://www.w3schools.com/whatis/whatis\\_bootstrap.asp](https://www.w3schools.com/whatis/whatis_bootstrap.asp)
- [8] <https://www.sitesbay.com/software-engineering/se-waterfall-model>

## Chapter 4 System Design

contents

4.1 System Architecture and Program Flow

4.1.1 Major modules

4.2 Detailed System Design

4.2.1 Class diagram

4.2.2 Data base

4.2.3 Interface description

### 4.1 System Architecture and Program Flow

#### 4.1.1 Major modules

The system consists of many modules like.

**Authentication:** the module responsible for user operations like login, register, etc. it has sub modules:

Register Module

Login Module

Reset Password Module

Forgot Password Module

**BankAccount:** the module responsible for handling bank accounts.

**Budget:** the module responsible for handling budget.

**BudgetItems:** the module responsible for handling budget items.

## 4.2 Detailed System Design

### 4.2.1 Class diagram

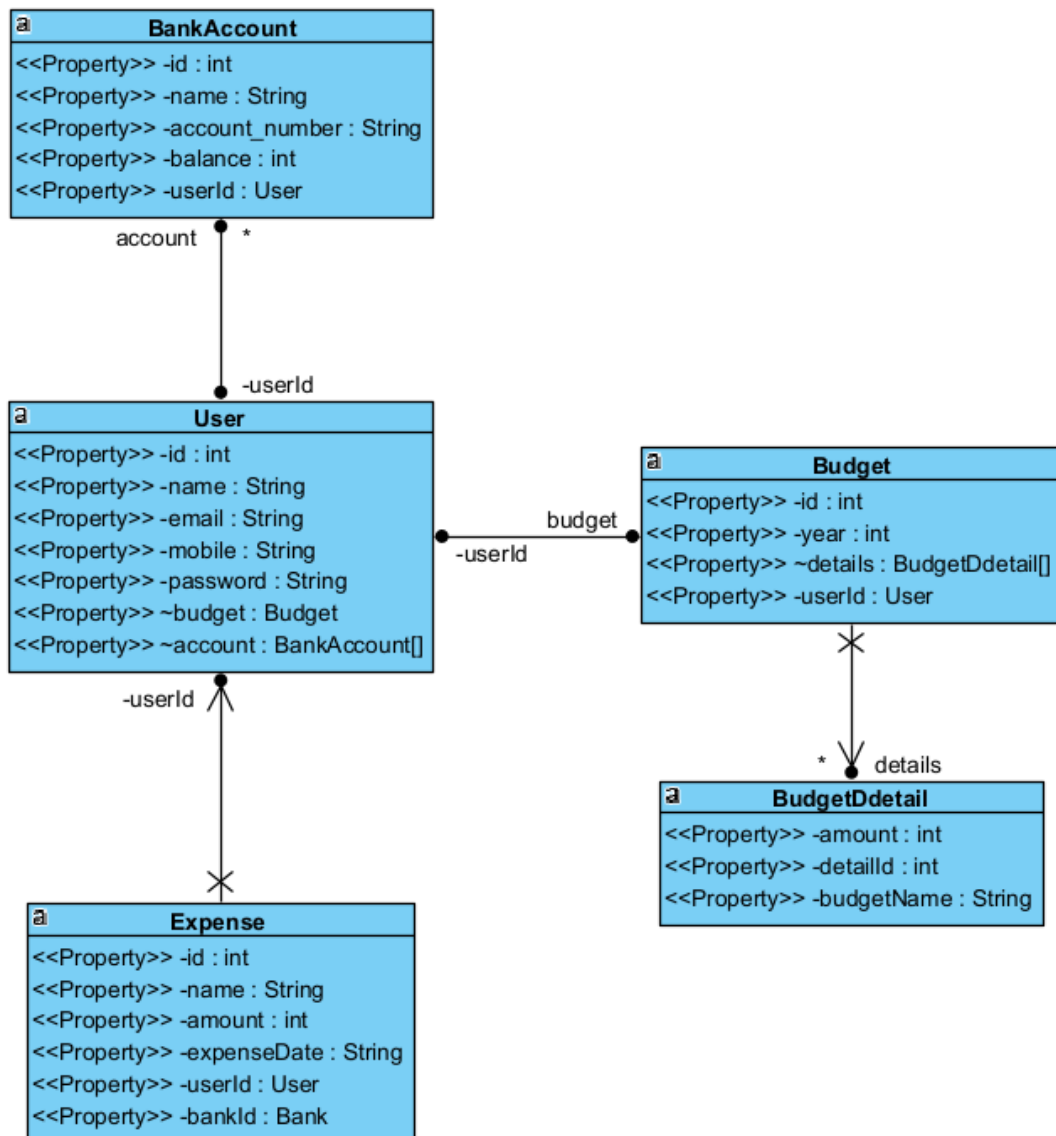


Figure 4.1: Class Diagram

## 4.2.2 Data base

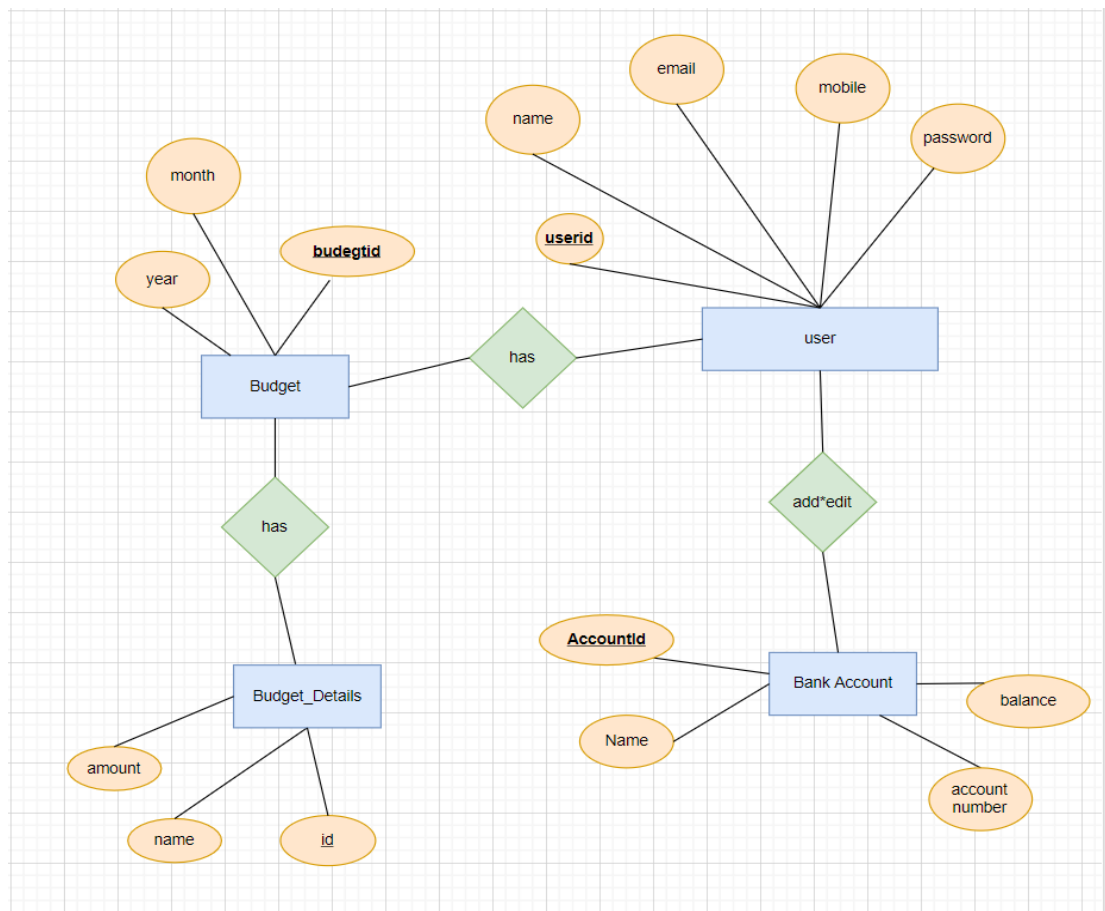


Figure 4.2: Entity Relationship Diagram



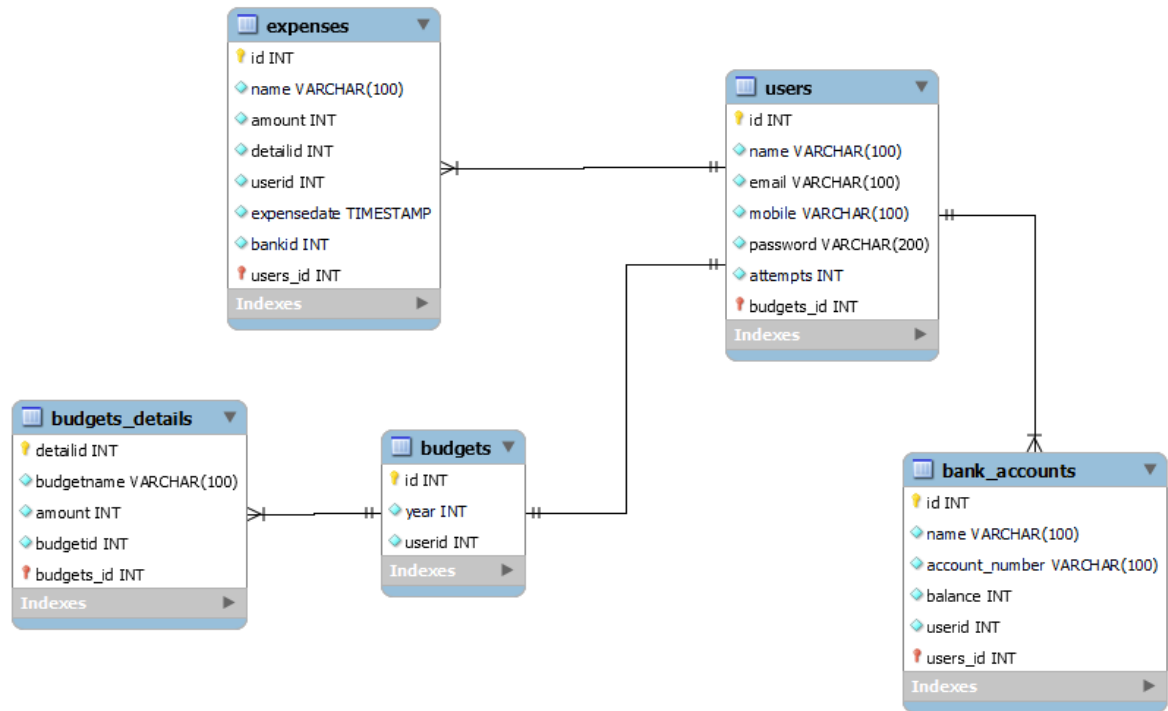


Figure 4.3: EERD t

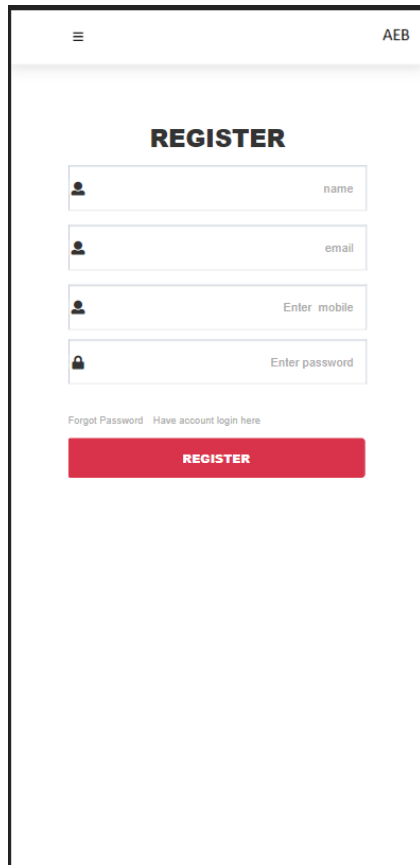
## Database Dictionary

Table Users			Table Bank_Accounts		
Field	Type	Key	Field	Type	Key
id	int	PK	id	int	PK
name	String		name	String	
email	String	unique	account_number	String	unique
mobile	String		balance	int	
password	String		userid	int	FK
Table Budget			Table Budget_Details		
Field	Type	Key	Field	Type	Key
id	int	PK	id	int	PK
month	int		name	String	
year	int		amount	int	
userid	int	FK	budgetid	int	FK

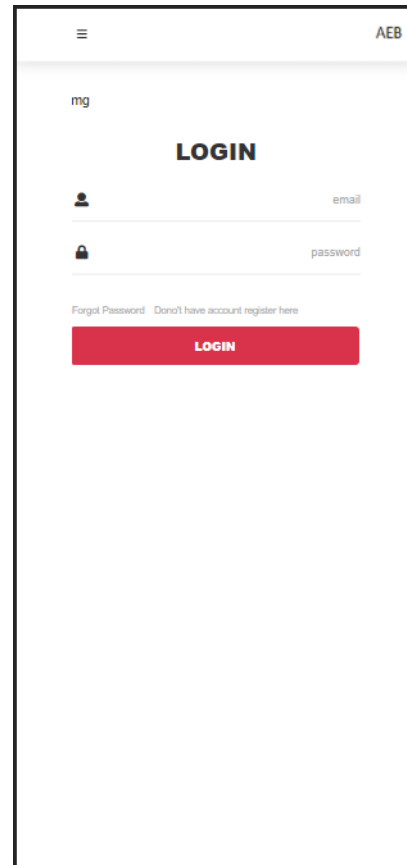
Table 4.1: Data Dictionary

### 4.2.3 Interface description

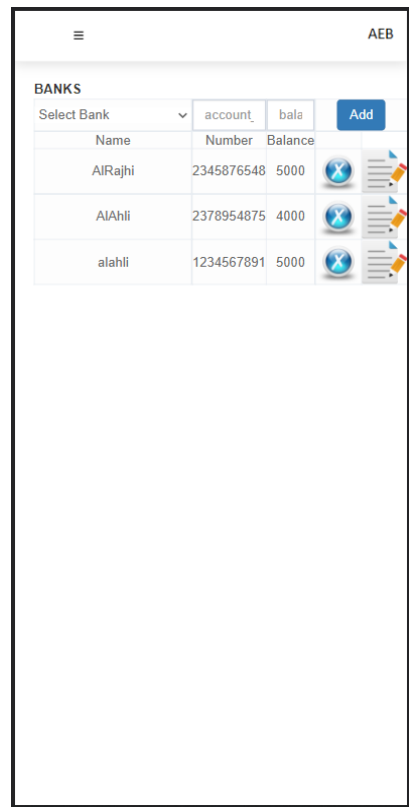
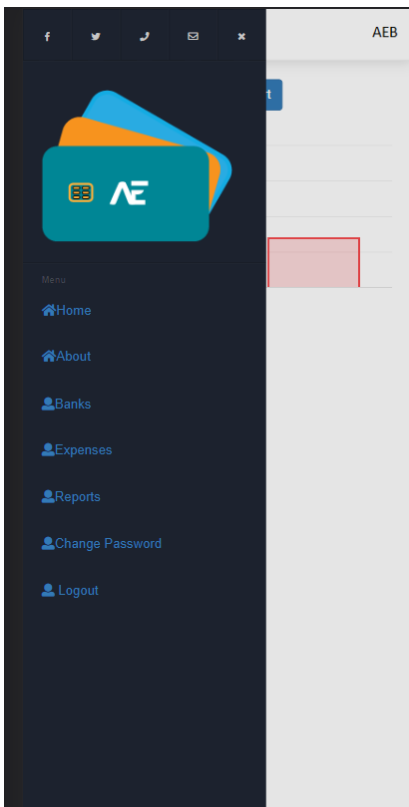
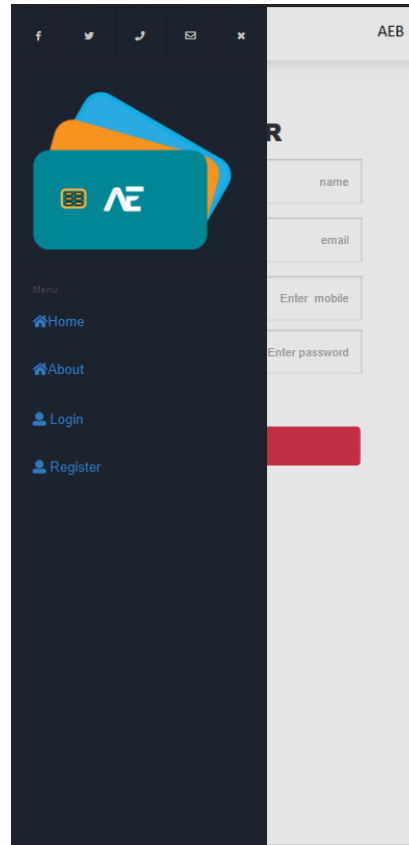
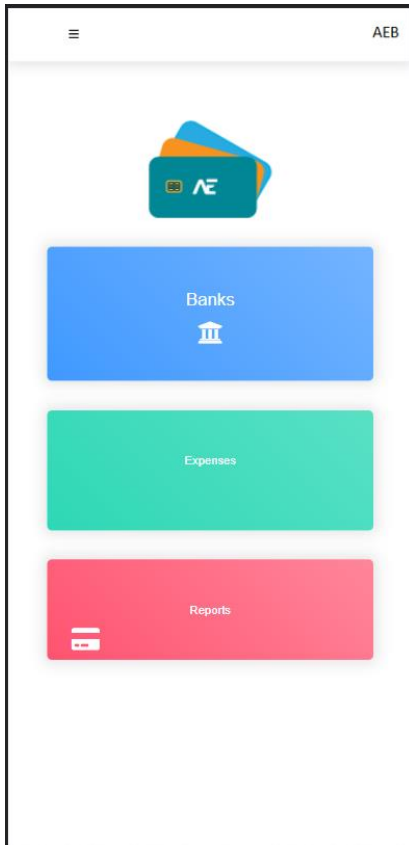
Some mobile Application interfaces (prototype):

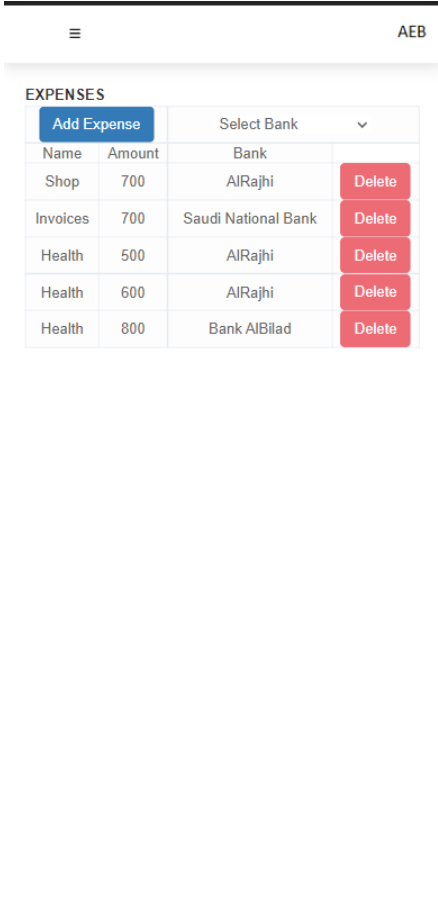
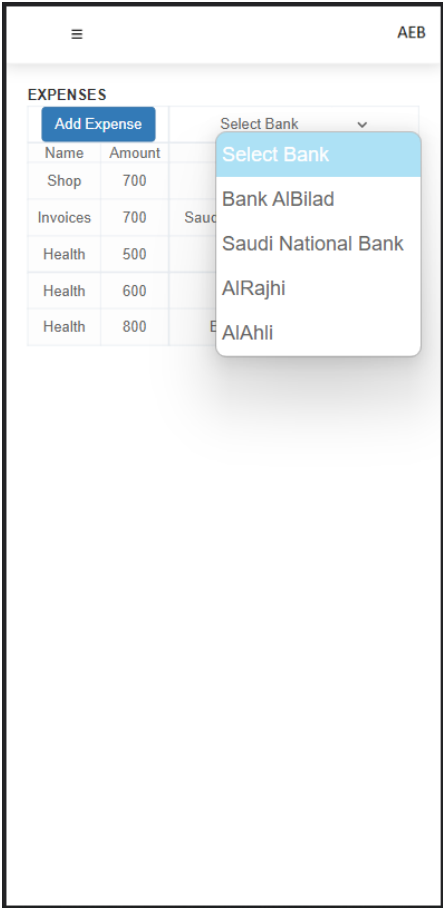
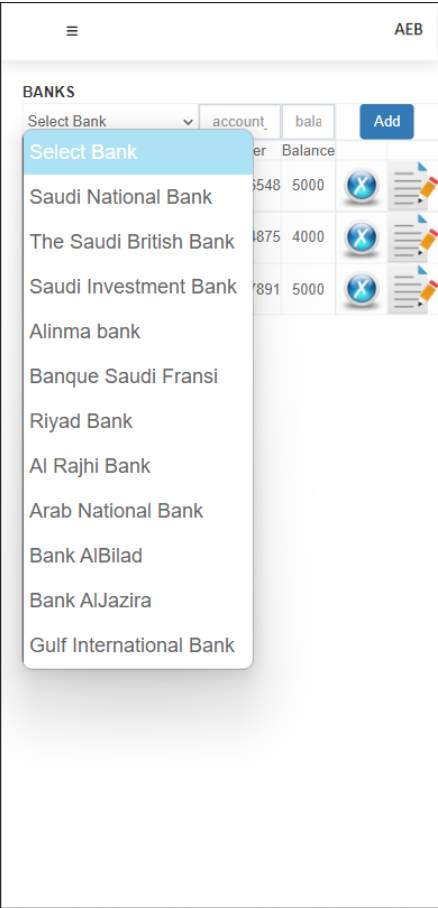


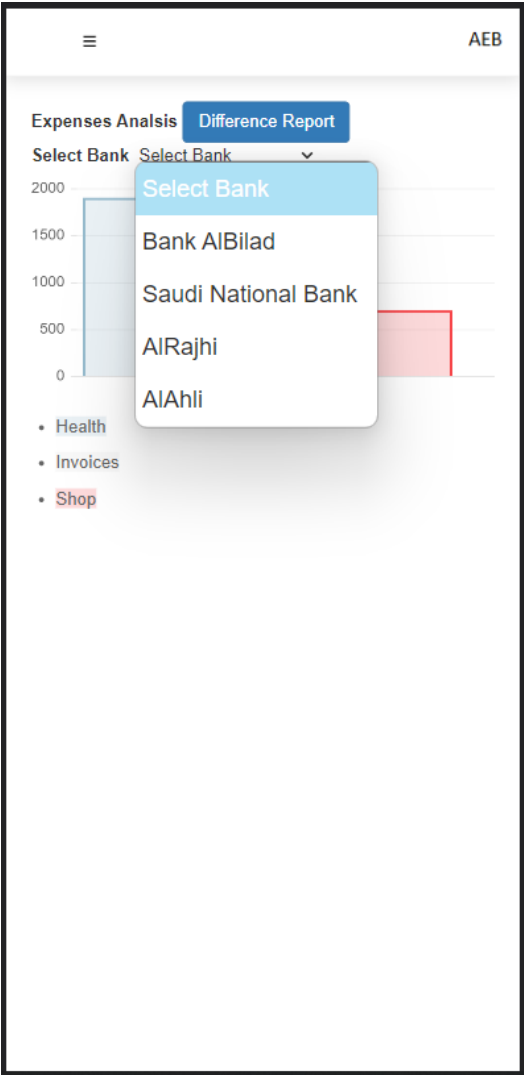
The REGISTER screen features a header with a hamburger menu icon on the left and the text 'AEB' on the right. The title 'REGISTER' is centered. Below the title are four input fields, each with a user icon on the left and a label on the right: 'name', 'email', 'Enter mobile', and 'Enter password'. At the bottom, there is a link 'Forgot Password' and a link 'Have account login here', followed by a red button labeled 'REGISTER'.



The LOGIN screen features a header with a hamburger menu icon on the left and the text 'AEB' on the right. The title 'LOGIN' is centered. Below the title are three input fields, each with a user icon on the left and a label on the right: 'mg', 'email', and 'password'. Below the input fields, there is a link 'Forgot Password' and a link 'Don't have account register here', followed by a red button labeled 'LOGIN'.







# **Chapter 5 System Implementation**

contents

5.1 Back-End Implementation

5.2 Front-End Implementation

## 5.1 Back-End Implementation

Our back end consists of five modules:

database.php : contains all required information to connect our MySQL database

Banks.php : contains all required methods to handle banks functions.

expenses.php: contains all required methods to handle expenses functions.

reports.php: contains all required methods to handle reports functions.

users.php: contains all required methods to handle user functions.

For example users.php

The file starts with

```
<?php  
  
require_once 'database.php';  
  
header("Access-Control-Allow-Origin: *");
```

The first file we include that has database information to connect with

The second line allows to reply request from outside the web site.

Then we declare response array that we will return to the user after populated.

```
$response = array();  
  
$action = $_POST["action"];
```

The action variable to know which action required from this file.

```
if ($action == "register") {  
  
    $name = $_POST['name'];  
  
    $email = $_POST['email'];  
  
    $mobile = $_POST['mobile'];  
  
    $password = $_POST['password'];  
  
    $userid = register($name, $mobile, $email, $password);
```

oo

```

if ($userid > 0) {
    $response["error"] = false;
    $response["message"] = "User is successfully added";
    $response["userdata"] = array();

    $tmp = array();
    $tmp["id"] = $userid;
    $tmp["name"] = $name;
    $tmp["mobile"] = $mobile;
    $tmp["email"] = $email;
    array_push($response["userdata"], $tmp);
}

```

If the first action is register then we receive other information required to register a new user then we call a method register and pass it the information like name, mobile, email and password and receive the result in the variable userid and if it is greater than zero then the registration is successful and we push two flags that registration is successful

```
$response["error"] = false;
```

\$response["message"] = "User is successfully added"; then we push the results in a temp array then we push it into the response array else we push two flags that registration is unsuccessful

```
$response["error"] = true;
```

```
$response["message"] = "Sorry User is not added";
```

```
else if ($action == "login") {
```

```
    $password = $_POST['password'];
```

```
    $email = $_POST['email'];
```

```
    $res = checkLogin($email, $password);
```

```
    if ($res == 0) {
```

```
        $response["error"] = true;
```

```
        $response["message"] = "Email or Password Wrong";
```



```

    } else if ($res == 1) {
        $results = getUserByEmail($email);
        $response["error"] = false;
        $response["message"] = "Successfully Login";
        $response["userdata"] = array();

        foreach ($results as $row) {
            $tmp = array();
            $tmp["id"] = $row["id"];
            $tmp["name"] = $row["name"];
            $tmp["email"] = $row["email"];
            array_push($response["userdata"], $tmp);
        }
    }
}

```

The second part if the action required is login then we receive the email and password and call method checkLogin that return 1 or 0 then we check the results if it is ==0 then the login operation failed and we return the results as a message and error flag as usual

```

$response["error"] = true;

$response["message"] = "Email or Password Wrong";

```

Else the login operation was done successfully and we get the user information from database by calling the function getUserByEmail and push all user information into the response array.

```

else if ($action == "update") {
    $name = $_POST['name'];
    $mobile = $_POST['mobile'];
    $id = $_POST['id'];

```

```

$count=updateUser($name,    $mobile , $id);
if($count>0) {
    $response["error"] = false;
    $response["message"] = "User is successfully updated";
}else{
    $response["error"] = true;
    $response["message"] = "User is not updated";
}
}

```

The previous part handles updating user information it receive the user information and calls the method updateUser and receive the return value if it greater than 0 then it is successful and we return a message else we return a message that updating is not successful.

```

else if ($action == "changepassword") {
    $password = $_POST['password'];
    $id = $_POST['id'];
    $result = changePassword($password, $id);
    if ($result >0) {
        $response["error"] = false;
        $response["message"] = "PASSWORD CHANGED";
    } else {
        $response["error"] = true;
        $response["message"] = "Sorry PASSWORD NOT CHANGED";
    }
}
}

```

The fourth part handling updating user password, it receives the new password and user id then calls method changePassowrd and passing it

new password and user id then receives result to check if it is successful or not to return the result.

```
echo json_encode($response);
```

This line return the response array to the the sender after converting the result to json.

The helper methods

```
function register($name, $mobile, $email, $password ) {  
    $password=md5($password);  
    $pdo = Database::connect();  
    $sql = 'INSERT INTO users (name,mobile,email,password )  
VALUES (?, ?, ?, ?) ';  
    $q = $pdo->prepare($sql);  
    $q->execute(array($name, $mobile, $email, $password));  
    $id = $pdo->lastInsertId();  
    Database::disconnect();  
    return $id;  
}
```

*This method aims to register a new user it accepts the required information and then it hashes the password then get a connection to the database then prepare the sql statement and get a prepared statement object then execute and pass it the required data then we get the last inserted id and disconnect database then return id.*

```
function checkLogin($email, $password) {  
    $newpassword = md5($password);  
    $sql = 'SELECT * FROM `users` WHERE email=? AND  
password=? And attempts<5';  
    $pdo = Database::connect();  
    $q = $pdo->prepare($sql);  
    $q->execute(array($email,$newpassword));  
    $result = $q->fetchAll(PDO::FETCH_ASSOC);
```

```

    if ($result) {
        return 1; // User password is correct
    } else {
        $attempts=getAttempts($email);
        if($attempts>4){
            return -1;
        }
        incrementAttempts($email,$attempts+1);
        return 0;// No User with this email
    }
}

```

This method checks user login it accepts email and password and it hashes the password and get connection to the database then prepare the sql statement and get a prepared statement object then execute and pass it the required data then we get the result if it greater then 0 then the operation is successful else it fails and increments attempts field in database.

```

function getAttempts($email) {
    $attempts=0;
    $sql = 'SELECT * FROM `users` WHERE email=?';
    $pdo = Database::connect();
    $q = $pdo->prepare($sql);
    $q->execute(array($email));
    $result = $q->fetchAll(PDO::FETCH_ASSOC);
    foreach ($result as $row) {
        $attempts= $row["attempts"];
    }
}

```

```

        return $attempts;
    }

    This function get failed user attempts to login
    function incrementAttempts($email,$attempts) {
        $pdo = Database::connect();
        $sql = 'UPDATE    users    SET    attempts=?    WHERE
(email=?) ';
        $q = $pdo->prepare($sql);
        $q->execute(array($attempts, $email));
        $count = $q->rowCount();
        Database::disconnect();
        return $count;
    }

```

This function increment user attempts to login by 1

```

function updateUser($name, $mobile, $id) {
    $pdo = Database::connect();
    $sql = 'UPDATE    users    SET    name=?,mobile=?    WHERE
(id=?) ';
    $q = $pdo->prepare($sql);
    $q->execute(array($name, $mobile, $id));
    $count = $q->rowCount();
    Database::disconnect();
    return $count;
}

```

This method update user information it accepts name ,mobile and id and get connection to the database then prepare the sql statement and get a prepared statement object then execute and pass it the required data then we get the result if it greater then 0 then the operation is successful else it fails.

```

function changePassword($password, $id) {
    $password = md5($password);
    $sql = ' UPDATE    users  SET    password=?  WHERE
(id=?) ';
    $pdo = Database::connect();
    $q = $pdo->prepare($sql);
    $q->execute(array($password, $id));
    $count = $q->rowCount();
    return $count;
}

```

This method update user password it accepts new password and user id and it hashes the password and get connection to the database then prepare the sql statement and get a prepared statement object then execute and pass it the required data then we get the result if it greater then 0 then the operation is successful else it fails.

```

function getUserByEmail($email) {
    $sql = 'SELECT    * FROM `users`  WHERE  email=?';
    $pdo = Database::connect();
    $q = $pdo->prepare($sql);
    $q->execute(array($email));
    $result = $q->fetchAll(PDO::FETCH_ASSOC);
    return $result;
}

```

This method return user information by email it accepts email and it and get connection to the database then prepare the sql statement and get a prepared statement object then execute and pass it the required data then return the result.

## 5.2 Front end Implementation

Folder structure

Fonts: contains fonts used in application

Images : contains images used in application

Js: contains all javascript modules

Styles: contains all css files

Views: contains all application views

Index.html: the main html file

We will explain one controller UserController with its views

```
angular.module('BankApp').controller('UserController', function  
($rootScope, $scope, $http, $state, $timeout, Urls) {
```

The controller starts with built in services and one defined service Urls.

```
$rootScope.user.name = "";  
$rootScope.user.email = "";  
$rootScope.user.password = "";  
$rootScope.user.id = 0;  
$rootScope.user.mobile = "";  
$scope.user = $rootScope.user ;
```

Then we empty RootScope user object and we assign it to Scope user object.

```
$scope.register = function () {  
    if ($scope.user.name == null || $scope.user.name.trim() === "") {  
        alert("You must Enter your name");  
        return;  
    }  
    if ($scope.user.mobile.trim() == "" || $scope.user.mobile.length !==  
12) {  
        alert("You must Enter your mobile number");  
        return;  
    }
```

```

    }

    if ($scope.user.password.trim() == "" || $scope.user.password.length < 6
    || $scope.user.password.length > 20) {

        alert("You must Enter password");

        return ;

    }

    $rootScope.showLoader();

    var data = $.param({
        action: "register",
        name: $scope.user.name,
        mobile: $scope.user.mobile,
        email: $scope.user.email,
        password:$scope.user.password,
    });

    $http.post(Urls.apiUrl + 'users.php', data, Urls.config)
        .then(function (result) {
            if (result.data.error == false) {
                $rootScope.user.name = result.data.userdata.name;
                $rootScope.user.email = result.data.userdata.email;
                $rootScope.user.id = result.data.userdata.userid;
                $rootScope.hideLoader();
                $state.go("home");
            } else {
                alert(result.data.message);
                $rootScope.hideLoader();
            }
        })

```



```

        }, function (error) {
            $rootScope.hideLoader();
        });
    };

```

The method register is responsible for registering a new user.

First it validates user information then collect this information into one object data then make http call to the our API users.php with the data required then waits for response.

If no error happened then we assign all data received from the server to the rootScope user else we display error message.

```

$scope.login = function () {
    $rootScope.showLoader();
    var data = $.param({
        action: "login",
        email: $rootScope.user.email.trim(),
        password: $rootScope.user.password.trim()
    });

    $http.post(Urls.apiUrl + 'users.php', data, Urls.config)
        .then(function (result) {
            if (result.data.error == false || result.data.error == 'false') {
                $rootScope.user = result.data.userdata[0];
                $rootScope.hideLoader();
                $rootScope.storeUser();
                $state.go("home");
            } else {
                alert("Email Or Password Error");
            }
        })
    };

```

```

        $rootScope.hideLoader();
    }
    , function (error) {
        $rootScope.hideLoader();
    });
};

```

The second method is login method which collect email and password into one data object then send it to the server and waits for result if it is ok then we assign data received from the server into rootScope.user and then we store it in local storage of browser so we keep user logged in then we take the user into home page.

Else we display the error message.

```

$scope.validateLogin = function ( ) {
    if (!$rootScope.validateEmail($rootScope.user.email)) {
        return false;
    }
    if ($rootScope.user.password == "undefined" ||
    $rootScope.user.password == "" || $rootScope.user.password == null) {
        return false;
    }
    if ($rootScope.user.password.length < 6 ||
    $rootScope.user.password.length > 20) {
        return false;
    }
    return true;
};

```

This method validate user data before login

```

$scope.change = function () {
    if($rootScope.user.id==undefined || isNaN($rootScope.user.id)){

```

```

    $state.go("home");
}

    var data = $.param({
        action: "changepassword",
        id:$rootScope.user.id ,
        password: $rootScope.user.password
    });

    $http.post(Urls.apiUrl + 'users.php', data, Urls.config)
        .then(function (result) {
            console.log(result);
            if (result.data.error == false) {
                alert("Password Changed");
                $state.go("home");
            } else {
                alert("Error happened");
            }
        });
};

```

This method is responsible for changing user password it collects data into data object then make post call to Users.php with data collected and wait for result and alert a message to the user by the result.

```

$scope.validateChangePassword = function ( ) {
    if ($rootScope.user.password == "" || $rootScope.user.password.length
    < 4 || $rootScope.user.password.length > 20) {
        return false;
    }
}

```

```

    if ($rootScope.user.password != $rootScope.user.repassword) {
        return false;
    }
    return true;
};

```

This method is responsible for validating user passwords if it ok then the send button will be enabled else be disabled.

```

    $rootScope.getUser();
}); // end module

```

This line gets user from local storage if stored already.

Now we will discuss views for example login view

```

<div class="page-content page-content-full" ng-controller="UserController">
    <div class="page-login top-10">
        
        <h3 class="uppercase ultrabold top-30 bottom-0" style="text-align:center;"> Login
        </h3>
        <div class="page-login-field top-15">
            <i class="fa fa-user"></i>
            <input type="text" placeholder="email" ng-model="user.email">
        </div>
        <div class="page-login-field bottom-20">
            <i class="fa fa-lock"></i>
            <input type="password" placeholder="password" ng-model="user.password">
        </div>
        <div class="page-login-links bottom-10">
            <a class="create float-left" href="#recover"><i class="fa fa-eye"></i>
Forgot Password </a>
            <a class="create float-left" href="#register"><i class="fa fa-eye"></i>
Dono't have account register here </a>
        </div>
    </div>

```

```

        </div>

        <button class="button bg-highlight button-full button-rounded button-s
uppercase ultrabold" ng-click="login()"> Login </button>

    </div>

</div>

<script type="text/javascript">
$(document).ready(function () {
closeMenu ();
});
</script>

```

The view is a html page which bind to UserController

It has two text boxes one bound to user email and the second was bound to user password then a button to login function

There are two links one for register a new account and the other for recover account

The code of java script is to close side menu after click on login link.

The second view is register view

```

<div class="page-content page-content-full" ng-controller="UserController">

    <div class="page-login top-10">

        <h3 class="uppercase ultrabold top-30 bottom-0 " style="text-align:center;"> Register </h3>

        <div class="page-login-field top-15">

            <i class="fa fa-user"></i>

            <input type="text" placeholder="name" class="form-control input-circle-right" ng-model="user.name"> </div>

        <div class="page-login-field top-15">

            <i class="fa fa-user"></i>

            <input type="text" placeholder=" email" class="form-control input-circle-right" ng-model="user.email">

        </div>

    <div class="page-login-field top-15">

```

```

        <i class="fa fa-user"></i>

<input type="text" ng-model="user.mobile" class="form-control input-circle-
right" placeholder="Enter mobile" required> </div>

<div class="page-login-field bottom-20">

    <i class="fa fa-lock"></i>

<input type="password" ng-model="user.password" class="form-control input-
circle-right" placeholder="Enter password"> </div>

<div class="page-login-links bottom-10">

    <a class="create float-left" href="#recover"><i class="fa fa-eye"></i>
Forgot Password </a>

    <a class="create float-left" href="#login"><i class="fa fa-eye"></i> Have
account login here</a>

    <div class="clear"></div>

</div>

<button class="button bg-highlight button-full button-rounded button-s
uppercase ultrabold" ng-click="register()"> Register </button>

</div>

</div>

<script type="text/javascript">
$(document).ready(function () {
closeMenu ();
});
</script>

```

The view is a html page which bind to UserController

It has four text boxes one bound to user email and the second was bound to user name and the third for mobile and the fourth for password then a button to register function

There are two links one for login and the other for recover account

The code of java script is to close side menu after click on login link.

## Chapter 6 Testing

Function	tested
Register	done
Login	done
Change Password	done
Recover Password	Not done
Update Profile	done
Bank Account	done
Budget item	done
Budget Report	done
Block user account	done