

Ain Shams University

Faculty of Engineering

CESS

Spring 2019



KAGAMY NB **BANKING SYSTEM**

Submitted by:

Ahmed Khaled Aly (17P5026)

Andrew Yasser Shaker (17P6069)

George Emad Younan (17P8182)

Karim Mikhael Farid (17P3061)

Meyer Wafik Wadea (17P6073)

Youssef Magdy Moneer (17P8207)

Submitted to:

Dr. Gamal Abd ElShafy

Contents

1. INTRODUCTION:.....	1
2. FUNCTIONAL REQUIREMENTS:.....	2
3. NON-FUNCTIONAL REQUIREMENTS:.....	3
3.1 Organizational Requirements:	3
3.2 Product requirements:	3
3.3 External requirements:	4
4. SYSTEM REQUIREMENT:	4
5. USE-CASE DIAGRAM:	5
6. NARRATIVE DESCRIPTION OF USE CASES:	6
7. REQUIREMENTS VALIDATION:	12
7.1 Requirements indices:.....	12
7.2 Requirements Traceability Matrix:	13
7.3 Source Traceability Matrix:	14
8. CLASS MODEL:.....	15
8.1 Noun Extraction Stage:	15
8.2 CRC Cards:.....	16
9. ACTIVITY DIAGRAMS:	22
10. SEQUENCE DIAGRAMS:	27
11. DETAILED CLASS DIAGRAM:.....	29
12. USER INTERFACE DESIGN:.....	30
13. OBJECT-CLIENT DIAGRAM:.....	31
14. PROCESS MODEL	32
15. USER GUIDE:.....	34

Our aim is to develop a Bank system that can satisfy the customers' financial needs with effectiveness and ease, with respecting the customers' data security and integrity. KAGAMY National Bank prioritizes the customer through different developing process models to predict how the user would deal with the system's different functionalities.

1. INTRODUCTION:

KAGAMY NB is a banking software system that is going to be developed as an interactive transaction-based application where the system allows the user to create a new account or login into an existing account that can be a transactional or deposit account.

The system should calculate the taxes and interests for each account depending on its type. Also, the user should be able to transfer money to another accounts or pay governmental bills as water, electricity, and taxes. Moreover, the system should provide the user with monthly bank statement and send notifications as SMS and e-mails when transactions occur. The system should allow the user to request customer service and help or know the bank's locations even without having an account, as well as giving the user the opportunity to request loans with its different types which are Car, premises , and projects loans.

The system should secure the users data and money using a well-known security software. Also, the users' data should be used only by admins and users themselves .The system should tolerate errors and be robust that it reopens in less than 2 minutes if a big failure occurs. However, that doesn't mean that errors should be frequent as there mustn't be more than 2% of failures per 10000 transactions. Moreover, high speed is needed as the system should be able to make 3 transactions per second for each user.

Governmental companies' laws should be applied while paying bills as well as Egyptian laws of banks and financial businesses of 1987. Also, international standards laws should be applied for online shopping.

2. FUNCTIONAL REQUIREMENTS:

- Create bank accounts.
- Calculate interests.
- Make money transactions.
- Check current balance.
- Calculate monthly interest.
- Calculate monthly taxes.
- Provide monthly bank statement.
- Customer service and problem solution.
- Provide currents loans information.
- Provide bank & ATMs locations.
- Request appointments for obtaining a credit card.
- Request appointments with managements for obtaining new loans.
- Send Notifications.
- Provide transactions history.

3. NON-FUNCTIONAL REQUIREMENTS:

3.1 Organizational Requirements:

- Programming Language: Java.
- Preferred programming paradigm: object- oriented programming.
- Accounts created by clients can be used after creating them by one working day of the bank.
- The bank does not enquire about the funds source.
- the banking system support different currencies (Euro, US Dollar& Egyptian pound).
- the system must be delivered within 1 year, and with monthly reviews to management and technical team to what has been done.
- Solving technical issues must be detected exactly and solved step by step.

3.2 Product requirements:

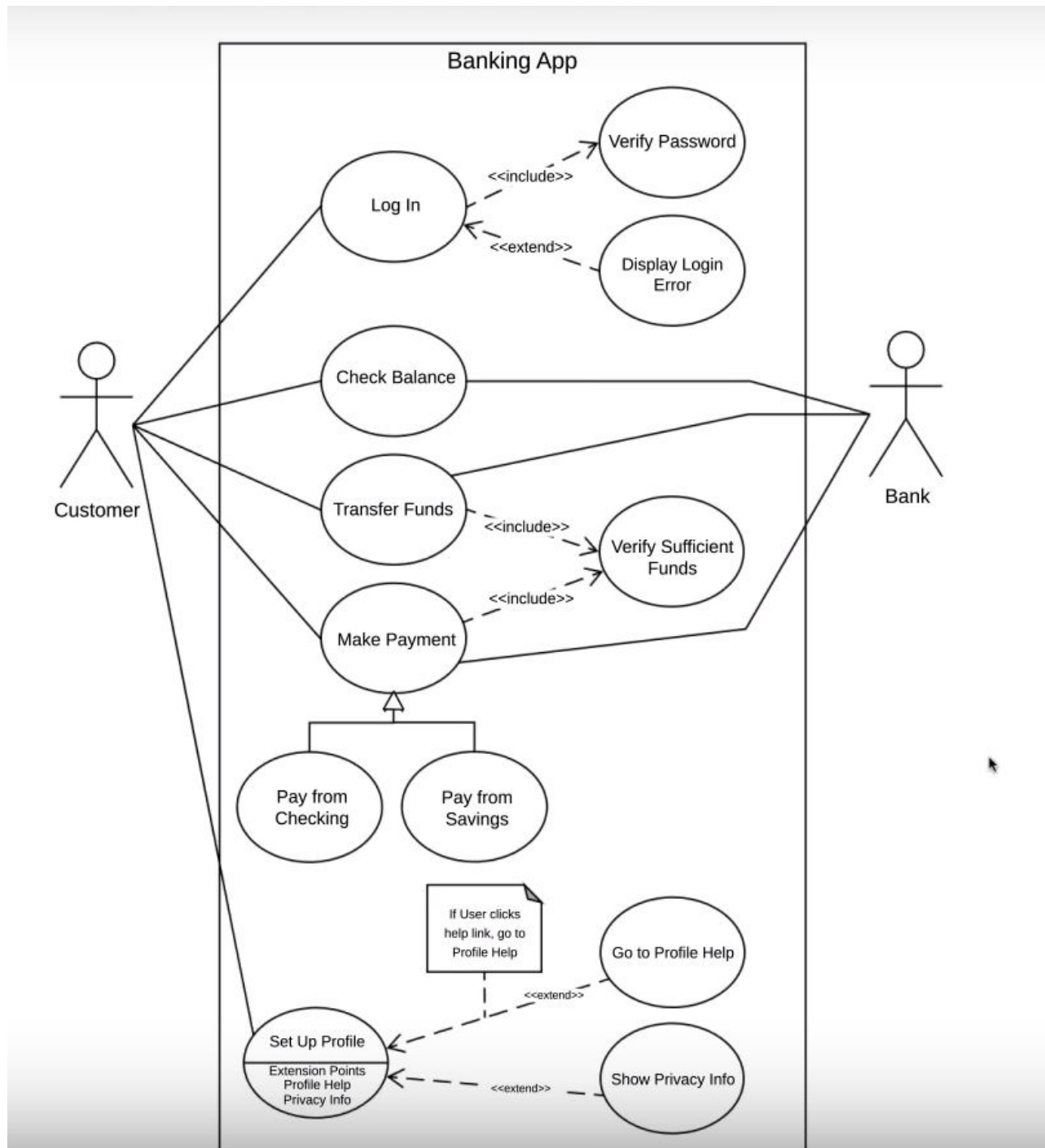
- Secure software provide safe storage of clients' personal data.
- Easy to use and maintain by inexperienced clients by offering one day training sessions.
- requires high speed (1 second for 3 transactions) and low response time (1 GB RAM minimum).
- doesn't require high graphics (minimum Radeon M470X graphics card).
- System percentage of failure must be less than 2% per 10000 transaction, and in case of system failure the system must restart after a duration of max 2 minutes.
- Tolerate common faults made by clients (not logging out before closing program)
- .
- available on Windows (xp-10), Mac, Linux.
- required storage: 1 Tb.
- require direct contact with bank servers and links with bank's website.
- Friendly user interface.

3.3 External requirements:

- Apply laws of governmental companies (Electricity, water & taxes).
- Apply laws of international standards to online shopping platforms.
- Clients personal information are secured in the bank database, with restricted access to verified employees (Management and admins).
- Apply Egyptian laws of banks and financial businesses of 1987.

4. SYSTEM REQUIREMENT:

- Money transaction
- pay bills
- transfer money from and to account
- online shopping.
- Creation of bank account.
- Customer service.
- Send Notifications.
- Send notifications if any transaction, login, or balance checking happens.
- Reporting problems
- Transaction error
- login failure

5. USE-CASE DIAGRAM:*Figure 1: Use Case Diagram*

6. NARRATIVE DESCRIPTION OF USE CASES:

LOGIN USE CASE

Use Case Name	Login
Related Requirements	User must have a user name and password
Goal in Contest	Check the User's details
Preconditions	User must have an account
Successful and condition	Details verified
Failed End Condition	Details not verified
Primary Users	User
Secondary User	Database

Table 1: Narrative description of Login use cases.

Main Flow	Step	Action
	1	User enters details
	2	Password verified
	3	Account details displayed
Extension	2.1	Compare the password entered with the password in the database
	2.2	Password mismatched
	2.3	Error message displayed

Table 2: Narrative description of Login use case.

TRANSFER FUNDS USE CASE

Use Case Name	Transfer Funds
Related Requirements	Account Present
Goal in Contest	Funds (Money) to be transferred
Precondition	Account Present, Enough Funds
Successful End Condition	Funds transferred
Failed End Condition	Funds not successful
Primary User	User
Secondary User	Data Base

Table 3: Narrative description of Transfer Funds use case.

Main Flow	Step	Action
	1	User enters details(Username and password)
	2	Password verified
	3	Account details displayed
	4	Request transferring funds
	5	Account to be transferred into entered
	6	Funds transferred
Extension	4.1	Compare the amount of funds requested to be transferred with the amount of funds in the account
	4.2	No sufficient funds
	4.3	Display error message

Table 4: Narrative description of Transfer Funds use case.

Check Balance Use Case

Use Case Name	Check Balance
Related Requirements	Account Present
Goal in contest	Check account's funds
Preconditions	Account present
Successful End Condition	Account funds displayed
Failed End Condition	None
Primary User	User
Secondary User	Data Base

Table 5: Narrative description of Check Balance use case.

Main Flow	Step	Action
	1	User enter details (Username and Password)
	2	Password Verified
	3	User Request to check balance
	4	Balance displayed

Table 6: Narrative description of Check Balance use case.

7. REQUIREMENTS VALIDATION:

7.1 Requirements indices:

1.1	Create bank accounts.
1.2	Make money transactions.
1.3	Check current balance.
1.4	Calculate monthly interest.
1.5	Calculate monthly taxes.
1.6	Provide monthly bank statement.
1.7	Customer service and problem solution.
1.8	Provide currents loans information.
1.9	Provide bank & ATMs locations.
2.1	Request appointments for obtaining a credit card.
2.2	Request appointments with managements for obtaining new loans.
2.3	Send Notifications.
2.4	Provide transactions history.

Table 7: Requirements indices.

7.2 Requirements Traceability Matrix:

Requirements ID	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.1	2.2	2.3	2.4
1.1												R	
1.2	D		D									D	
1.3	D												
1.4	D				D								
1.5	D												
1.6	D	R	D	D	D							D	D
1.7	D											R	
1.8	D											R	
1.9	R												
2.1	D											D	
2.2	D							R				D	
2.3	D					R				R	R		R
2.4	D	D											

Table 8: Requirements Traceability Matrix.

7.3 Source Traceability Matrix:

Requirements ID	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.1	2.2	2.3	2.4
Client	T	T	T	T	T	T	T		T	T	T	T	T
Management	T						T	T		T	T		

Table 9: Source Traceability Matrix.

8. CLASS MODEL:

8.1 Noun Extraction Stage:

Nouns: bank account & account type, taxes, Interests, governmental bills, water, electricity, taxes, user, monthly bank statement, notifications, SMSs, e-mails, transactions, system, customer service, locations, loans, Car loans, premises loans, projects loans.

Excluded nouns that lie outside the problem boundary: governmental bills, water, electricity.

Assumptions:

- Bank account is an abstract class and there will be two accounts types Transactional and deposit bank account that inherits bank account.
- Taxes and interest are attributes in bank account class and will be calculated differently in transactional and deposit bank account classes using and overridden methods.
- Transaction is a class with attributes names, description and amount.
- Monthly bank statement is a method in bank account class that will provide calculated taxes and interests and transaction.
- customer service is a method called support in bank account class.
- Notifications will be a method called send notifications that will send automated e-mails and SMSs to clients.
- Locations is a static method that can be used by a user that does not have bank account.
- Loan is class with sub classes: car loan, project loan and premises loan.

8.2 CRC Cards:

<p>CLASS</p> <p>Bank Account</p> <p>SUB CLASSES: Deposit bank account -Transactional bank account</p>
<p>RESPONSIBILITY</p> <ol style="list-style-type: none">1.Update and maintain personal data of users.2.Provide technical support to users to solve problems Concerning bank accounts.3.show current loans information of clients.4.Provide appointments to users to request cards.5.show clients transactions.6.Show clients current balance.7.Provide monthly bank statements to users.8.Send notifications to users via E-mails and phone numbers.9.Provide interest and taxes calculations to user.
<p>COLLABORATION</p>

Figure 2: CRC card.

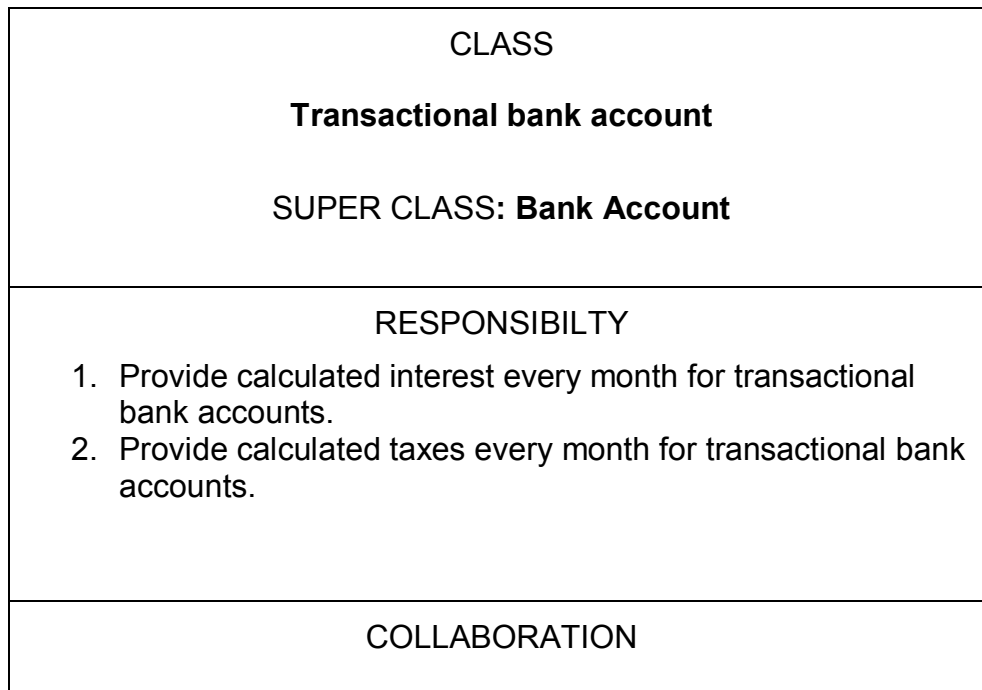


Figure 3: CRC card of Transactional bank account.

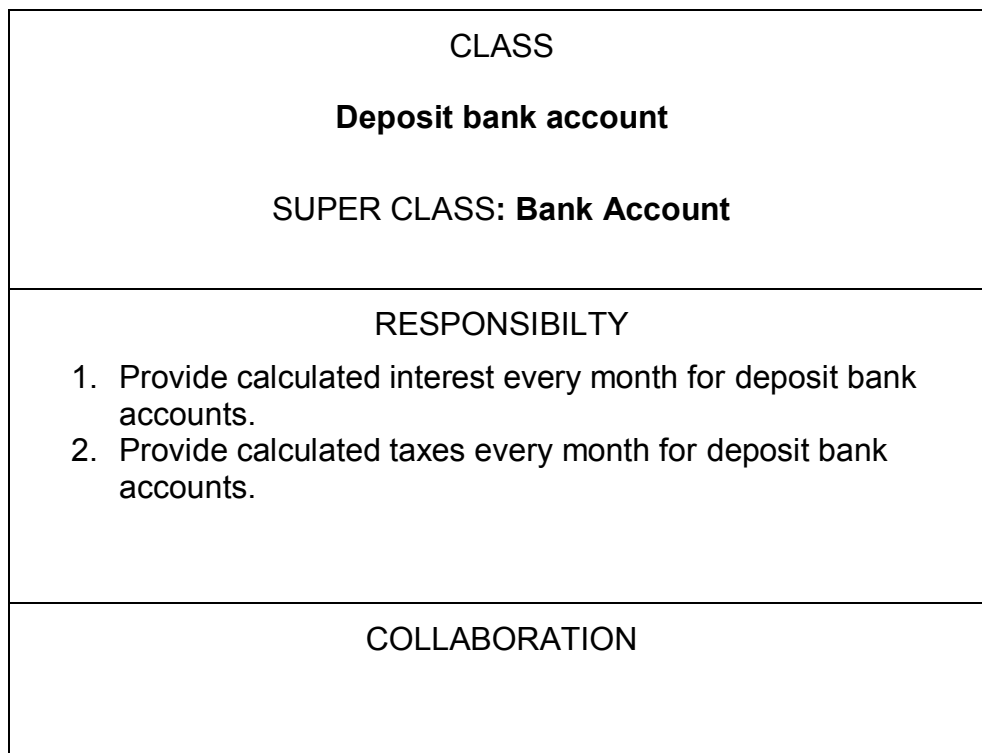


Figure 4: CRC card of Deposit bank account.

<p>CLASS</p> <p>Loan</p> <p>SUB CLASSES: Car Loan - Project Loan - Project Loan</p>
<p>RESPONSIBILITY</p> <p>1.Adds new loan.</p> <p>2.Set Amount of the new loan.</p> <p>3.Set securities of the new loan.</p> <p>4. send message to Bank Account to show current loans.</p>
<p>COLLABORATION</p> <p>1.Bank Account</p>

Figure 5: CRC card of Loan.

<p>CLASS</p> <p>Car Loan</p> <p>SUPER CLASSES: Loan</p>
<p>RESPONSIBILITY</p> <p>1.Set wanted car type to request loan.</p>
<p>COLLABORATION</p>

Figure 6: CRC card of Car Loan.

<p>CLASS</p> <p>Project Loan</p> <p>SUPER CLASSES: Loan</p>
<p>RESPONSIBILITY</p> <p>1.Set Set percentage of project success to request loan.</p>
<p>COLLABORATION</p>

Figure 7: CRC card of Project Loan.

<p>CLASS</p> <p>Premises Loan</p> <p>SUPER CLASSES: Loan</p>
<p>RESPONSIBILITY</p> <p>1.set address of premises to request car loan to bank. 2.set area of the land to bank to request loan</p>
<p>COLLABORATION</p>

Figure 8: CRC card of Premises Loan.

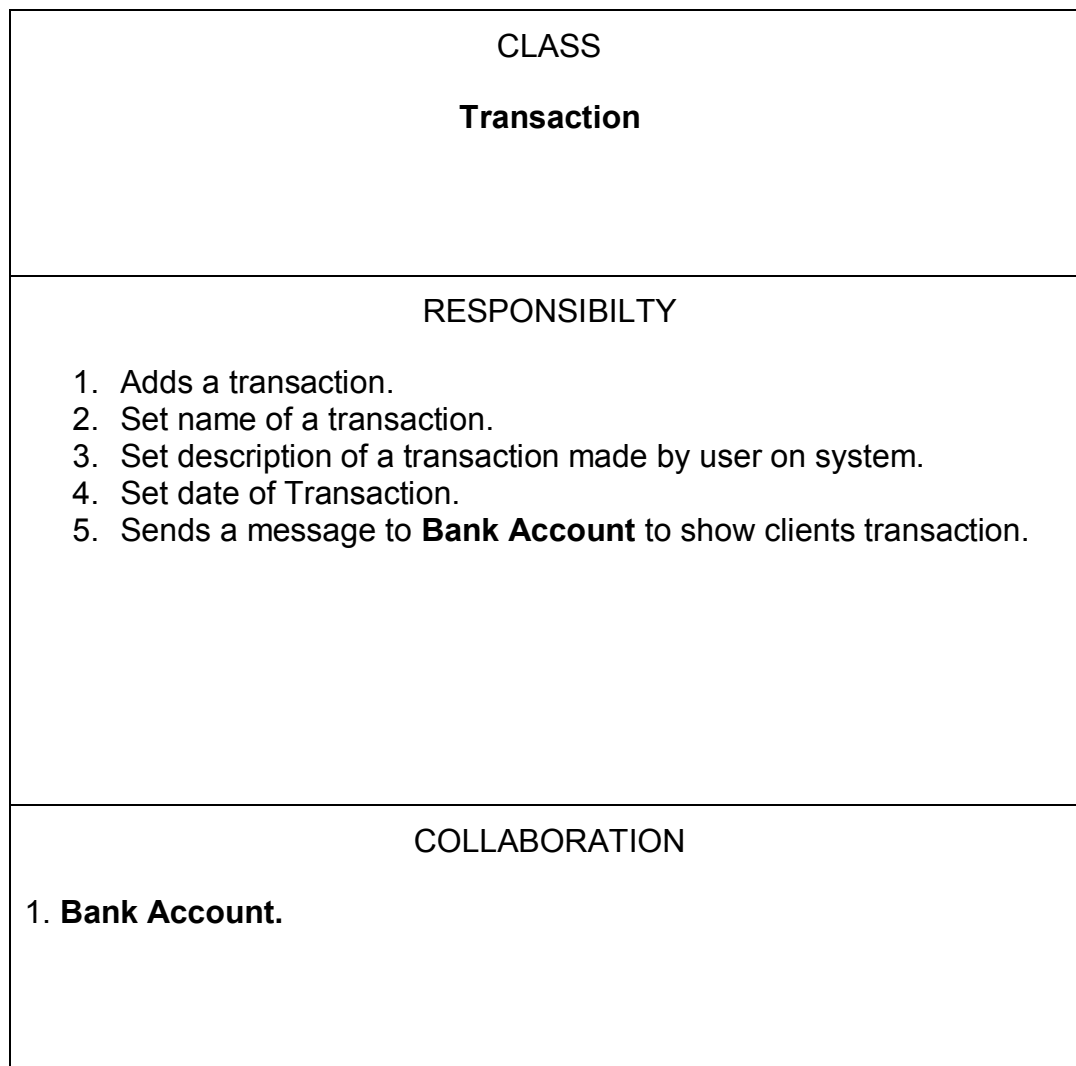
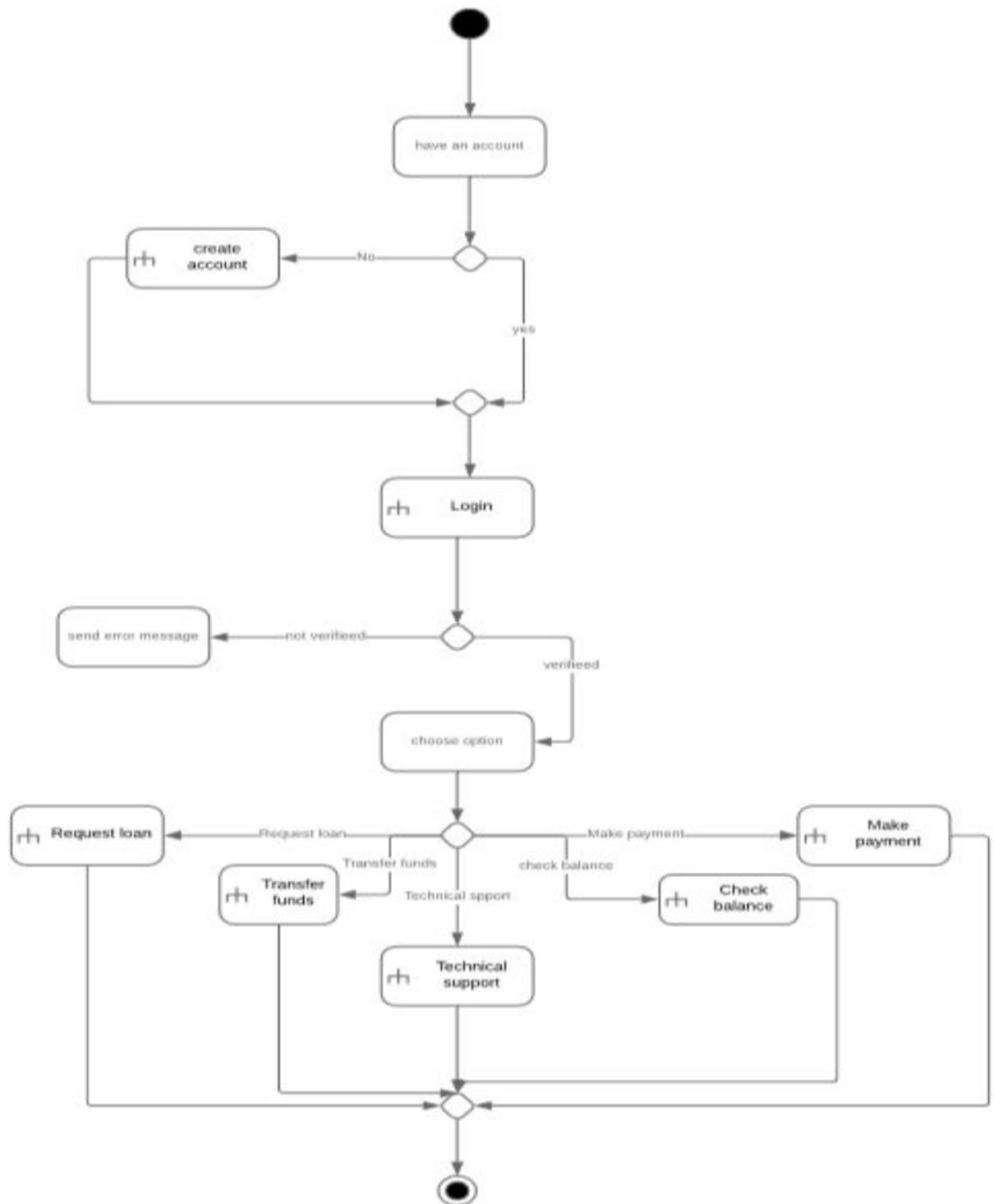


Figure 9: CRC card of Transaction.

9. ACTIVITY DIAGRAMS:*Figure 10: Activity Diagram.*

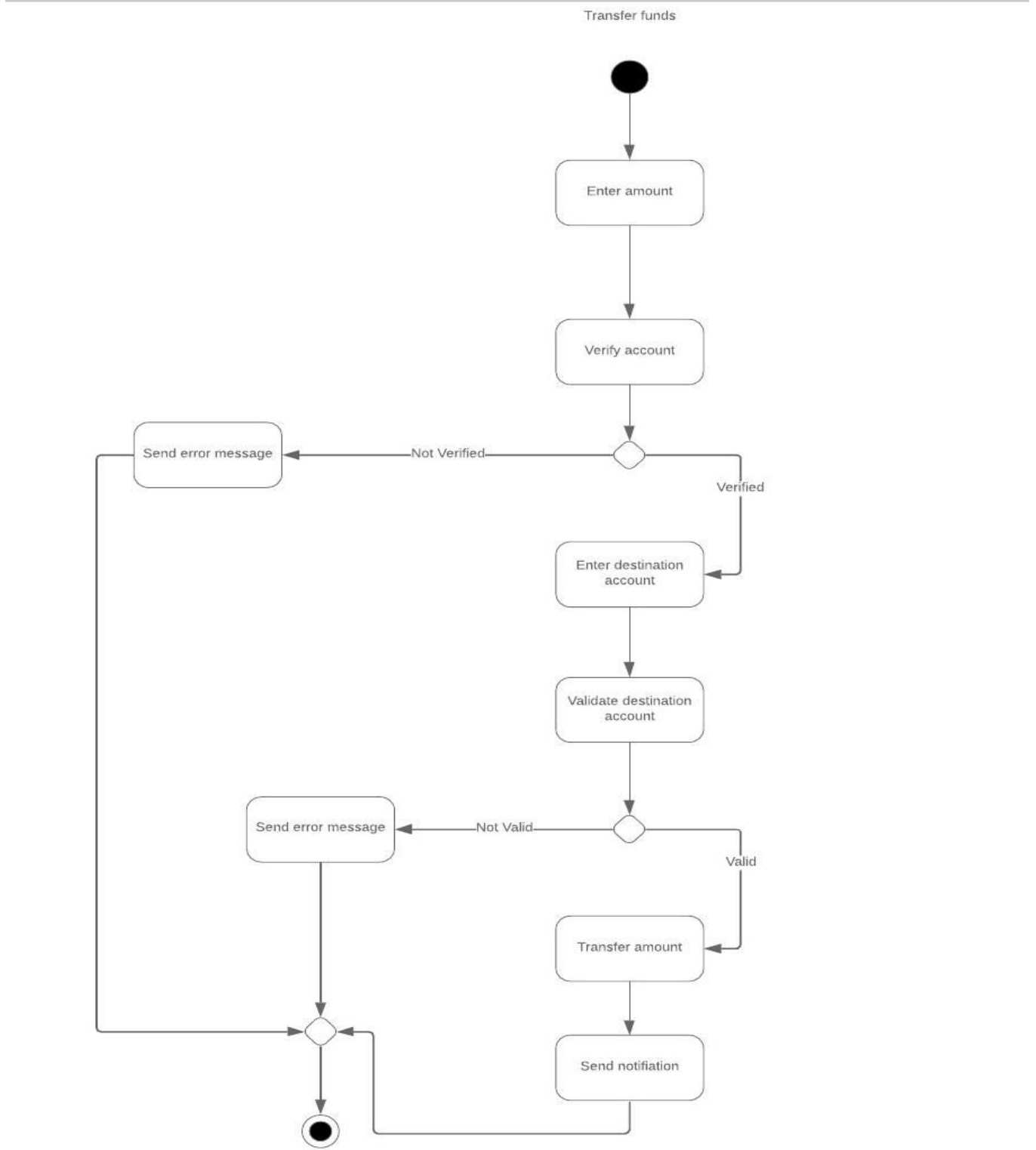


Figure 11: Activity Diagram of transfer funds.

Technical Support

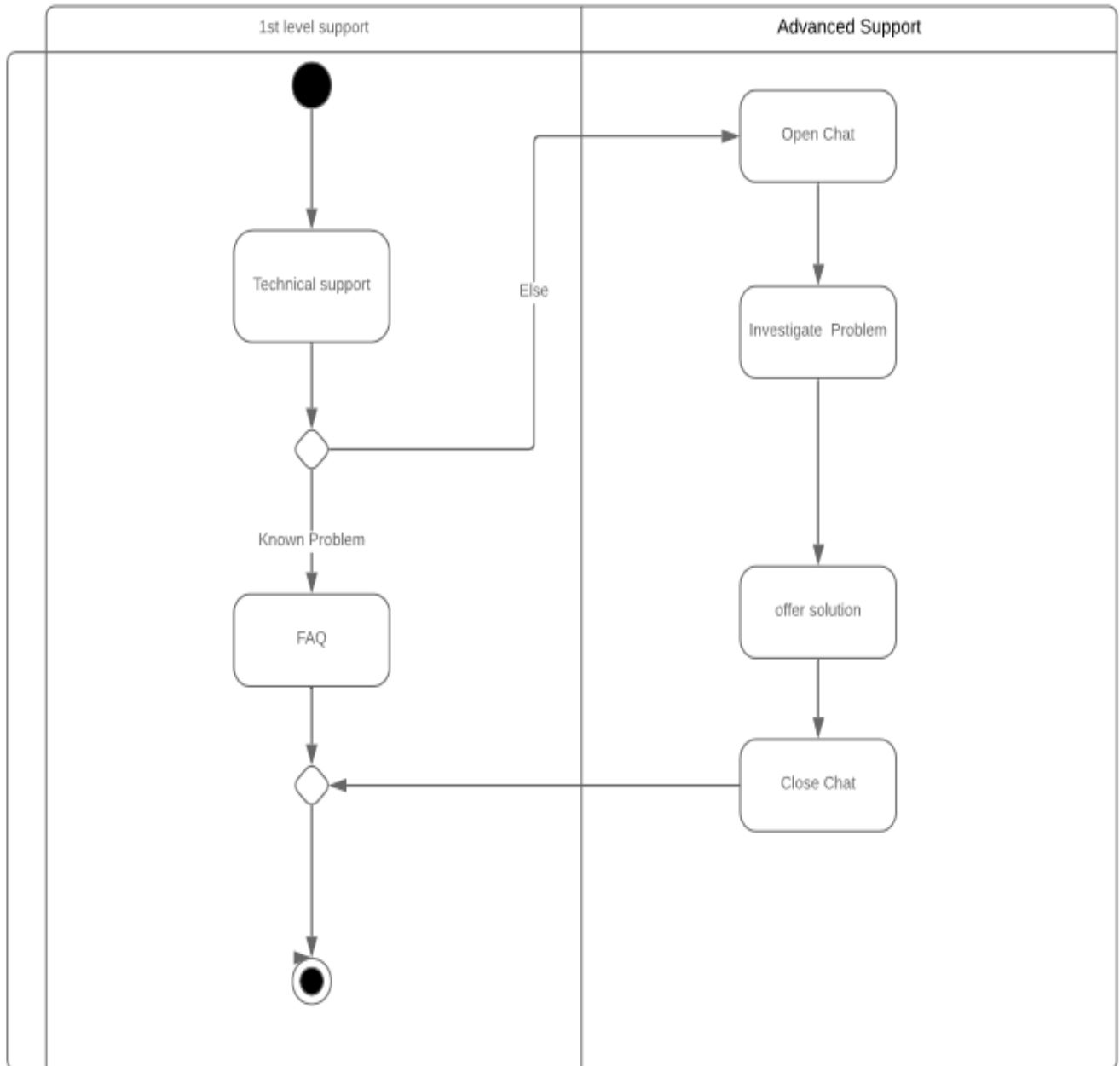


Figure 12: Activity Diagram of technical support.

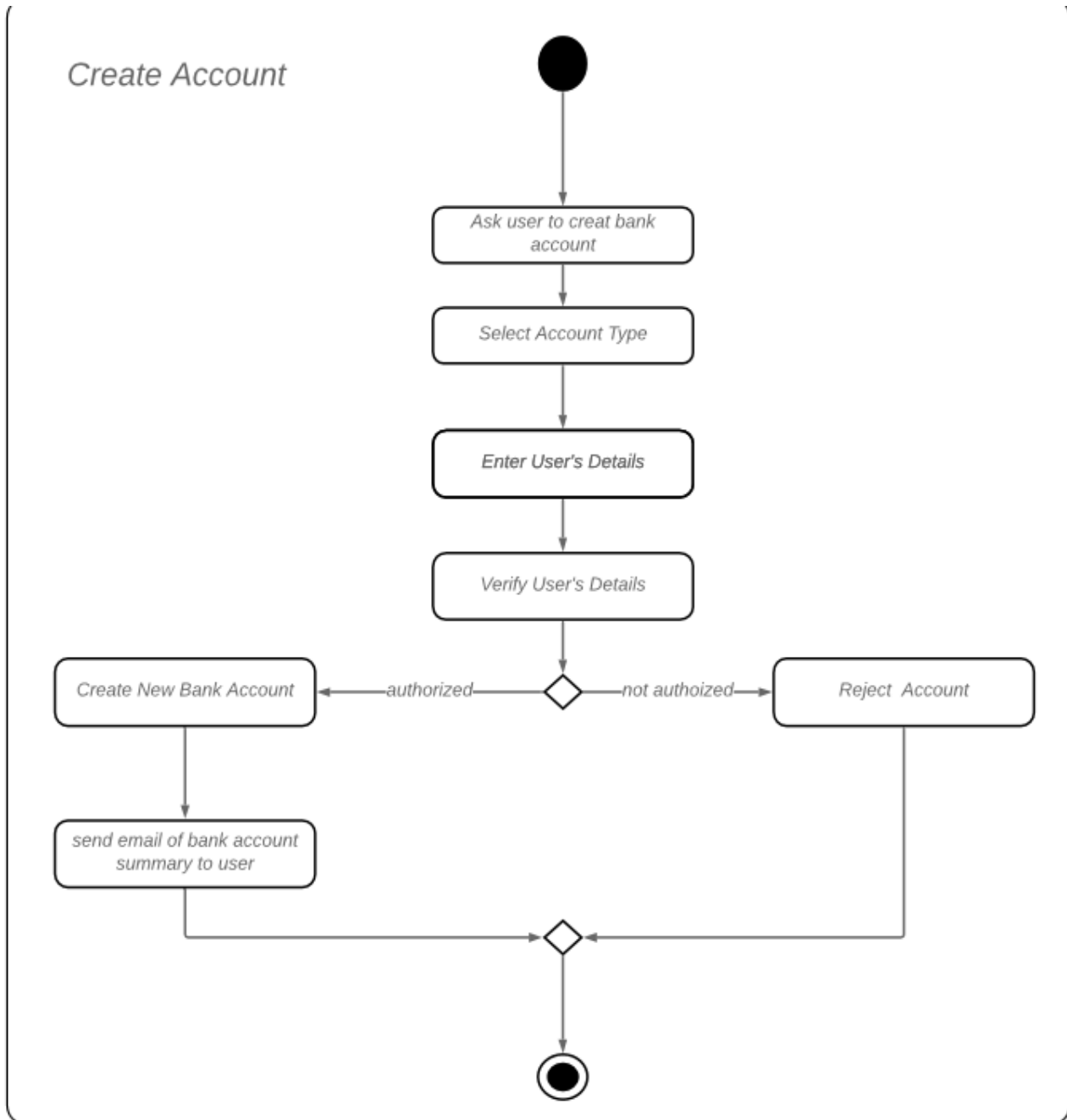


Figure 13: Activity Diagram of Create Account.

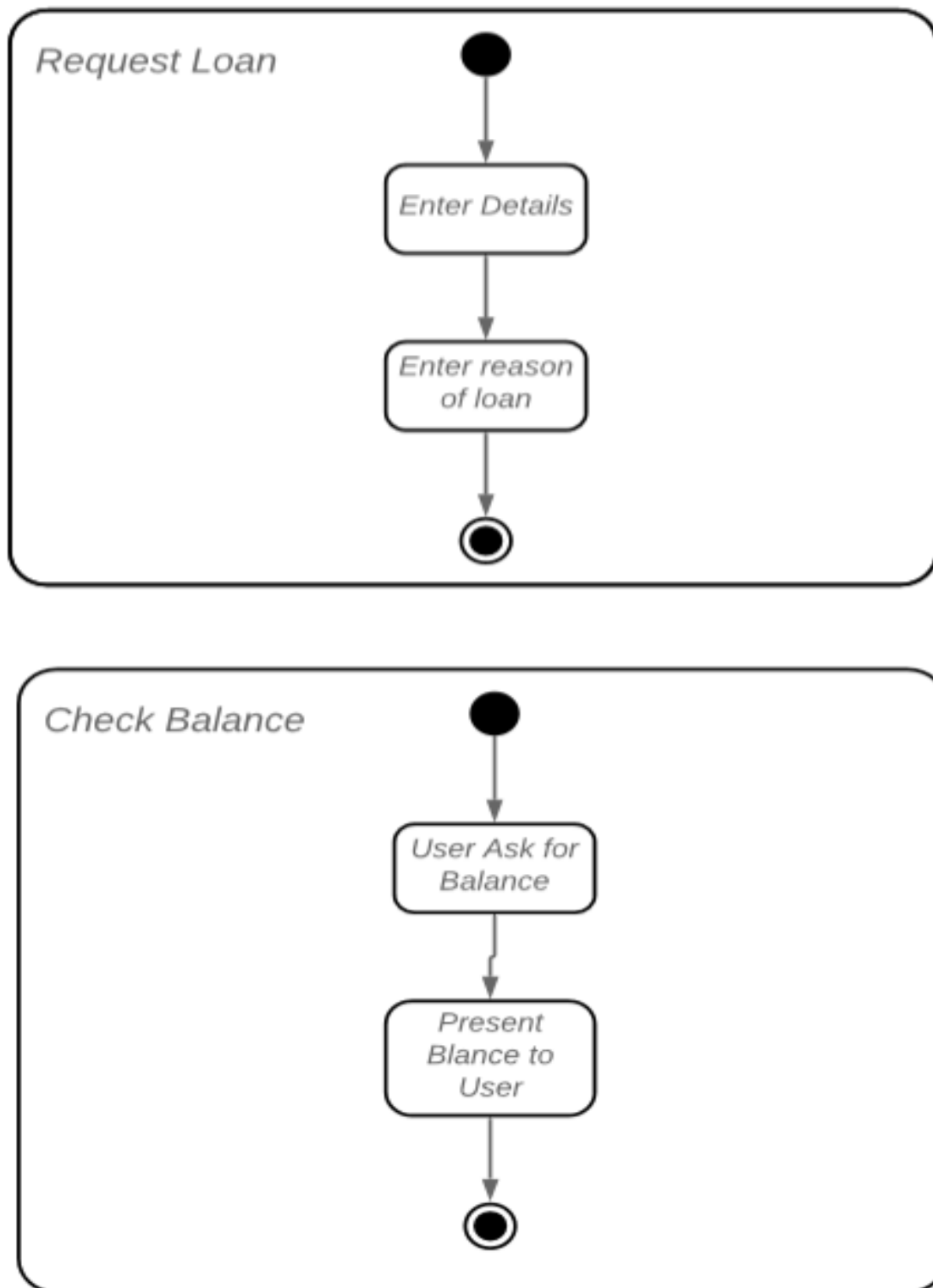
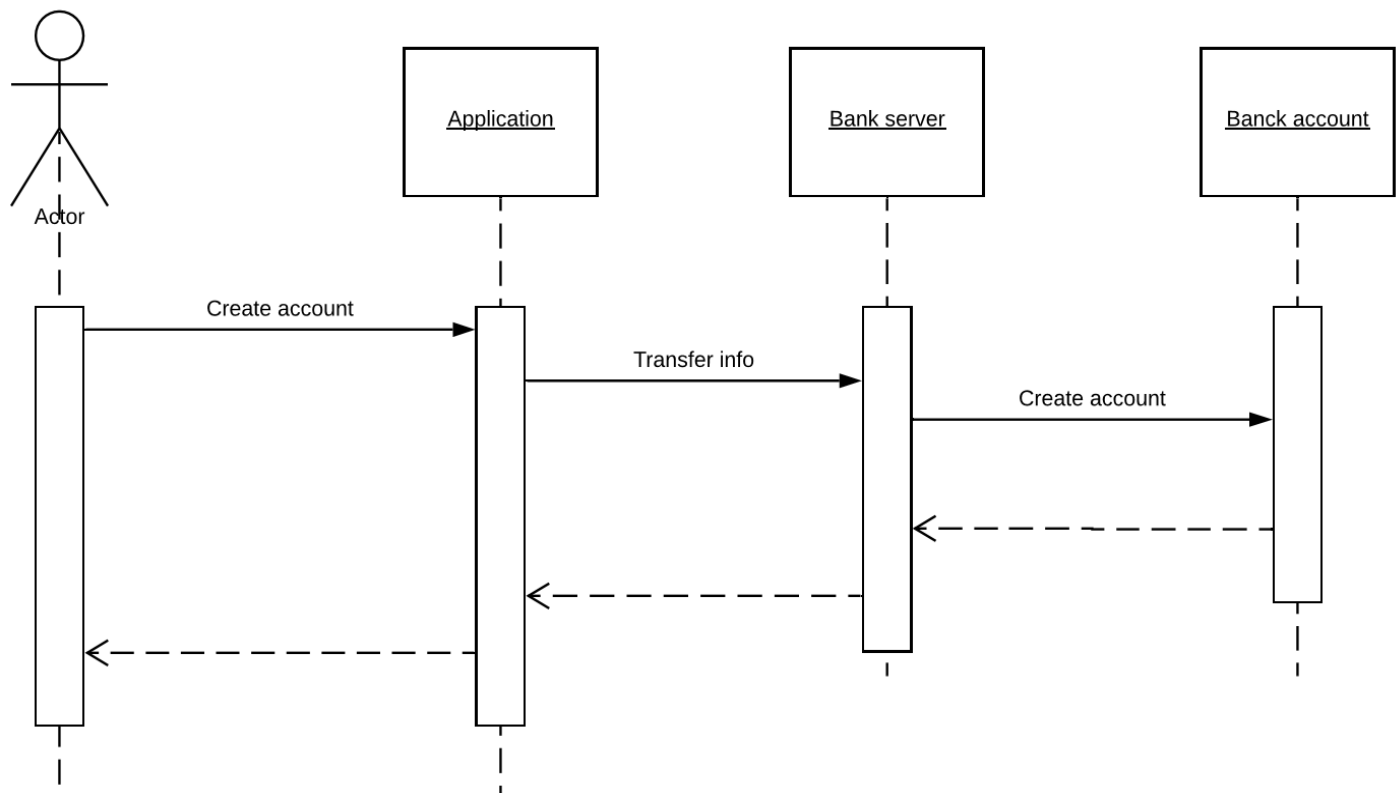


Figure 14: Activity Diagram of Request Loan and Check Balance.

10. SEQUENCE DIAGRAMS:*Figure 15: Sequence Diagram.*

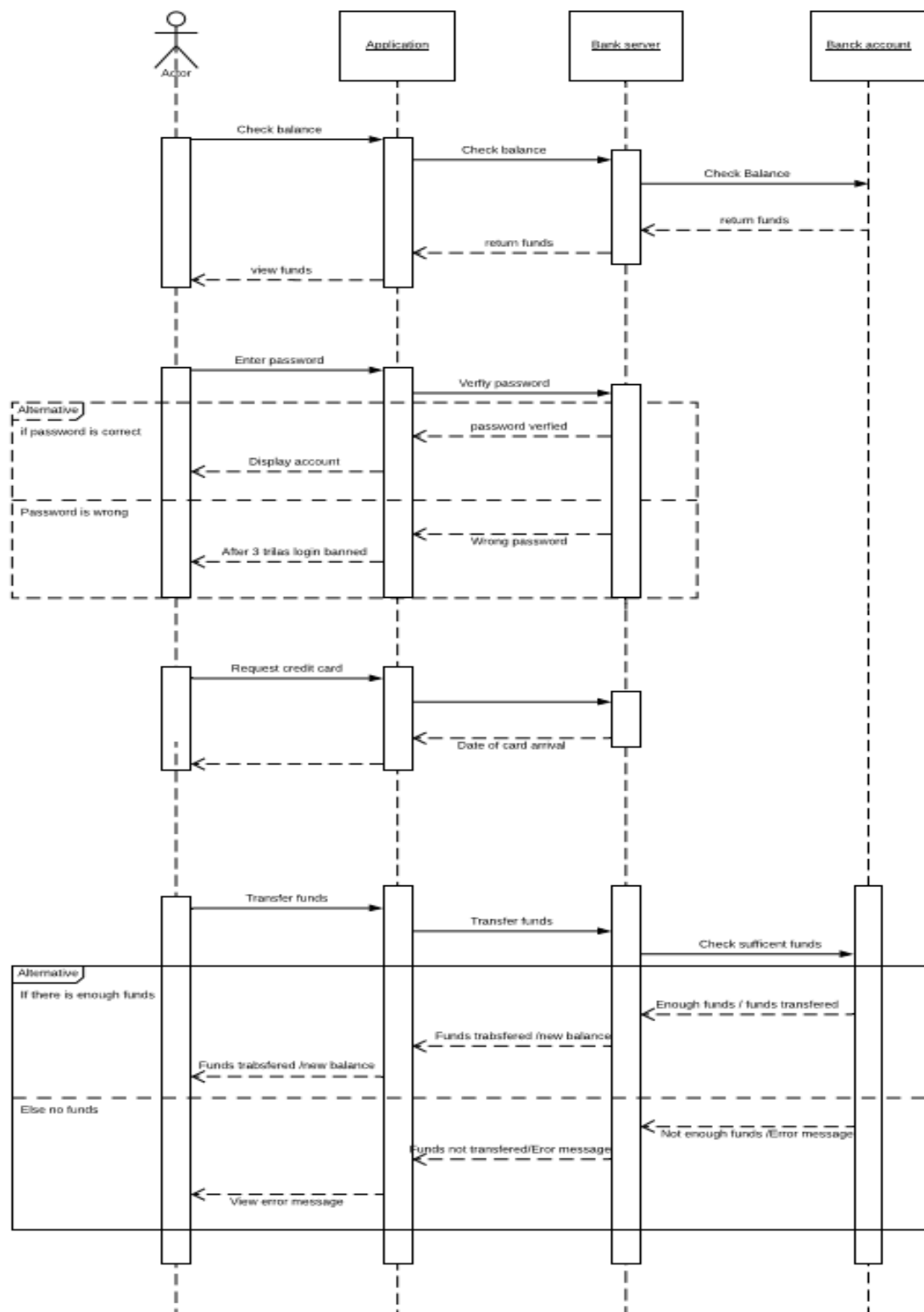


Figure 16: Sequence Diagram.

11. DETAILED CLASS DIAGRAM:

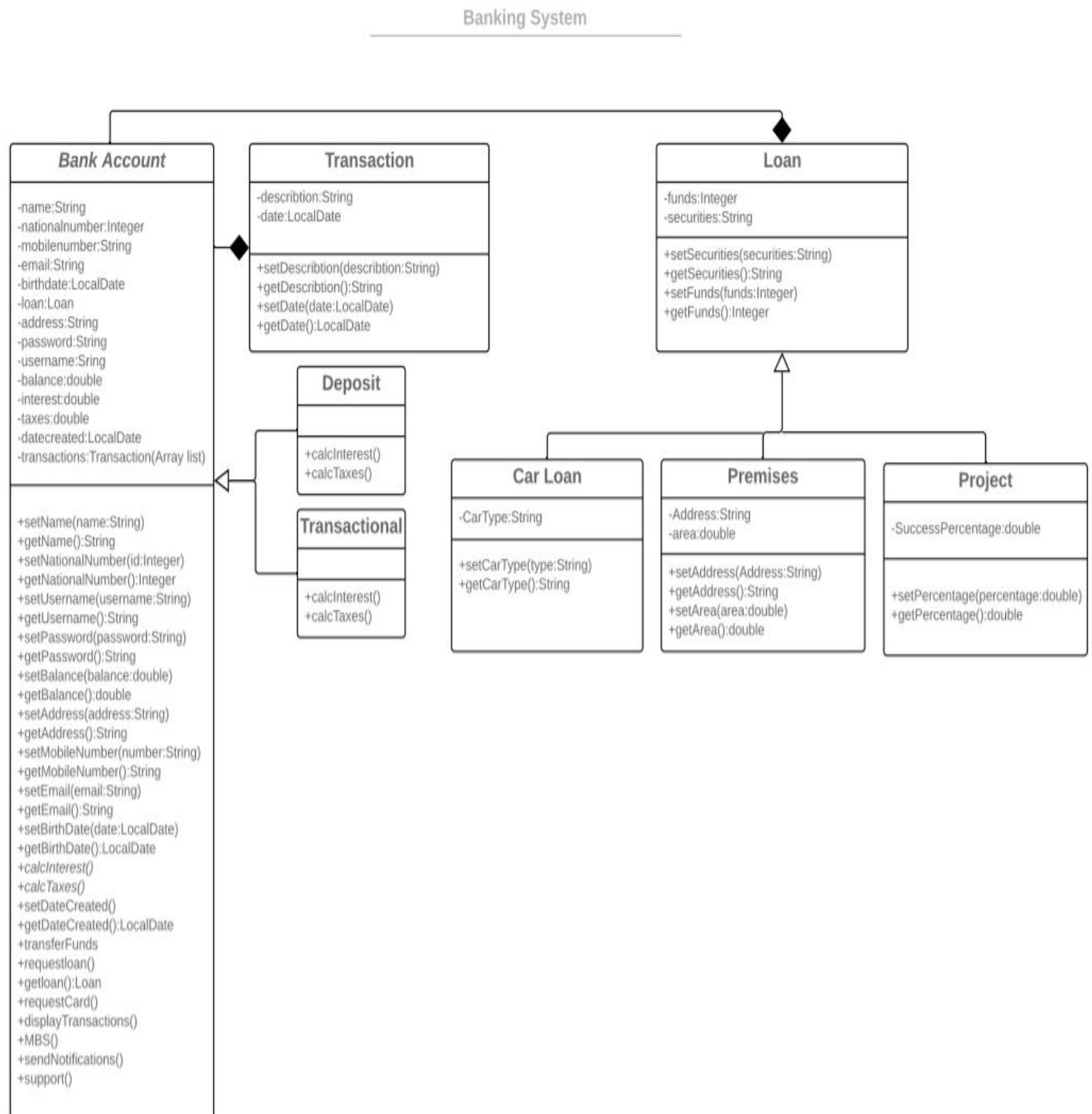
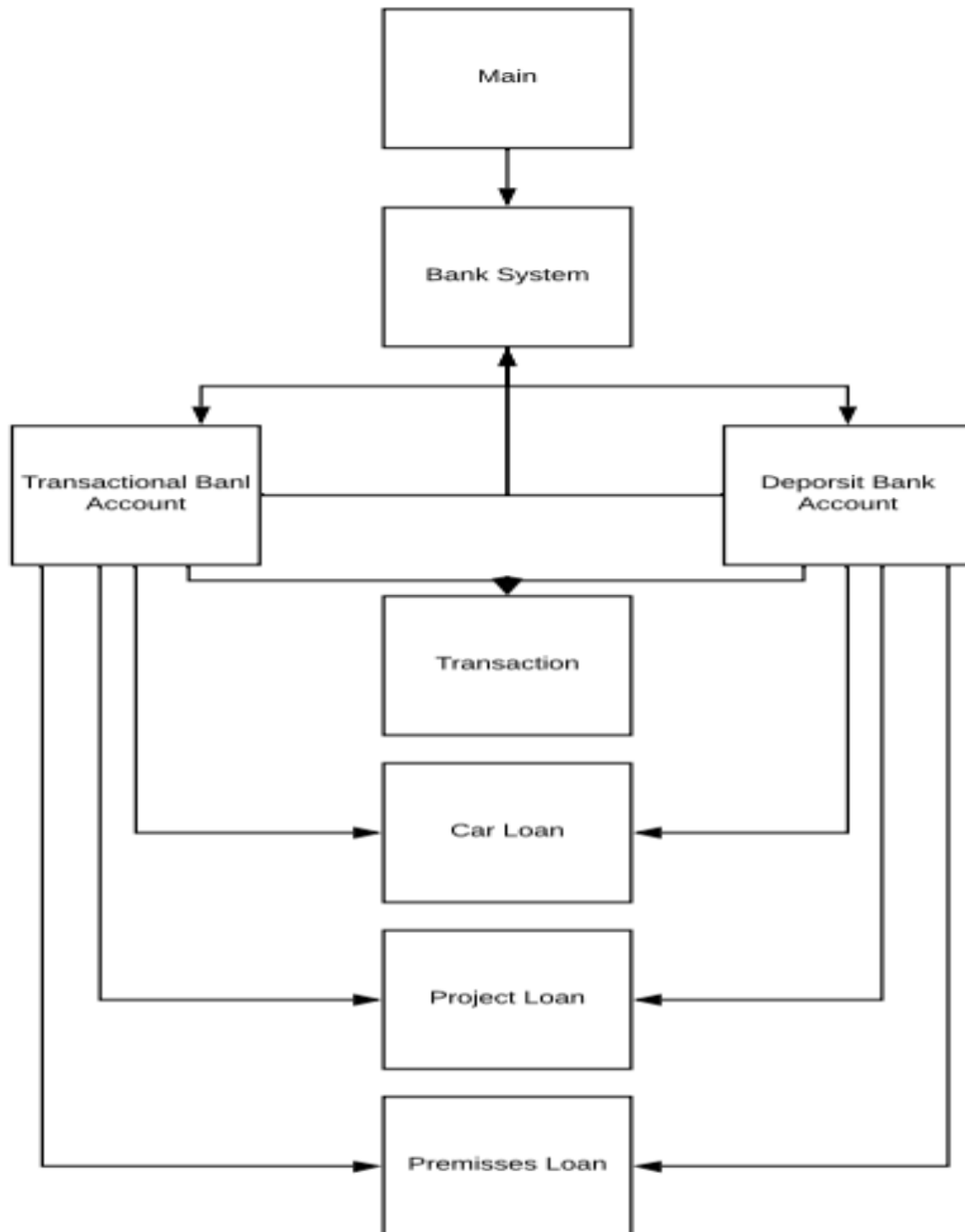


Figure 17: Detailed Class Diagram of Banking System.

12. USER INTERFACE DESIGN:

Mostly we used RGB : #197f63 color in buttons and headers ,which is a derivative of green palette as the logo is in green color, so that the theme of the software app is consistent. Moreover, for most buttons drop shadow effect was used to obtain a stylish button rather than the classic JavaFx control button. However, for the body of the app white color is used as it mixes well with RGB: #197f63.

The app has a fixed header of buttons that can be accessed by any user which are “Home”, “Contact”, “Location”, “Policy” where each opens a scene containing details of its content. Also , the “Home” scene contains 2 accessible buttons which are “Sign In” and “Sign Up”. However, there are buttons not accessible except by the signed in users which are “Check Balance”, “Request Card” , “Request Loans”, “View Transactions” and “Transfer Funds”. The user interaction with the state of the system is mostly throughout the prediscrbed buttons.

13. OBJECT-CLIENT DIAGRAM:*Figure 18: Object Client Diagram.*

To make KAGAMY NB we used waterfall model as we developed the system through waterfall model's phases which is:

- **Requirements:** The first phase involves understanding what need to be design and what is its function, purpose etc. Here, the specifications of the input and output or the final product are studied and marked.
- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The software code to be written in the next stage is created now.
- **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.
- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. The software designed, needs to go through constant software testing to find out if there are any flaw or errors. Testing is done so that the client does not face any problem during the installation of the software.
- **Maintenance:** This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. These modifications arise either due to change requests initiated by the customer, or defects uncovered during live use of the system. Client is provided with regular maintenance and support for the developed software.

Benefits we encountered for using waterfall model:

- Waterfall development allowed us to gain advantage for control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.
- The waterfall model progresses through easily understandable and explainable phases and thus it is easy to use.
- It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- In this model, phases are processed and completed one at a time and they do not overlap. Waterfall model works well for smaller projects where requirements are very well understood.

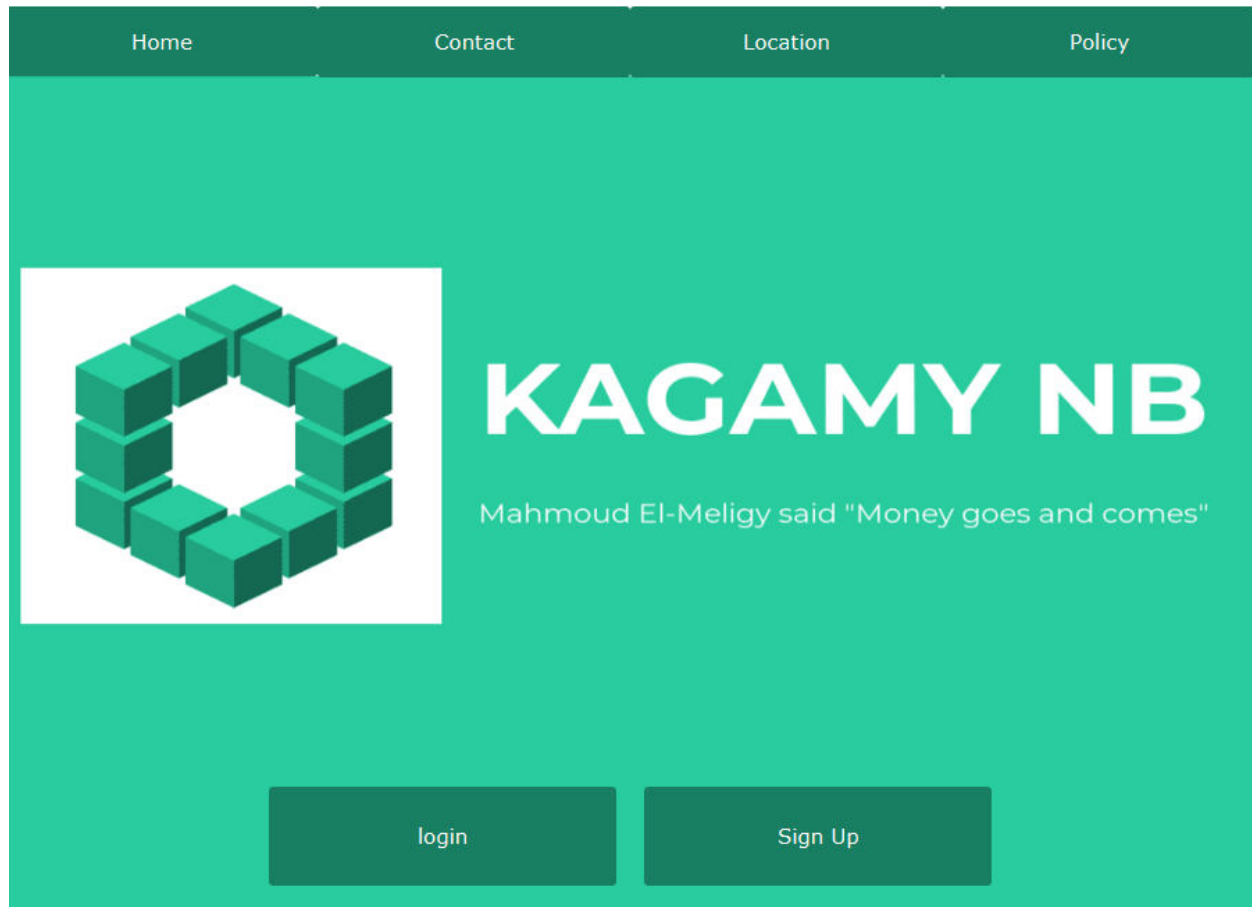
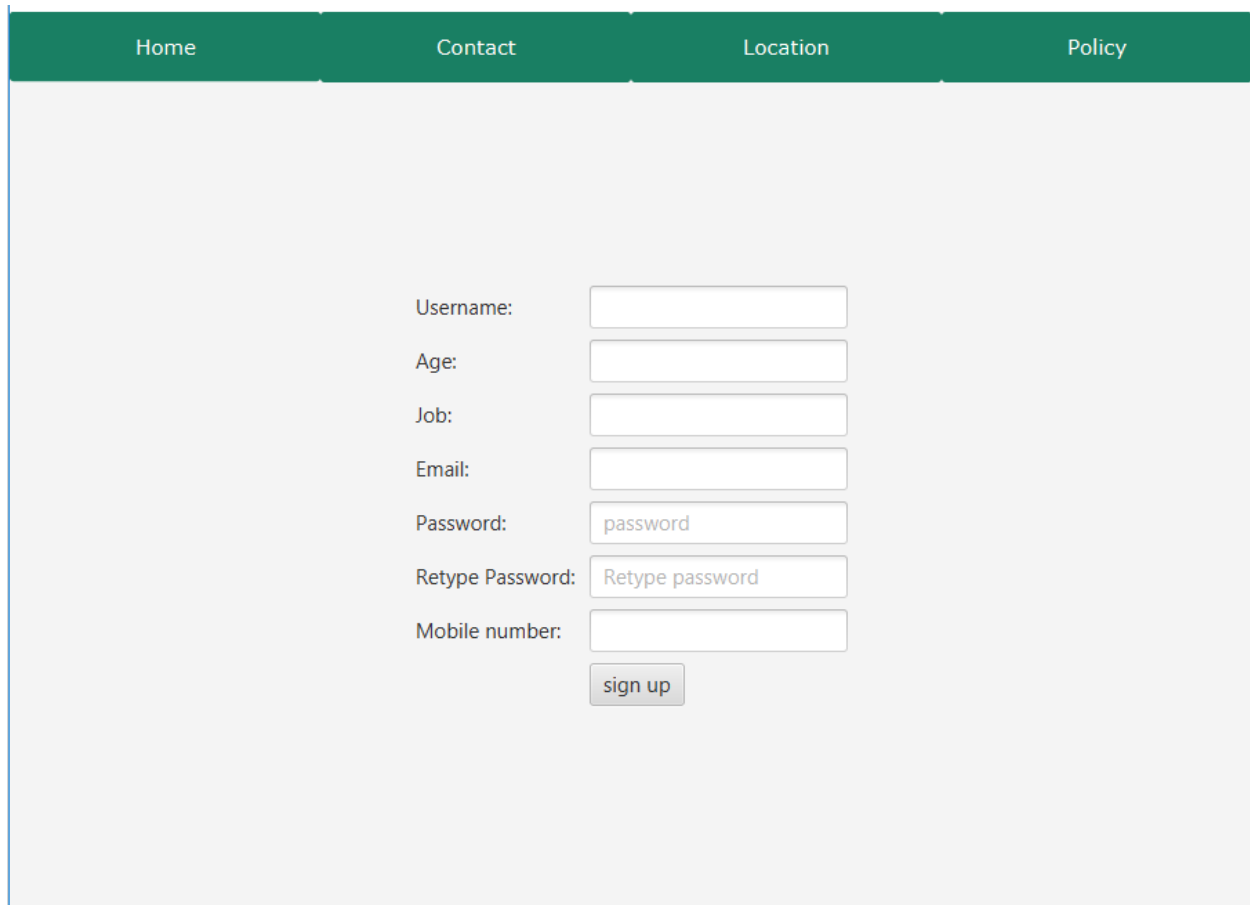
15. USER GUIDE:

Figure 20: Open Window.

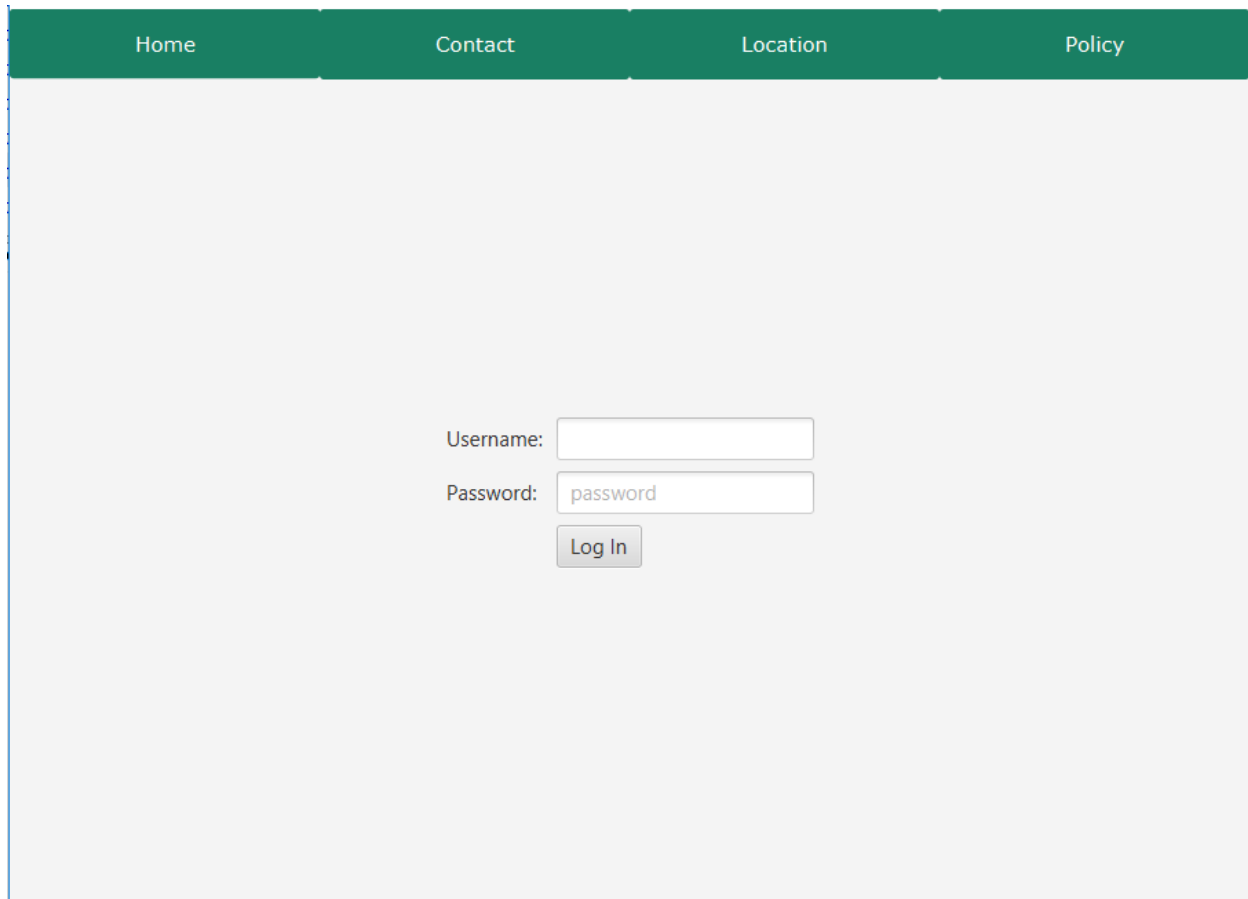
This is the window that will open when the Application starts. The user can access the Contact, Location, and Policy buttons even if he does not login or sign up, but he can not access the home button with all of its contents unless he sign up (if he does not have an account), or login (if he already has an account.)



The image shows a web interface for signing up. At the top, there is a dark green navigation bar with four tabs: 'Home', 'Contact', 'Location', and 'Policy'. Below this bar is a light gray main content area. In the center of this area is a sign-up form. The form consists of several labeled input fields stacked vertically: 'Username:', 'Age:', 'Job:', 'Email:', 'Password:', 'Retype Password:', and 'Mobile number:'. The 'Password' field contains the text 'password' and the 'Retype Password' field contains 'Retype password'. Below these fields is a gray button labeled 'sign up'.

Figure 21: Sign up.

The desired username is entered. In addition to the username, the age, the job, the email, the password, password confirmation and the mobile number are entered as details of the new account.



A screenshot of a web application's login page. At the top, there is a dark green navigation bar with four links: 'Home', 'Contact', 'Location', and 'Policy'. Below this bar is a large, light gray rectangular area. In the center of this area is a login form. The form consists of two text input fields: the first is labeled 'Username:' and is empty; the second is labeled 'Password:' and contains the text 'password'. Below these fields is a gray button with the text 'Log In'.

Figure 22: Login.

The username and password are entered in the fields. If it matches the username and password of an account in the database, the user has successfully logged in.

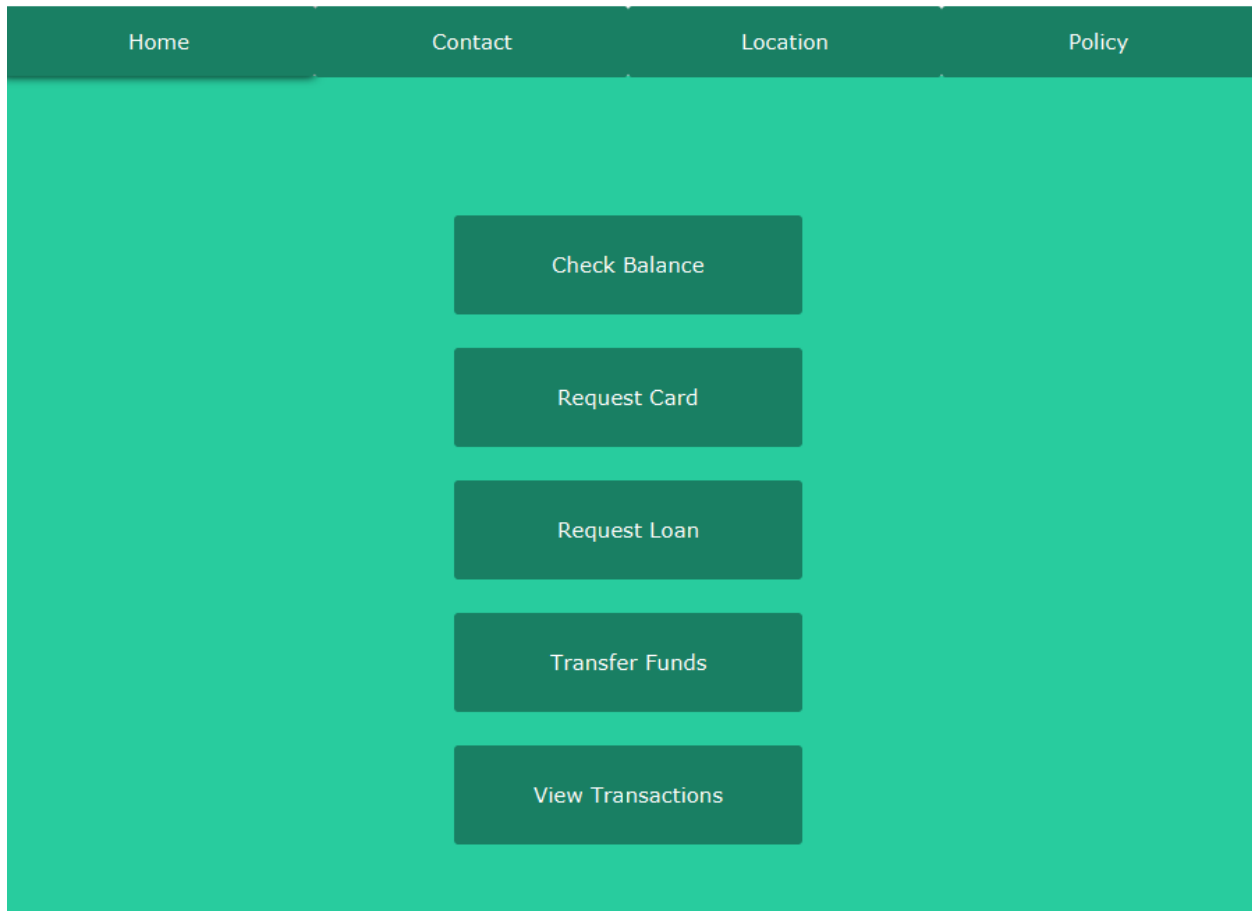
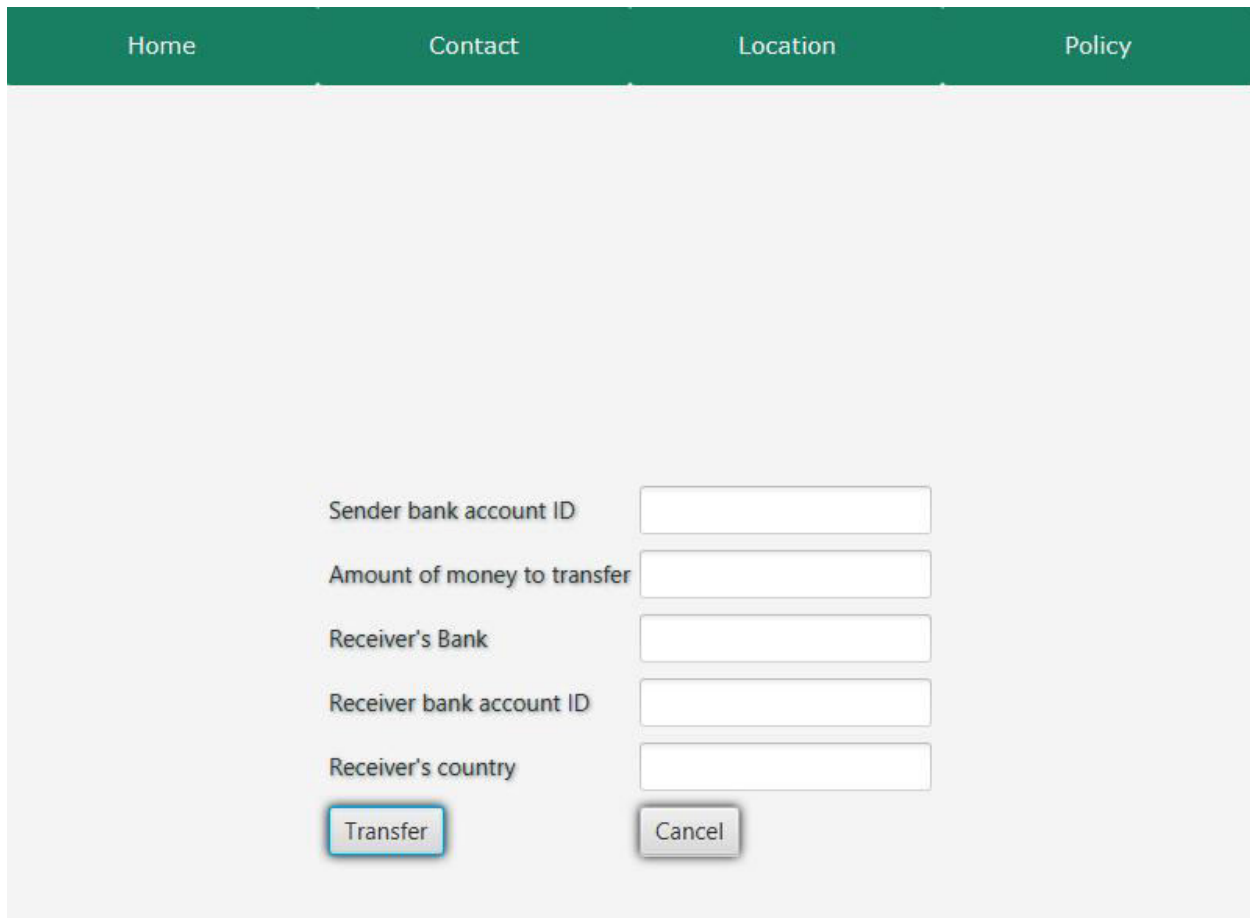


Figure 23: Functions.

After the user has successfully logged in, the following scene appears with 5 buttons; each with a unique function to be made (stated on each button). “Check Balance” button view the balance of the account, while “Request Card” shows the date of receiving the credit card.



Home Contact Location Policy

Sender bank account ID

Amount of money to transfer

Receiver's Bank

Receiver bank account ID

Receiver's country

Transfer Cancel

Figure 25: Transfer Funds.

When the user presses the Transfer funds button in the home page, he will see this window, and he should fill these labels (Sender bank account ID, amount of money to transfer, Receiver's bank, Receiver bank account ID, and Receiver's country) and then press the Transfer button.

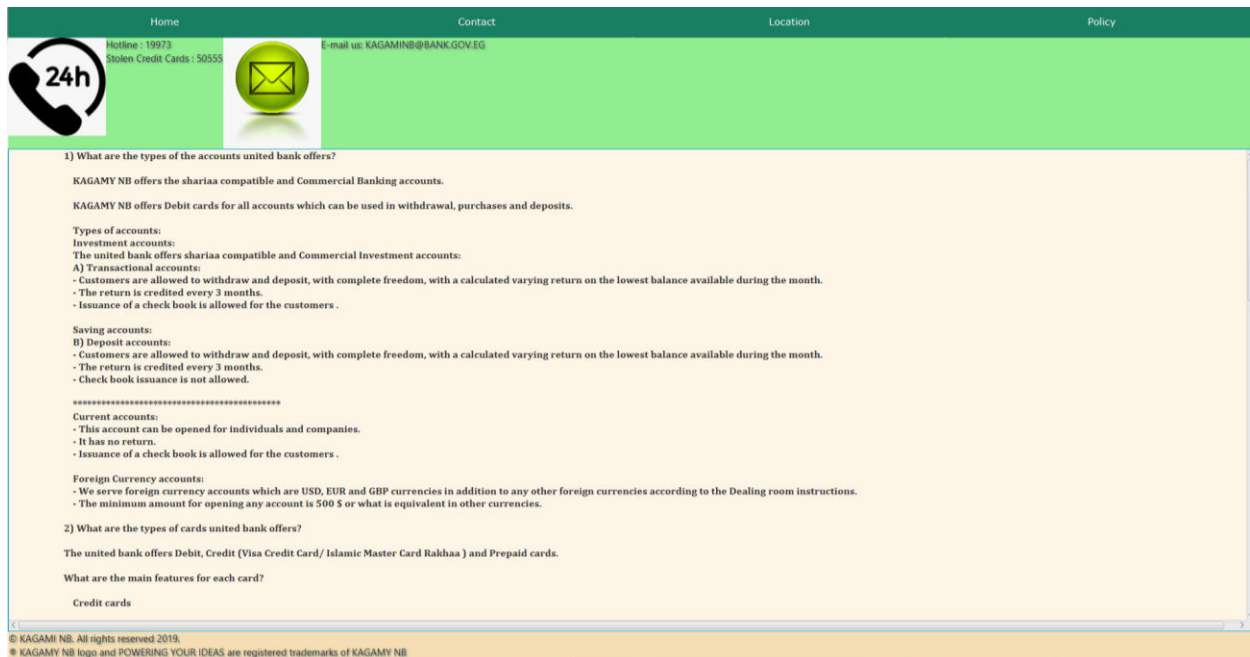


Figure 26: Contact.

When the user clicks the “Contact” button a window appears which helps the user by answering frequently asked questions (FAQ). The user can scroll through the FAQs easily. However, if the user doesn’t find answers he can call the number in the top of window or write an e-mail to the email to the right of the number.

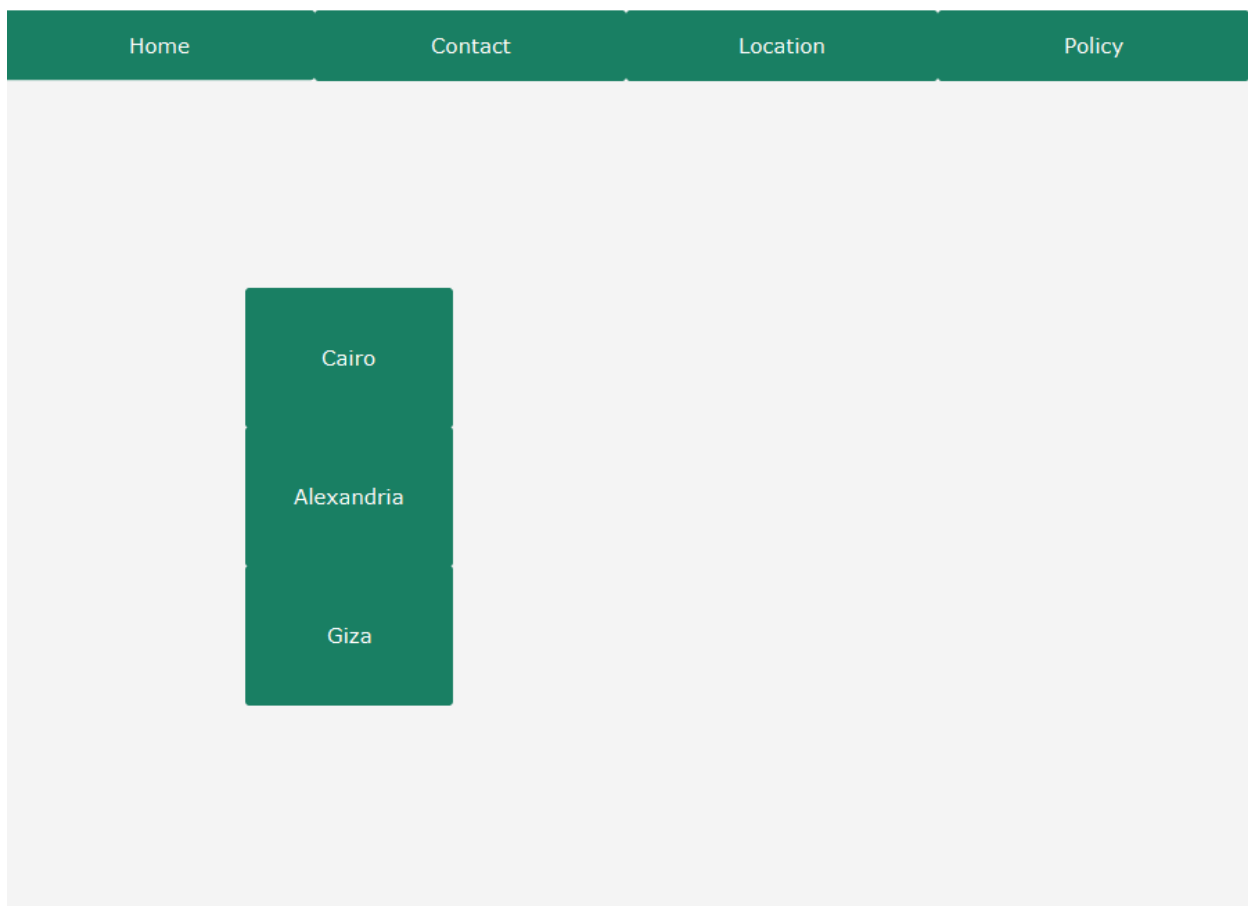


Figure 27: Location.

When the user clicks the “Location” button a window appears where the user gets to click any from 3 buttons to choose where is the governorate he would like to see the bank’s branches locations in.

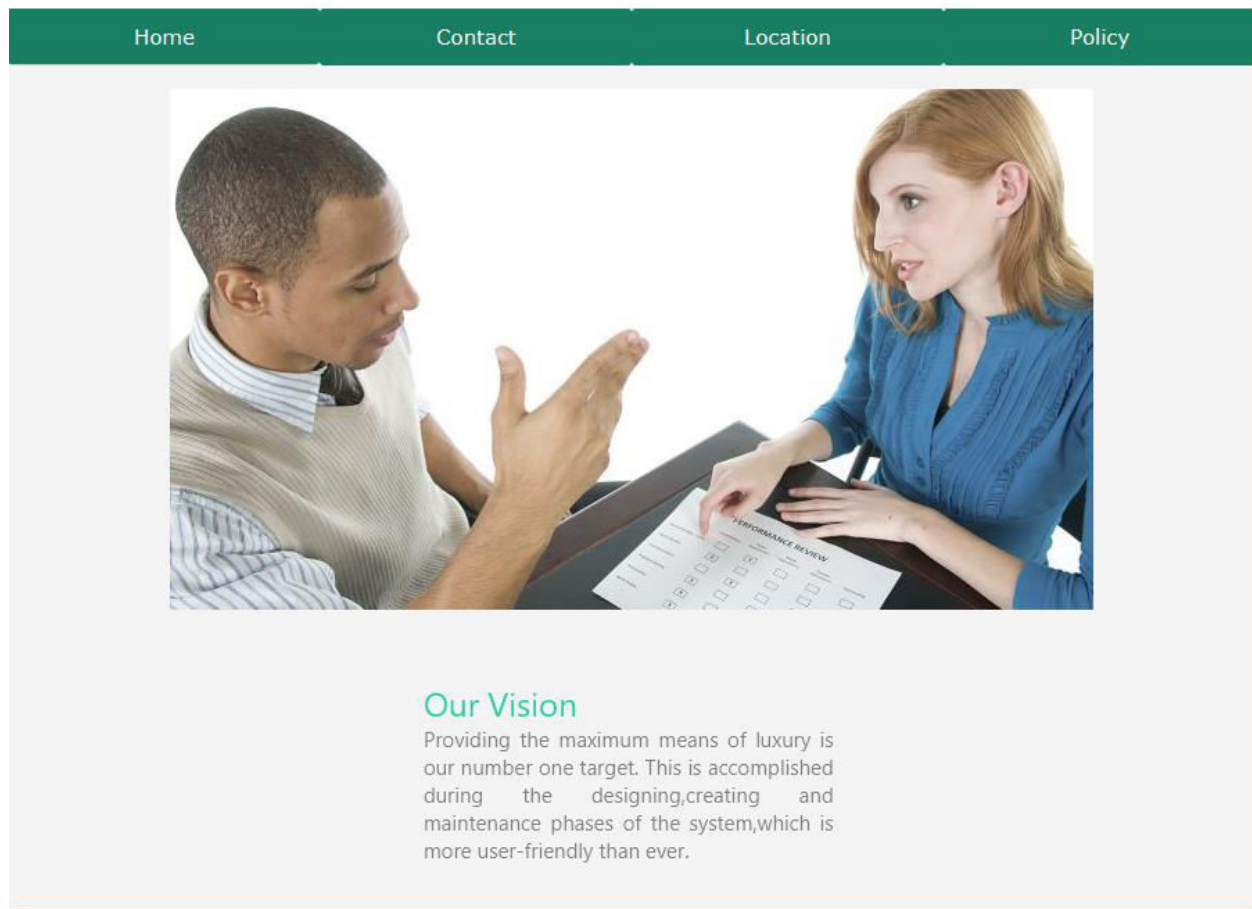


Figure 28: Policy.

When the user press the Policy button, this window will be displayed, so he will see the bank vision.