BANKING MANAGEMENT SYSTEM

SOFTWARE ENGINEERING PROJECT REPORT

B.Sc. (H) Computer Science

“Only Knowledge can provide salvation”

Submitted By-:

Karan Singh Manral	Aditya Raj	Anshul Bhandari	Harsh Kumar Singh
20035570045	20035570004	20035570020	20035570036

Department of Computer Science

Keshav Mahavidyalaya

University of Delhi

Acknowledgement

Apart from the efforts of team, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

The completion of any inter-disciplinary project depends upon cooperation, and combined efforts of several sources of knowledge.

We are eternally grateful to our teacher **Dr. Sumit Agarwal** for his even willingness to give us valuable advice and direction under whom we executed this project. His constant guidance and willingness to share his vast knowledge made us understand this project and its manifestation in great depths and helped us to complete the assigned tasks.

We are also thankful to all the people who helped in building of this project.

Karan Singh Manral (20035570045)

Aditya Raj (20035570004)

Anshul Bhandari (20035570020)

Harsh Kumar Singh (20035570036)

Certificate

This is to certify that Software Engineering project report entitled “Banking Management System” is the work carried out by **Karan Singh Manral, Anshul Bhandari, Aditya Raj, Harsh Kumar Singh,** students of BSc(H) Computer Science IV Semester, Keshav Mahavidyalaya, University of Delhi under the supervision of **Dr. Sumit Agarwal.**

This report has not been submitted to any other organization/institution for the award or any other degree/diploma.

Supervisor	**Teacher In Charge**	**Principal**
Dr. Sumit Agarwal	Ms. Maulein Pathak	Prof. Madhu Pruthi

List of Figures used in project

List of Tables used in project

Table of Contents

Problem Statement

A bank encompasses several branches and each branch further has many customers who hold one or more accounts in that branch. Bank offers five types of deposit accounts namely Savings, Current, fixed deposit, joint, minor. Any person who holds an account has either of these accounts. Any person if wants to become a new customer to the bank also has to choose among these types. New customer to the bank can either open an individual account, a joint account or a minor account (in case the applicant is under 18 years of age) as per his/her wish. Every Savings Bank Account whether minor/joint or individual has very basic features of cash deposit and cash withdrawal. There is no restriction on the number and the amount of deposits, however a mandatory uniquely identified PAN number is required for doing cash transactions exceeding Rs.49,999. The bank pays an interest annually on the savings lying in the savings bank account. It also provides the facility of fund transfer through which the money from one customer’s account is transferred to other account. The money transfer can be done within the same bank or with the account holder of any different bank however, transferring money to another bank’s account holder charges some minor penalty. Current Bank Account is another type of bank account having its own distinct features that are managed differently. It is mainly opened by businessmen to carry out their business deals promptly and smoothly. No restrictions are made on the number and the amount of withdrawals as long as the account holder has funds in his/her account. Bank pays no interest on the funds kept in the current account.

In fixed deposit bank accounts, whole deposit is made at once and it lasts for 15 days to 10 years with high rates of interests. No withdrawals are allowed however, the bank allows the customer to close the account before maturity in the case of emergency but with a certain deduction in rate of interest. The fixed deposit can be renewed after its maturity.

All these bank details and records can’t be managed manually. Therefore, a need for software is realized that can manage all the details at both customer and bank authorities’ side. The main aim of the software is to automate all the functioning of bank both at the customer's and bank authorities’ side. Thus, Banking Management System is introduced to resolve this issue.

Two views will be managed having their own different functioning; one for the customer and other one for the bank authorities.

Among the bank authorities, different persons will manage data and records for different bank related works with each person having a different view of the bank database. For each work, the bank has different manager who has access only to the details that are related to his section of work. The bank manager has all the details and has access to all the data and records managed by different section managers.

For every cash deposit and cash withdrawal made by the customer holding savings bank account, the bank authority respective to this section will update the bank database and the customer’s account database by entering all the transaction details(customer name, account number, phone number, deposited/withdrawal amount and PAN number in case of transaction is of more than Rs. 49,999). All these type of bank transactions details are updated in bank database both for the customer and bank authorities and can be found in the “Account Details” section of the database for that particular account. All the details related to the transaction are automatically updated to the bank’s database with the help of the account number.

For fixed deposit account, the bank offers a rate of interest on every fixed deposit made by the customer and that rate of interest varies accordingly with the time period for which FD is made. If the fixed deposit account is opened from the main account balance, the amount is deducted from the bank balance and new fixed deposit account is opened for every fixed deposit made by the customer. The details of amount deduction is updated in “account details” section of database of particular account, and the FD details are updated in “FIXED DEPOSIT” section of bank database by the bank authority seeking this section by giving all the details of customer and fixed deposit request made by that customer. Also the software will enable the bank authorities to update/modify customer details or information relates to customer account whenever any customer requests for the same. The customer has to fill KYC form of the bank along with all the ID proofs and other related documents to provide the bank with all his details if he wants to make modifications, bank authorities will then verify the documents and update the desirable changes to the “customer’s details” section of the bank database.

1 Introduction

1.1 Purpose

The main purpose of this software is to simplify the tedious task of banking by providing this service in a user-friendly environment. It also aims at increasing the efficiency and reducing the drawbacks of existing manual banking process, thus making it more convenient for the customers to do banking as when they require.

This project is supported by a well-designed DBMS in which customer's account information is integrated together. A friendly UI is also provided so that the requests made by the user give correct results by accessing the information stored in the database.

1.2 Scope

This software allows the users to apply for new account and to update the previous account. It gives them the facility to deposit money in the account or withdraw money from the account. It also give the facility to apply for loan.

1.3 Definitions

- BMS - Banking Management System
- UI – User Interface
- DBMS – Database Management System

1.4 Overview

The rest of the document deals with all the main features of this software. It not only describes various functions but also gives details about how these functions are related to each other. Apart from the data flow diagrams, the document also contains size and cost estimates for developing this system. Various risks associated with the system have also been mentioned along with the risk management sheets. The timeline chart describing how the entire project was scheduled has been attached followed by the use cases, sequential design and architectural design of the software. At the end a pseudo code for the "Transaction System Module" has been provided. A flow graph has been generated from code corresponding to this module, cyclomatic complexity has been computed, boundary value analysis is done, equivalence case testing is applied and test cases that were used to test the system have also been mentioned.

1.5 Process Model

We choose the **Waterfall Model** because of the following reasons: -

- The requirements are well stated and understood before in hand.
- The requirements are fixed and work can proceed in linear manner.
- In this model we have to complete one stage before proceeding to next. So, we have clearly defined stages and well understood milestones.
- The Waterfall model provides a structured approach.
- It is relatively simple and easier to understand approach as compared to other models.
- The advancement in program does not need to be checked upon by the customer during the process. So, this model is perfect for our project.

2 Software Requirement Specification

2.1 Description

2.1.1 Product Perspective

The manual banking system suffers from following drawbacks: -

- The existing system involves a lot of paper work and manual calculation which has led to inconsistency and inaccuracy in the maintenance of data.
- The data, which is stored on the paper only, may be lost, stolen or destroyed due to natural calamity like fire, water etc.
- The existing system is sluggish and time-consuming causing inconvenience to customers and the banking staff.
- Since the number of customers have drastically increased therefore maintaining and retrieving detailed record of customer is extremely difficult.
- A bank has many branches in the country, an absence of a link between these branches lead to lack of coordination and communication.

Hence the banking management system is proposed with the following Product perspective: -

- The computerization of the banking system will reduce a lot of paperwork and hence the load on the bank staff.
- The machine performs all calculations. Hence chances of error are almost nil.
- The customer details, balance can easily be retrieved and any required addition, deletion or updation can be performed.
- The system provides for user-ID validation, hence unauthorized access is prevented.

2.1.2 Product Functions

The “BMS” software is an android application. There are various user interfaces related with this software. These interfaces help the user to interact with the software and provide the necessary information for net banking.

The entire functionality of this software can be subdivided into fields/modules. The names of the fields involved in the banking management system are: -

1. Customer Management
2. Loan System
3. Transaction System

MODULE 1: CUSTOMER MANAGEMENT

Using this module, the user first provides his details like his name, father name, address, phone no., email etc. and can open a new account or can update an existing account.

MODULE 2: LOAN SYSTEM

This module is for customer to apply for loan by providing required documents and details like time period of loan, amount of loan and get the detailed description of EMIs.

MODULE 3: TRANSACTION SYSTEM

This module allows the customers to deposit and withdraw money from their account using some private info like pin, signature or OTP.

Various interfaces involved in BMS have been designed and implemented using Android Java. All the information about customer is maintained in SQLite files which act as database for the software.

2.1.3 User Characteristics

1. The user must be comfortable working with English Language.
2. The user should have basic knowledge of android mobile and internet.

2.1.4 General Constraints

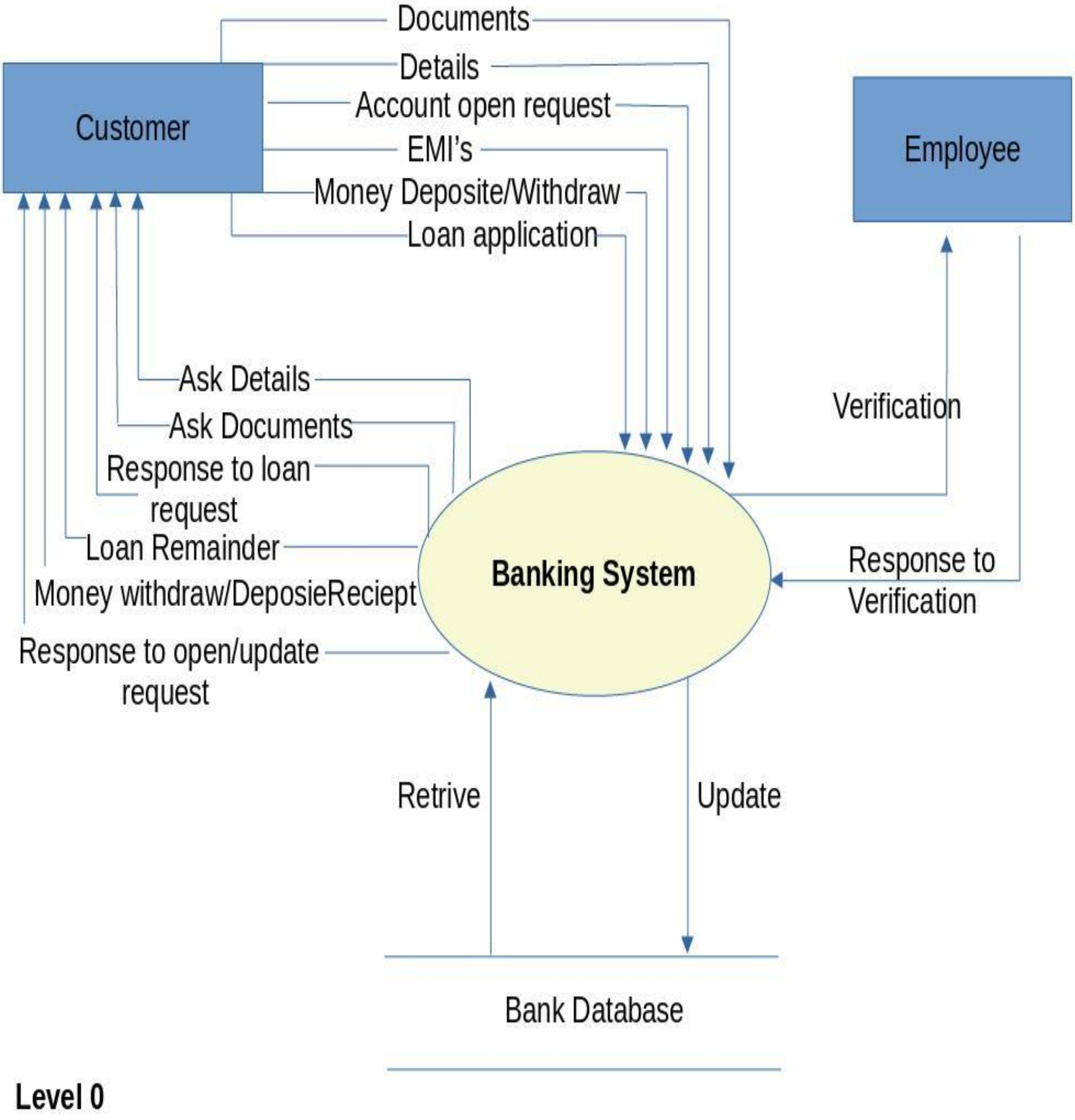
The database of the software is not fully accessible to every user of the software. Making changes in the database requires specific permissions that have been given to certain specific individuals i.e., banking staff.

2.1.5 Assumptions and Dependencies

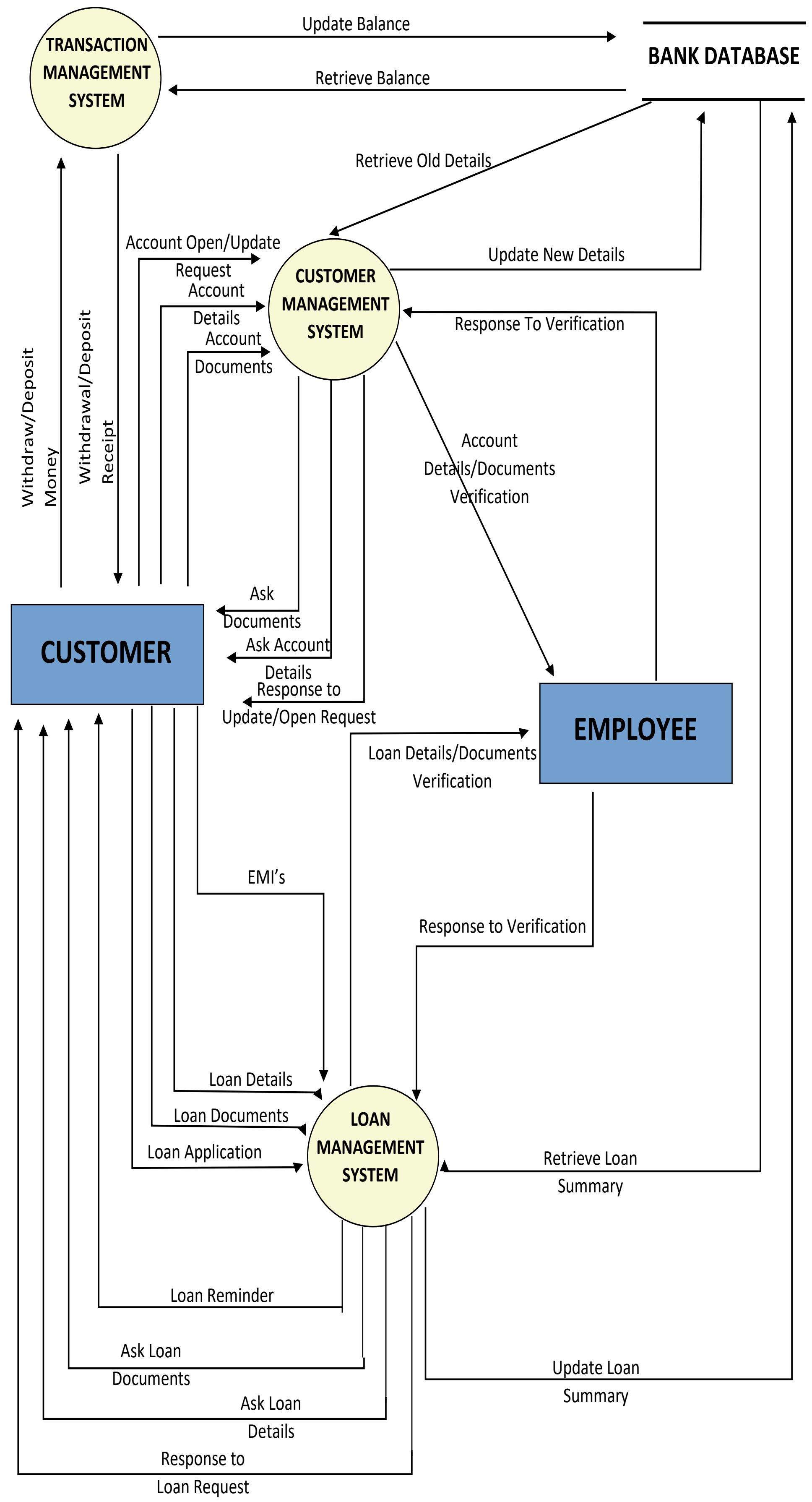
The BMS is assumed to be compatible with the android systems on which it has been loaded for customer use.

2.2 Data-Flow Diagram

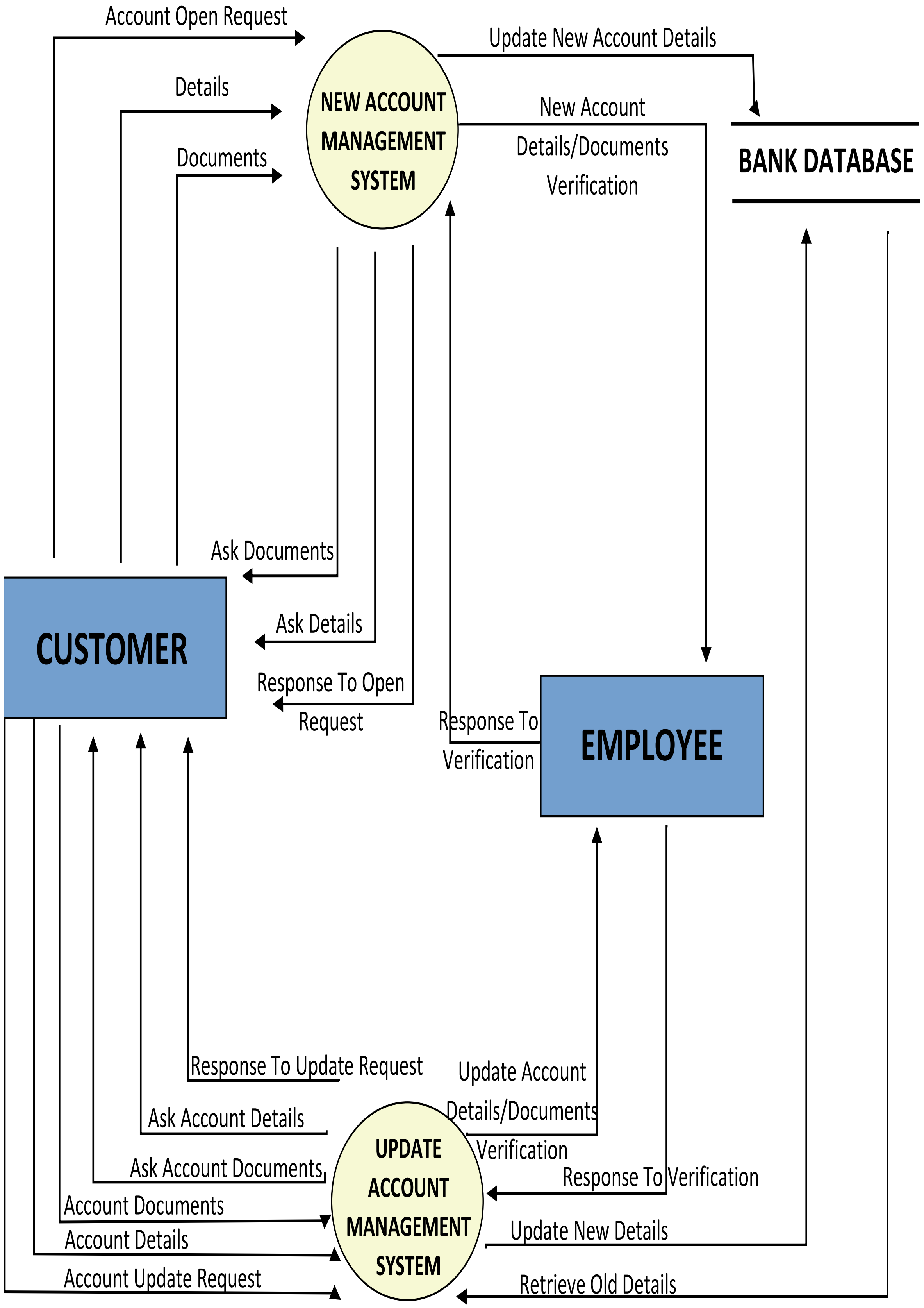
2.2.1 DFD Level 0



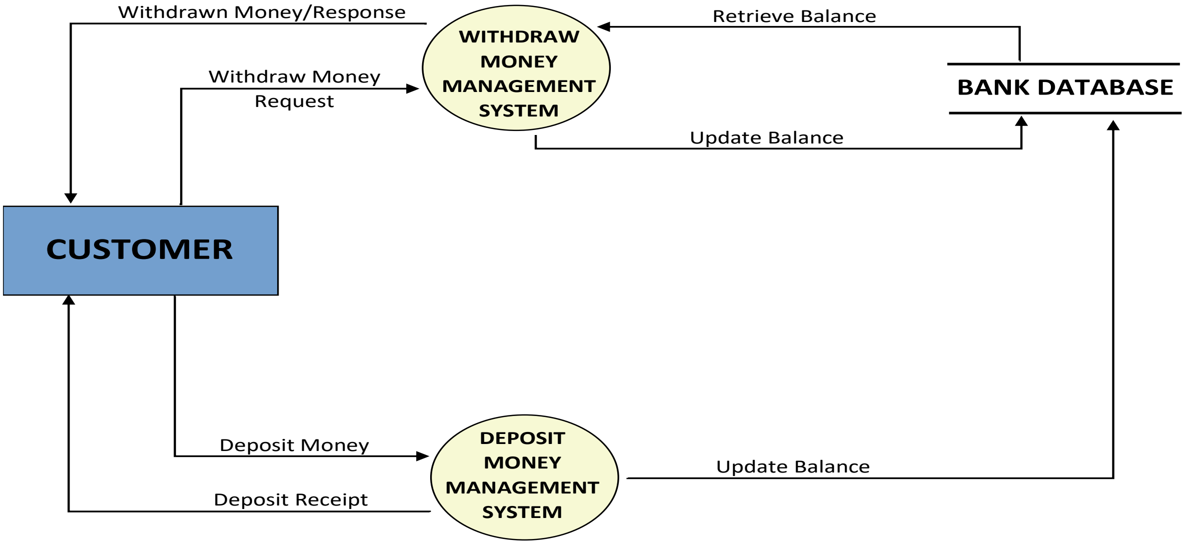
2.2.2 DFD Level 1



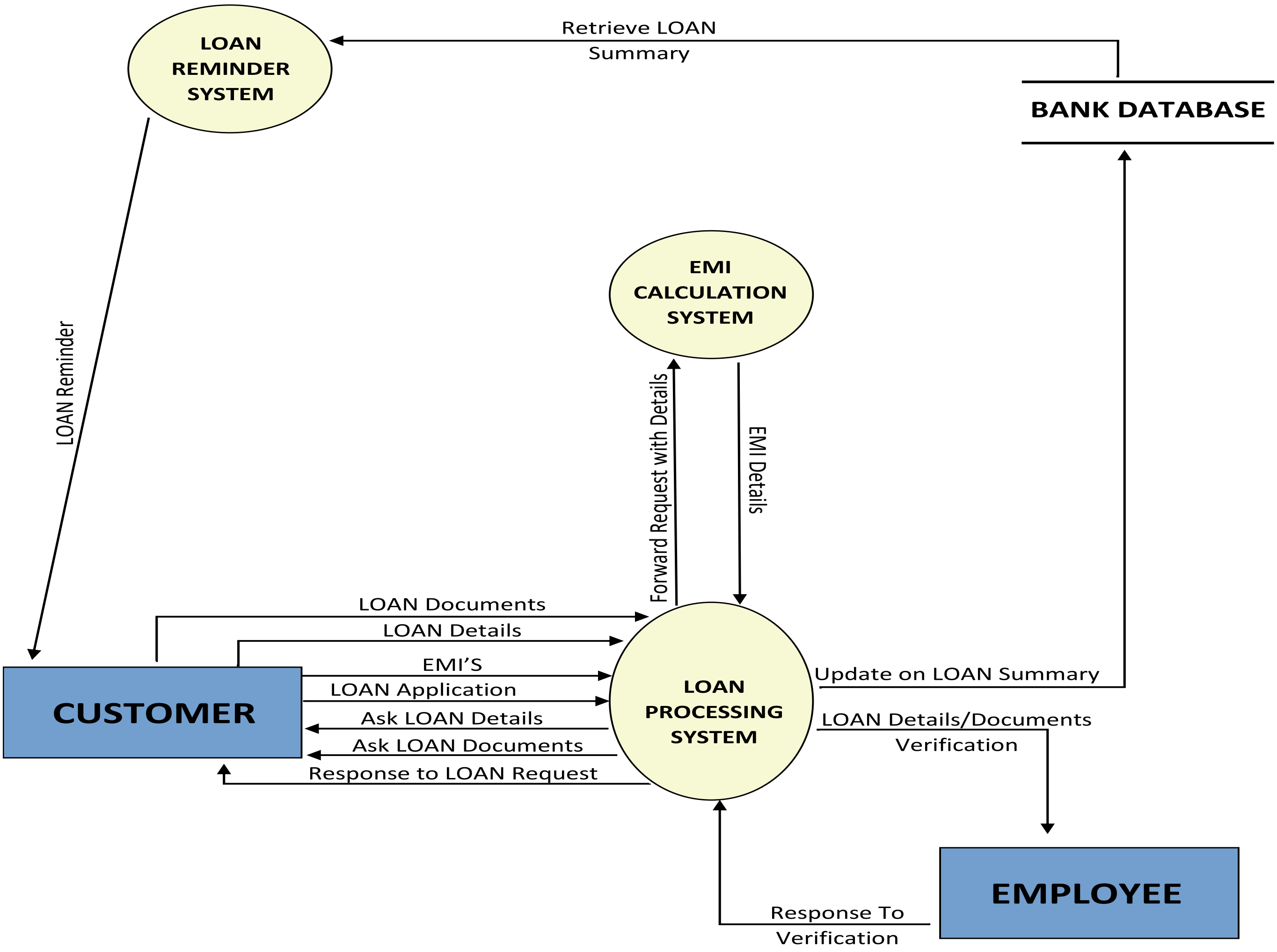
2.2.3 DFD Level 2 Customer Management



2.2.4 DFD Level 2 Transaction System



2.2.5 DFD Level 2 Loan System



2.2.6 Data Dictionary

The data dictionary, or Meta data repository, as defined in the IBM Dictionary of Computing, is a “centralized repository of information about data such as meaning, relationships to other data, origin, usage and format.”

Our Banking Management System data dictionary can be of following type: -

Data	Description
Documents	Account documents OR Loan documents
Details	Account details OR Loan details
Account documents	ID Proof, Residence Proof, Date of Birth Proof
Loan documents	ID Proof, Residence Proof, Birth Proof, Salary slip, PAN Card, Loan application
Account details	Name, Address, Phone Number, Account number, Signature
Loan details	Loan time period, Loan amount, Loan type
Name	First name, Middle name, Last name
Address	House No., Village/City name, State name, PIN code
Phone number	Digit + Digit + Digit + Digit + Digit + Digit + Digit + Digit + Digit + Digit
Account number	Digit + Digit + Digit + Digit + Digit + Digit + Digit + Digit + Digit + Digit + Digit + Digit
Deposit Receipt	Date of deposit, Amount deposited
Response	Approval | Rejection
Loan Reminder	Reminder message, Last date to pay EMI

Account Documents- Documents related to account of a customer in our bank in order to verify customer’s identity such as passbook, pan card, residence proof, Aadhaar card, etc.

Loan Documents- Documents related to loan that a customer has taken from our bank such as pan card details, residence proof, Aadhaar card, salary details, EMI’s etc.

Account Details- Details of Customer related to his account in our bank such as his name, date of birth, address, phone number etc.

Loan Details- Details related to the loan that customer has taken from bank such as loan type, loan amount, EMI’s, Loan Time Period.

Account Number- Unique 12-digit number given to each account opened in our bank.

Deposit Receipt- Receipt of the amount user has deposited in his account.

Response- Response given by our employee/system to customer for verification of validity of documents/details provided by customer.

Loan Reminder- Reminder given to customer by our system for paying due amount of the Loan/EMI’s.

3 Project Management

3.1 Size and Cost Estimation

3.1.1 Complexity Adjustment Table

Sr No	Questions	Grade Value
1	Does the system require reliable backup and recovery?	5
2	Are specialized data communications required to transfer information to or from the application?	3
3	Are there distributed processing functions?	3
4	Is performance critical?	0
5	Will the system run in an existing, heavily utilized operational environment?	5
6	Does the system require on-line data entry?	5
7	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	5
8	Are the ILFs updated online?	5
9	Are the inputs, outputs, files, or inquiries complex?	2
10	Is the internal processing complex?	2
11	Is the code designed to be reusable?	3
12	Are conversions and installations included in the design?	0
13	Is the system designed for multiple installations in different organizations?	5
14	Is the application design to facilitate change and for ease of use by the user?	5

VALUE ADJUSTMENT FACTORS, Σfi= 48

Functional Point Table

Information	Estimated Count			Weighing Factor			Weighing Count
	Simple	Average	Complex	Simple	Average	Complex	
External Inputs	3	4	2	3	4	6	37
External Outputs	1	4	1	4	5	7	31
External Enquiries	0	3	0	3	4	6	12
Internal Logical Files	0	1	0	7	10	15	10
External Interface Files	0	0	0	5	7	10	0

Total FP=90

Thus, the UNADJUSTED FUNCTIONAL POINTS (UFP) ARE:

UFP = Total I/P + Total O/P + Total ILF + Total EIF + Total Enquiries

= 37+31+12+10+0
= **90**

Now, Complexity Adjustment Factor (CAF) is :

CAF =0.65+(0.01* Σfi)
=0.65+(0.01*48)
=1.13

So the Functional Points (FP) is:

FP = UFP * CAF

FP= 101.7

3.1.2 Size Estimation

Assuming the average productivity for this kind of system is 6.5 FP/pm i.e. 6.5 Functional Points per month then the estimates are:

Size or Efforts = Total FP’s / Average Productivity

=101.70/6.5
=15.65 person per month i.e. 16 person per month

3.1.3 Cost Estimation

Considering the labor rate=Rs 25000/pm
Total Cost of the Project = Total Efforts * Labor Rate

=16*25000
=Rs. 4,00,000

3.2 Risk Table

Risks	Category	Probability	Impact	Risk Exposure RE=Probability*Impact
Quality not maintained	DE	60%	0.75	0.450
Requirements not properly documented and understood	CU	50%	0.25	0.125
Delivery deadline will be tightened	BU	40%	0.50	0.200
Lack of skill	ST	40%	0.50	0.200
Building the wrong product	CU	20%	0.25	0.050

0.25 – Catastrophic	PD - Process Definition
0.50 – Critical	ST - Staff-size and experience
0.75 – Marginal	DE - Development Environment
1.00 – Negligible	CU - Customer Characteristics
	BU - Business Impact
	PS - Product Size

3.2.1 Risk Information Sheets

Risk Information Sheet

Risk ID: 1	Date: 16/01/2022	Prob: 60%	Impact: Marginal

Description:

- Many business owners significantly underestimate the value of good quality software. Believe it or not, bad quality software is one of the top reasons why businesses fail.
- Software quality is the most essential part. Future of the organization depends on the quality of software provided by them.
- Having bad software quality can lead to users' dissatisfaction.

Context/Refinement:

- The head of the team or the developer might not be aware of the industry or domain he's writing software for which can lead to poor technical requirements compliance which automatically results in bad software.
- Inexperienced software engineering team would lead to unreliable and unstable software that would be full of technical bugs.

Mitigation/Monitoring:

- Training of Software Engineering Team
- Test early and Test often with Automation.
- Implement quality controls from the beginning at every step.
- Plan for a changeable environment.
- Echo the importance of quality assurance through the entire software development process.
- Encourage innovations and have good communication among employee, core team and customer.

Management/Contingency plan/Trigger:

- Should this occur, the organization would thoroughly test the software again and provide enough time and budget to team for fixing issues.

Current Status:

- Mitigation steps initiated.

Originator:	Assigned:

<table>
<tr><th colspan="4">Risk Information Sheet</th></tr>
<tr><td>Risk ID: 2</td><td>Date: 22/01/2022</td><td>Prob: 50%</td><td>Impact: Catastrophic</td></tr>
<tr><td colspan="4"><b>Description:</b>
• Many business owners significantly underestimate the value of documentation. Believe it or not, bad documentation is one of the top reasons why businesses fail. When defining the scope of software development, documentation becomes increasingly essential.
• Documentation is the workspace for the entire project, its core so to speak, and the link between the developers and the end-users.
• The success of any project and the delivery of its software capabilities become impossible without clear descriptions and accurate documentation.
• Mistakes in the documentation that ultimately lead to time loss, chaos, and which eventually leads to project and business failure. This negatively affects both the clients and the contractors.</td></tr>
<tr><td colspan="4"><b>Context/Refinement:</b>
• Operating system of different user mismatches.
• Differing technical abilities of different users.
• Lack of understanding of the specific information that needs to be included for coding purposes.</td></tr>
<tr><td colspan="4"><b>Mitigation/Monitoring:</b>
• In order to prevent this from happening, members who are in charge of developing the documentation will keep in contact with each developer on the team.
• Meetings will be held routinely to offer documentation suggestions and topics.
• Any topic deemed missing by a particular developer will be discussed and it will be decided whether or not to add that particular topic to the documentation.
• In addition, beta testers will be questioned about their opinion of the documentation.</td></tr>
<tr><td colspan="4"><b>Management/Contingency plan/Trigger:</b>
• Should this occur, the organization would call a meeting and discuss the addition of new topics, or removal of unnecessary topics into the documentation.</td></tr>
<tr><td colspan="4">Current Status:
• Mitigation steps initiated.</td></tr>
<tr><td colspan="2">Originator:</td><td colspan="2">Assigned:</td></tr>
</table>

Risk Information Sheet

Risk ID: 3	Date: 24/01/2022	Prob: 40%	Impact: Critical

Description:

- Tightening of deadline could lead to stress on employees and eventually employees leaving job.
- It can lead to a poor-quality project.
- Due to it, sometimes testing is not done up to mark.

Context/Refinement:

- Inexperienced staff can lead to delay in project.
- Old technology can be a cause for delay in project delivery.

Mitigation/Monitoring:

- Let programmers designate their own terms and reduce pressure on them.
- Watch over terms of validity.
- Adjust understanding between software engineering team, management and customer.

Management/Contingency plan/Trigger:

- Should this occur, the company will hold meeting with customers and try to negotiate some more time for delivery of system.

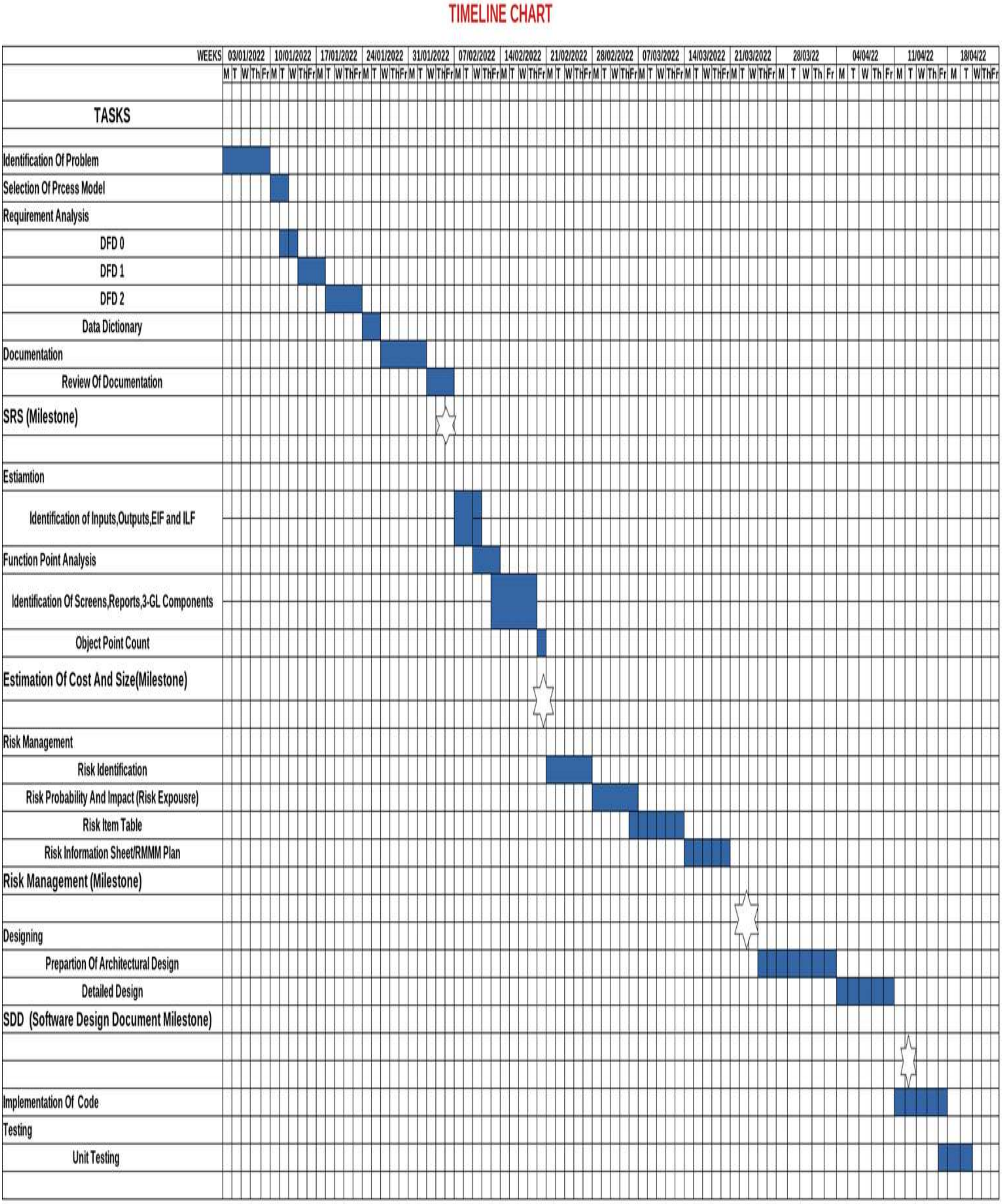
Current Status:

- Mitigation steps initiated.

Originator:	Assigned:

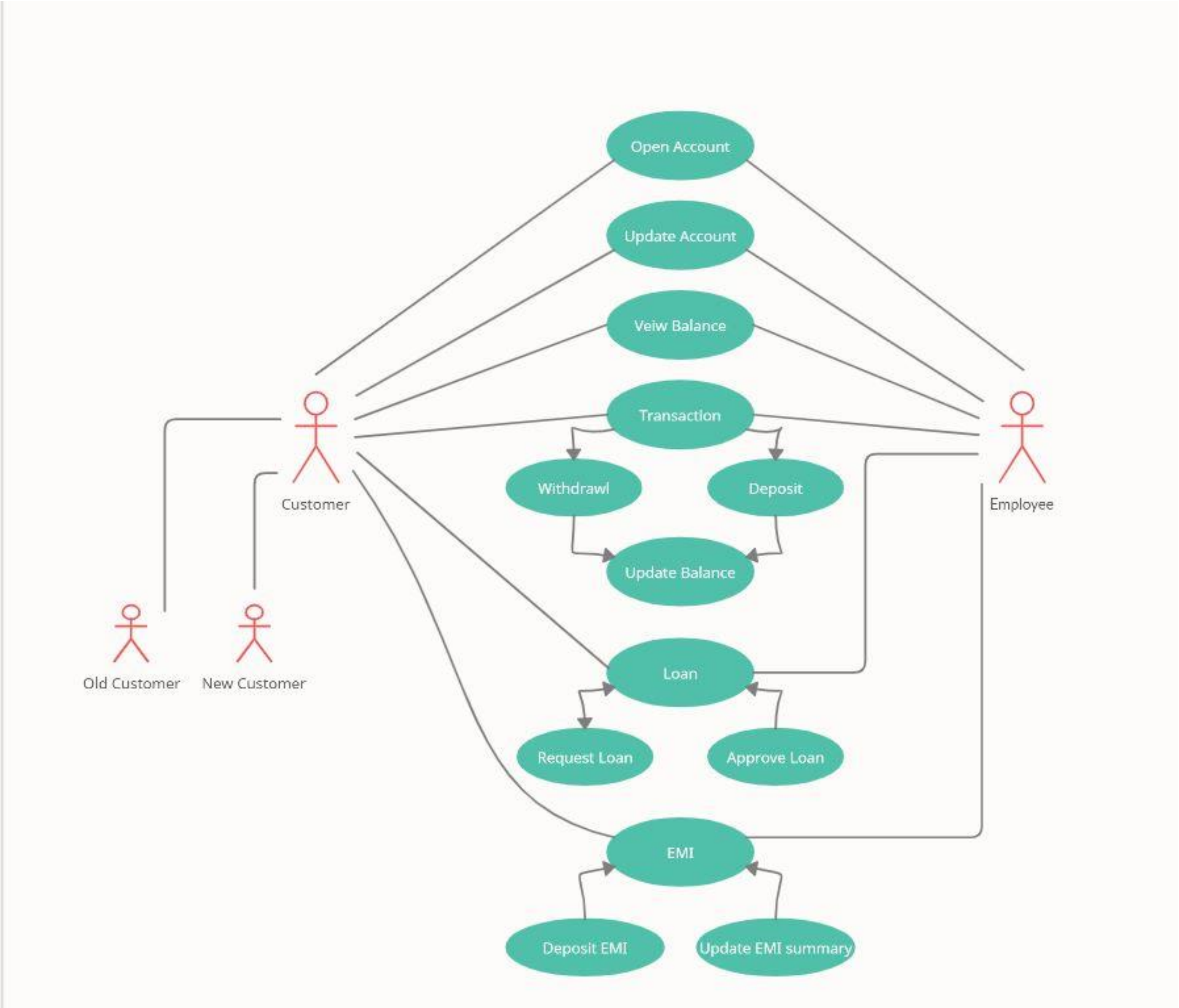
Risk Information Sheet			
Risk Id- 4	**Date-**23/01/2022	**Prob -** 40%	**Impact-** Critical
Description- • Lack of Skills can cause delay in delivery of project. • It could lead to a poor quality software.			
Context- • Shortage of Software Engineers with proper knowledge. • Organization is not willing to pay desired salary to employee with proper skills. • Relatively New Staff			
Mitigation- • We can resolve this by seeking the help of senior developer and increasing our efforts in improving our skills. • We can also introduce skill training/workshops for our employees to improve their skills.			
Management/Contingency plan/Trigger: • Risk exposure is estimated to be 0.200. So, allocating extra funds within project contingency cost and allocating staff accordingly.			
Current status- • Mitigation steps initiated.			
Originator:		**Assigned:**	

3.3 Timeline Chart



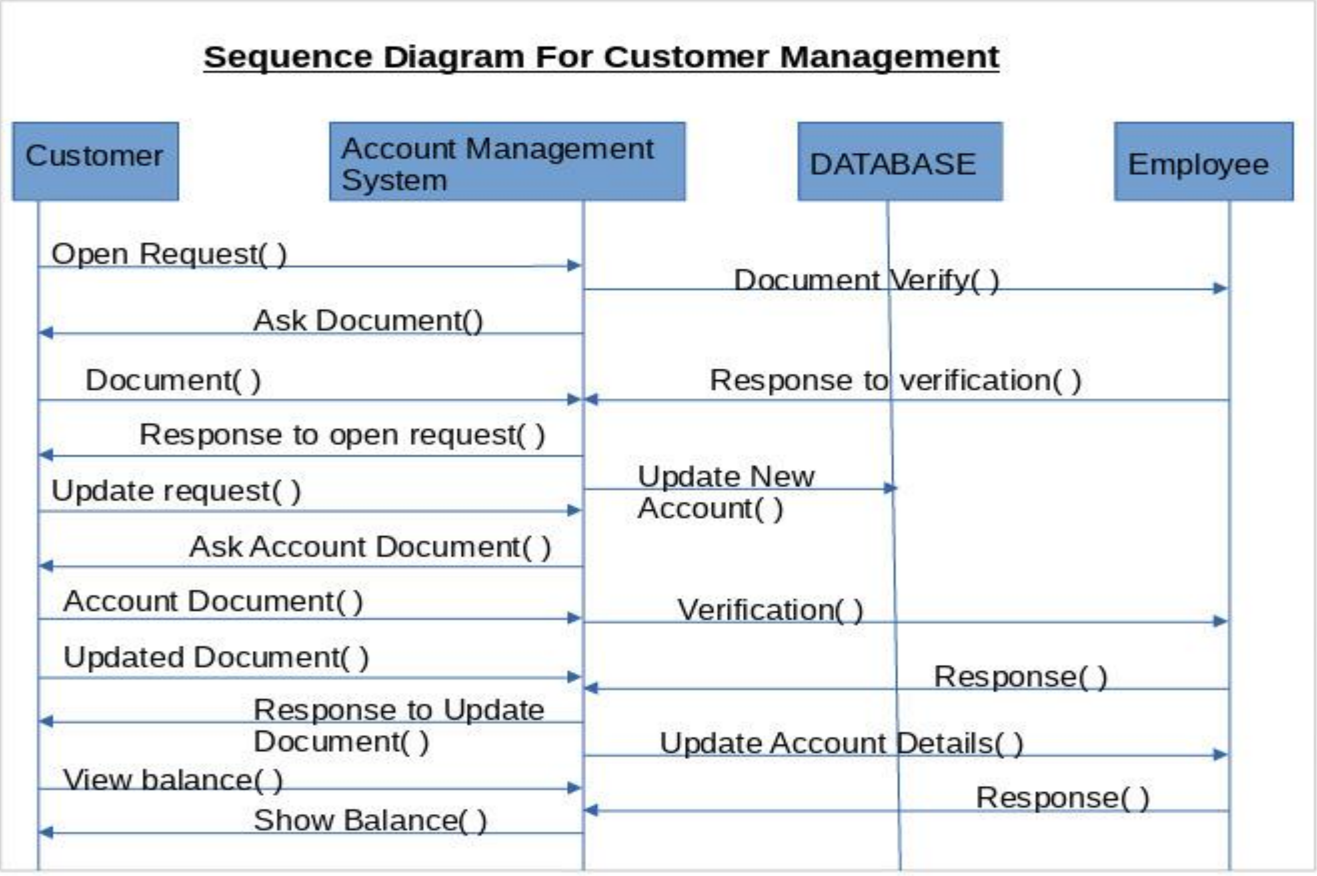
4 Design Engineering

4.1 Use Case

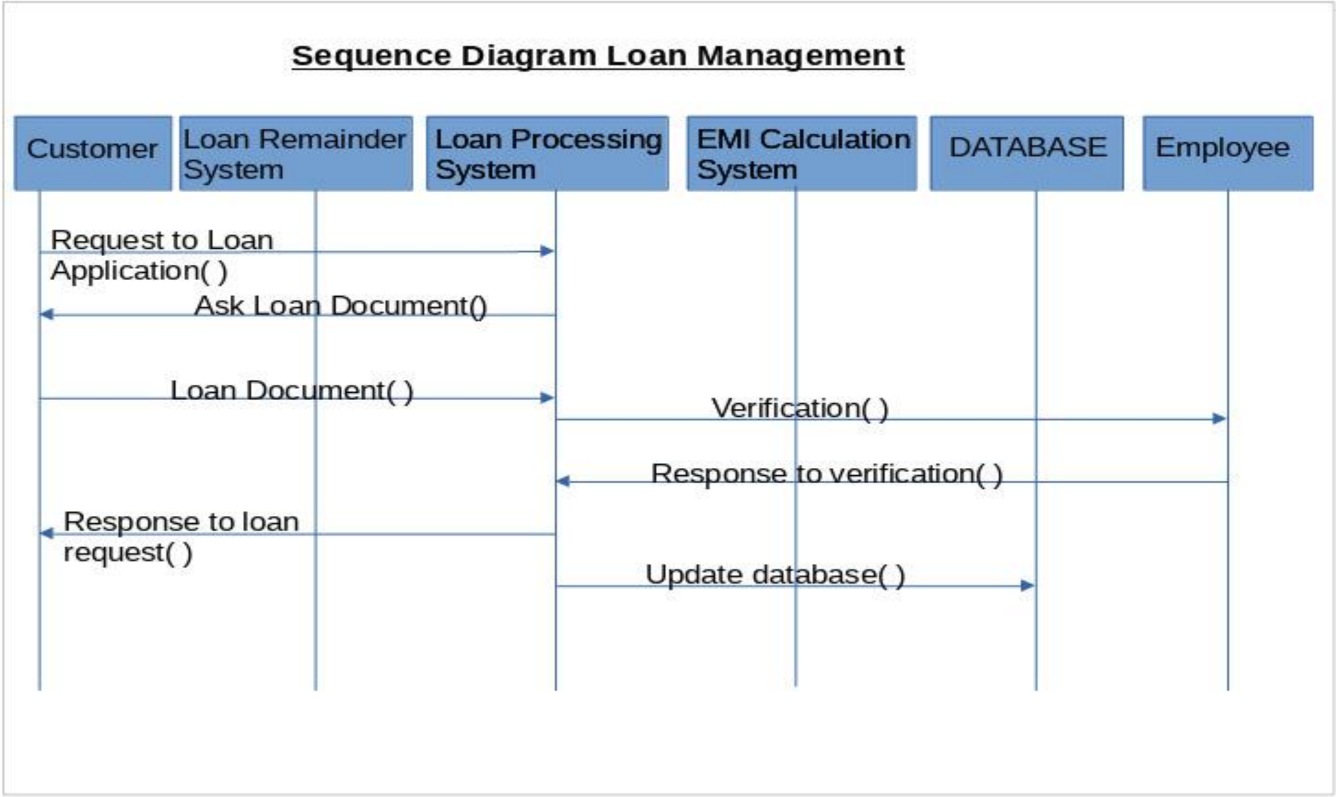


4.2 Sequential Design

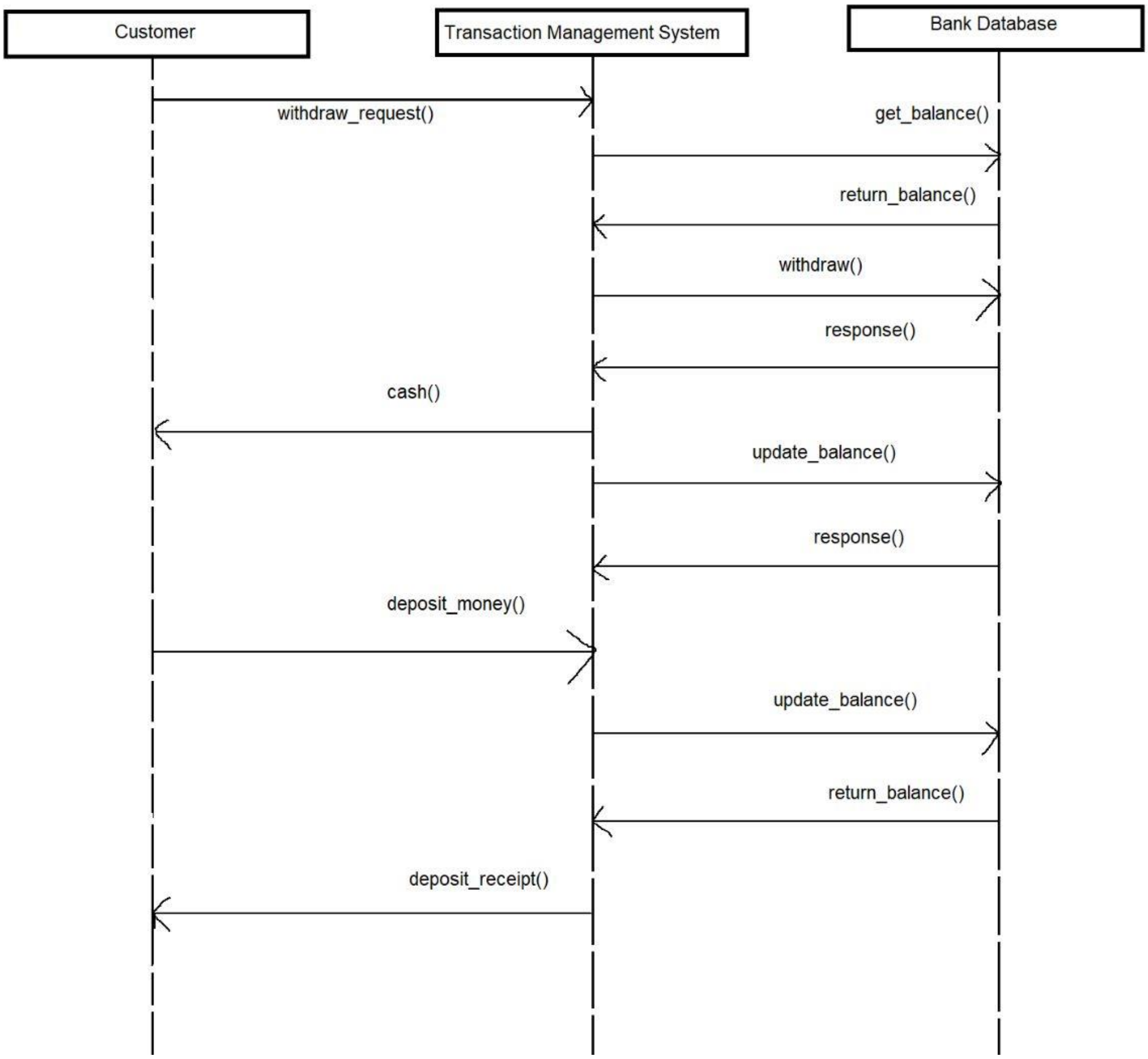
Customer Management System



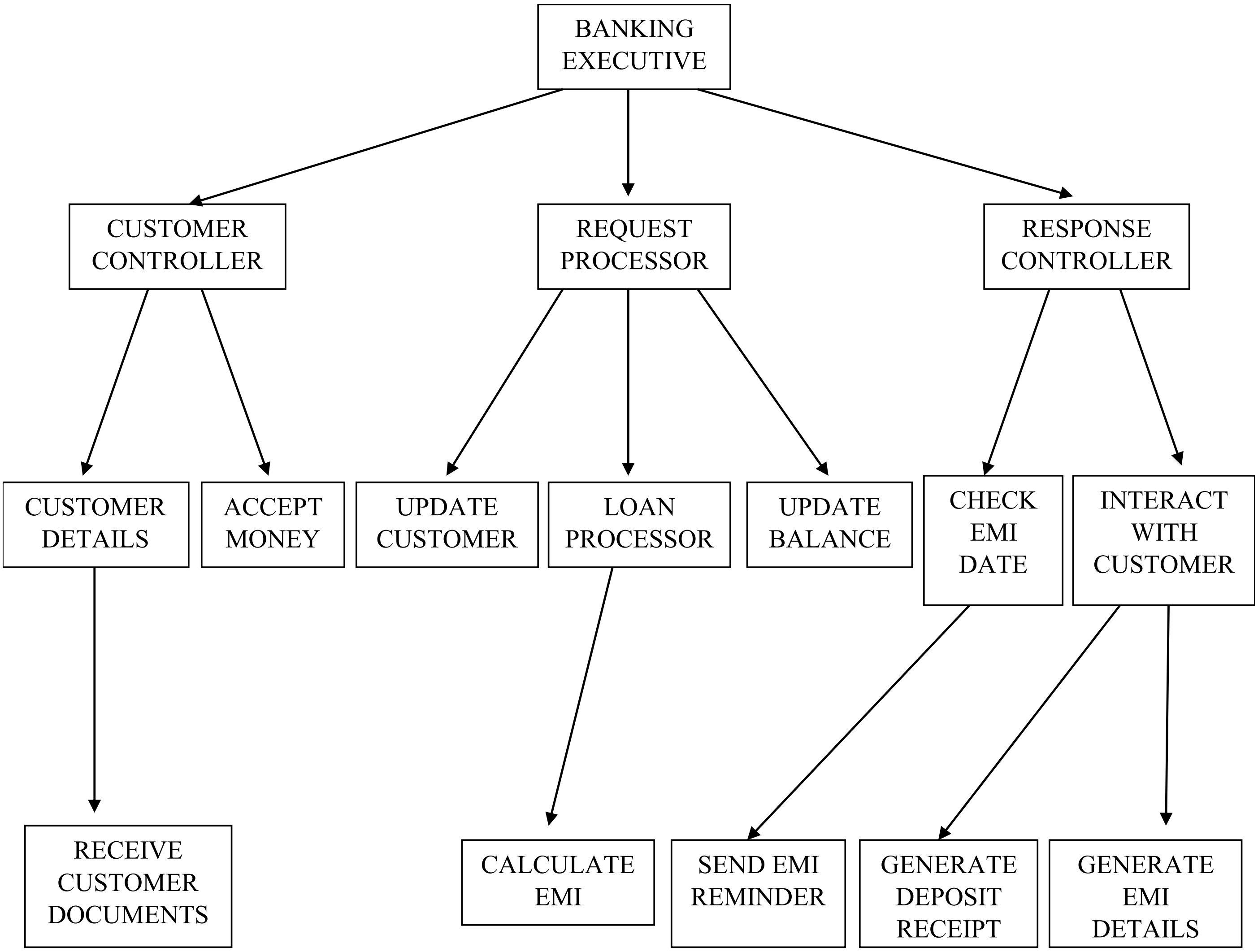
Loan Management System



Transaction Management System



4.3 Architectural Design



4.4 Database Design

Branch Table

Field	Type	NULL	Key	Default
bID	INT	NO	Primary	NULL
bNAME	VARCHAR(50)	NO		NULL
bCITY	VARCHAR(50)	NO		NULL

Customer Table

Field	Type	NULL	Key	Default
PAN_NO	VARCHAR(50)	NO	Primary	NULL
cNAME	VARCHAR(100)	NO		NULL
bDate	DATE	NO		NULL
ADDRESS	VARCHAR(500)	NO		NULL
PhNUMBER	VARCHAR(10)	NO		NULL
UID	VARCHAR(20)	NO		NULL
PWD	VARCHAR(15)	NO		NULL
MPIN	CHAR(6)	YES		NULL

Employee Table

Field	Type	NULL	Key	Default
bID	INT	NO	Foreign	NULL
eID	VARCHAR(50)	NO	Primary	NULL
eNAME	VARCHAR(100)	NO		NULL
bDate	DATE	NO		NULL
ADDRESS	VARCHAR(500)	NO		NULL
phNUMBER	VARCHAR(10)	NO		NULL

Account Table

Field	Type	NULL	Key	Default
bID	INT	NO	Foreign	NULL
ACC_NO	INT(12)	NO	Primary	NULL
Balance	DECIMAL(20,2)	YES		0.00

Loan Table

Field	Type	NULL	Key	Default
LOAN_ID	VARCHAR(50)	NO	Primary	NULL
ACC_NO	INT(12)	NO	Foreign	NULL
LOAN_TYPE	VARCHAR(50)	NO		NULL
DURATION	VARCHAR(50)	NO		NULL
AMOUNT	DECIMAL(20,2)	NO		NULL

Transaction Table

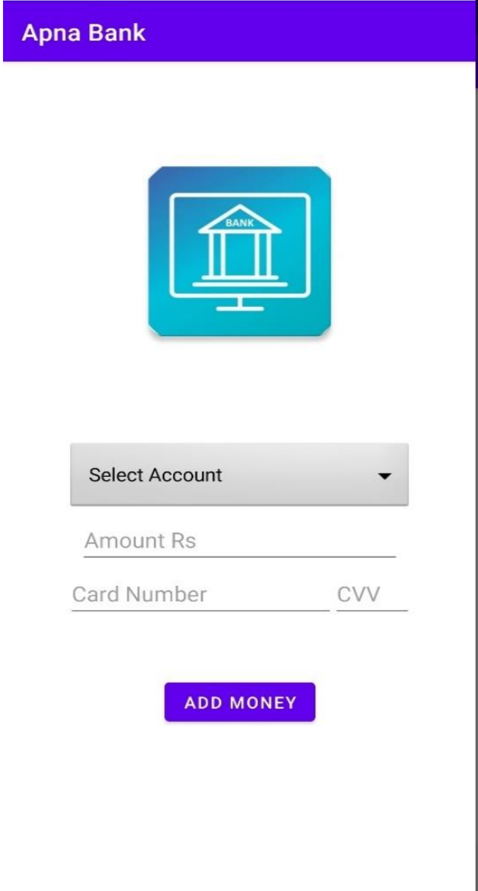
Field	Type	NULL	Key	Default
T_ID	INT	NO	Primary	NULL
TYPE	VARCHAR(50)	NO		NULL
ACC_NO	INT(12)	NO	Foreign	NULL
Amount	DECIMAL(20,2)	NO		NULL

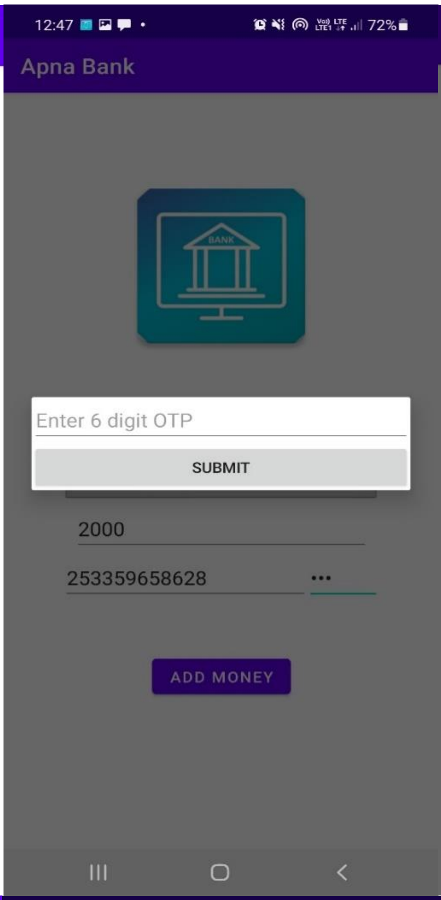
5 Implementation

Money Deposit Code

```
package com.example.apnabank;

import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.app.AppCompatDelegate;
import androidx.core.app.NotificationCompat;
import androidx.core.app.NotificationManagerCompat;

import android.app.AlertDialog;
import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.text.InputFilter;
import android.text.InputType;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.Spinner;                                                    1
import android.widget.Toast;

import java.util.Arrays;
import java.util.Random;
import java.util.concurrent.ExecutionException;

public class Deposit extends AppCompatActivity {

 String pan="",otp="";
 Button dep;
 EditText cno,cvv,amount;

 @Override
 protected void onCreate(Bundle savedInstanceState) {
  AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_deposit);
  Spinner selectAccount = findViewById(R.id.spinner);
  dep = findViewById(R.id.addmoney);
  cno = findViewById(R.id.cardno);
  cvv = findViewById(R.id.cvv);
  amount = findViewById(R.id.amount);

 Intent intent = getIntent();
 pan = intent.getStringExtra("pan");
 BackgroundTask b = new BackgroundTask(Deposit.this);

 b.execute("https://dusc.000webhostapp.com/profile.php","SELECT ACC_NO FROM account
where PAN_NO='"+pan+"'");

 String temp = null;

 try {
  temp = b.get();
                                            2
  String acc_nos[] = temp.split(";");
  String[] newArray = Arrays.copyOf(acc_nos, acc_nos.length + 1);
  newArray[0] = "Select Account";
  System.arraycopy(acc_nos, 0, newArray, 1, acc_nos.length);
  ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
android.R.layout.simple_spinner_dropdown_item, newArray);          3

  selectAccount.setAdapter(adapter);
 }
 catch (ExecutionException | InterruptedException e) {
      e.printStackTrace();                                           4
 }

dep.setOnClickListener(new View.OnClickListener() {
   @Override                                                          5
   public void onClick(View view) {
    if(selectAccount.getSelectedItem().toString().equals("Select Account"))  6
     Toast.makeText(Deposit.this, "Select valid account to deposit",   7
Toast.LENGTH_SHORT).show();

    else if(amount.getText().toString().equals(""))                   8
     Toast.makeText(Deposit.this, "Enter valid amount", Toast.LENGTH_SHORT).show(); 9

    else  if(Double.parseDouble(amount.getText().toString())<100.0)   10
     Toast.makeText(Deposit.this, "Enter valid amount more than 99",  11
Toast.LENGTH_SHORT).show();

    else  if(Double.parseDouble(amount.getText().toString())>100000.0) 12
     Toast.makeText(Deposit.this, "Enter valid amount less than 100001",13
Toast.LENGTH_SHORT).show();

    else if((cno.getText().toString()).length()!=12)          14
     Toast.makeText(Deposit.this, "Enter correct card number", 15
Toast.LENGTH_SHORT).show();

    else if((cvv.getText().toString()).length()!=3)  16
     Toast.makeText(Deposit.this, "Enter correct cvv number",   17
Toast.LENGTH_SHORT).show();

    else
    {
      depositMoney(selectAccount.getSelectedItem().toString(),    18
               Double.parseDouble(amount.getText().toString()));
    }
   }
 });                                                              19
}

private  void depositMoney(String accNO,double amt)
{
  Random rnd = new Random();
  int number = rnd.nextInt(999999);
  otp = String.format("%06d", number);
  NotificationChannel notificationChannel = new NotificationChannel("002","APNA
BANK",NotificationManager.IMPORTANCE_DEFAULT);
  notificationChannel.setDescription("OTP for transaction");
  NotificationManager notificationManager = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
  notificationManager.createNotificationChannel(notificationChannel);
  Notification.Builder builder = new Notification.Builder(Deposit.this,"002");

  builder.setSmallIcon(R.drawable.bank).setContentTitle("OTP").
      setContentText(otp).setPriority(Notification.PRIORITY_DEFAULT);

  NotificationManagerCompat notificationManagerCompat =
                      NotificationManagerCompat.from(Deposit.this);
  notificationManagerCompat.notify(002,builder.build());

  AlertDialog dialog = new AlertDialog.Builder(Deposit.this).create();
  dialog.setCanceledOnTouchOutside(false);
  EditText enterotp = new EditText(Deposit.this);
  enterotp.setInputType(InputType.TYPE_CLASS_NUMBER |                      20
InputType.TYPE_NUMBER_VARIATION_PASSWORD);
  enterotp.setEms(6);
  enterotp.setFilters(new InputFilter[] { new InputFilter.LengthFilter(6)
});
  enterotp.setHint("Enter 6 digit OTP");
  Button submit = new Button(Deposit.this);
  submit.setText("Submit");
  LinearLayout.LayoutParams lp = new
LinearLayout.LayoutParams(LinearLayout.LayoutParams.MATCH_PARENT,
LinearLayout.LayoutParams.WRAP_CONTENT);

  enterotp.setLayoutParams(lp);
  LinearLayout ll = new LinearLayout(Deposit.this);
  ll.setOrientation(LinearLayout.VERTICAL);
  ll.setLayoutParams(lp);
  ll.addView(enterotp);
  ll.addView(submit);
  dialog.setView(ll);
  dialog.show();

  submit.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View view) {
    if(enterotp.getText().toString().equals(otp))      21
    {
     BackgroundTask b = new BackgroundTask(Deposit.this);
     b.execute("https://dusc.000webhostapp.com/update.php","acc "+accNO+" amt "+amt+"
type ADD");                                                                  22
     try {
      String res = b.get();     23
      if(res.equals("true"))     24
      {
       long num = (long) Math.floor(Math.random() * 9_000_000_000L) +
        1_000_000_000L;
       String tid= String.format("%10d", num);
       BackgroundTask bt = new BackgroundTask(Deposit.this);                25

      bt.execute("https://dusc.000webhostapp.com/insert.php","INSERT INTO transaction
      values('"+tid+"','CREDIT','"+accNO+"',"+amt+")");

      String res1 = bt.get();     26
      if(res1.equals("true"))     27
      {
       Toast.makeText(Deposit.this, "Transaction Successful",
Toast.LENGTH_SHORT).show();
       dialog.dismiss();
       Intent intent = new Intent(Deposit.this,HomeScreen.class);      28
       intent.putExtra("pan",pan);
       startActivity(intent);
       finish();
      }
      else
      {
       Toast.makeText(Deposit.this, "Transaction Failed", Toast.LENGTH_SHORT).show();
       dialog.dismiss();
       Intent intent = new Intent(Deposit.this,HomeScreen.class);
       intent.putExtra("pan",pan);                                          29
       startActivity(intent);
       finish();
      }
     }
     else
     {
      Toast.makeText(Deposit.this, "Transaction Failed", Toast.LENGTH_SHORT).show();
      dialog.dismiss();
      Intent intent = new Intent(Deposit.this,HomeScreen.class);         30
      intent.putExtra("pan",pan);
      startActivity(intent);
      finish();
     }


    } catch (ExecutionException | InterruptedException e) {
           e.printStackTrace();                                  31
    }
   }
   else
   {
    Toast.makeText(Deposit.this, "Transaction Failed", Toast.LENGTH_SHORT).show();
    dialog.dismiss();
    Intent intent = new Intent(Deposit.this,HomeScreen.class);
    intent.putExtra("pan",pan);
    startActivity(intent);
    finish();                                                                 32
   }
  }
 });
}
}
```

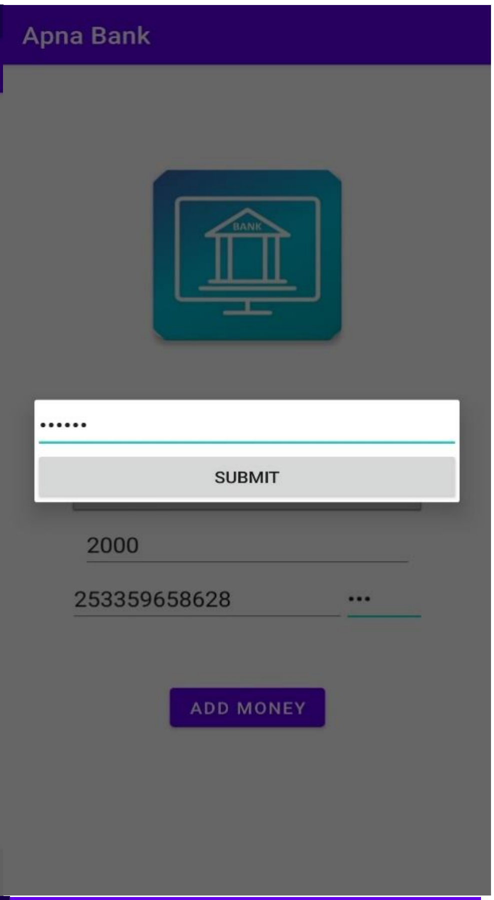
5.1 Output Screenshots





2000

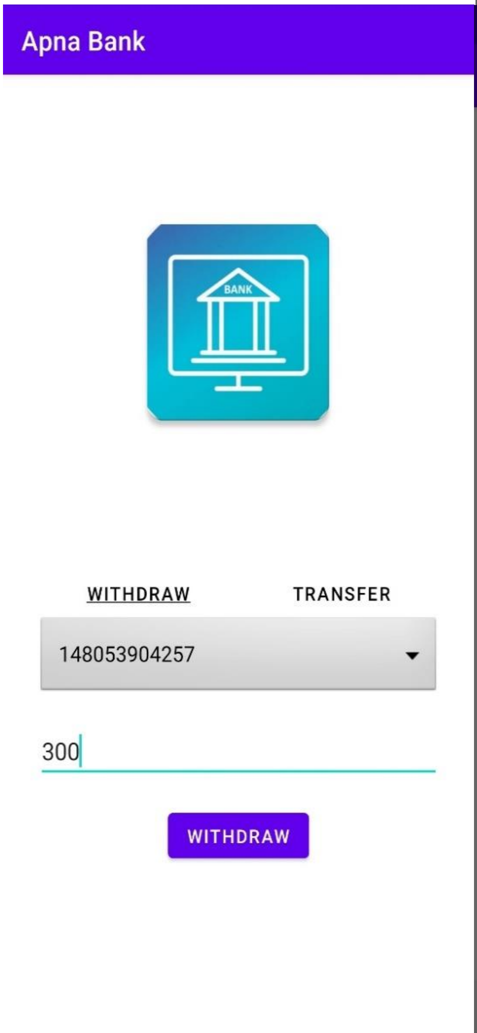
253359658628 •••



••••••

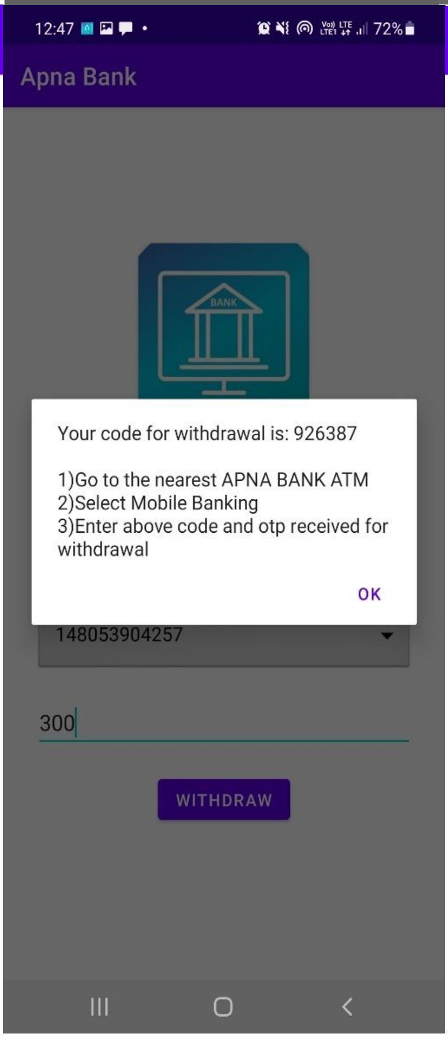
2000

253359658628 •••



148053904257

300

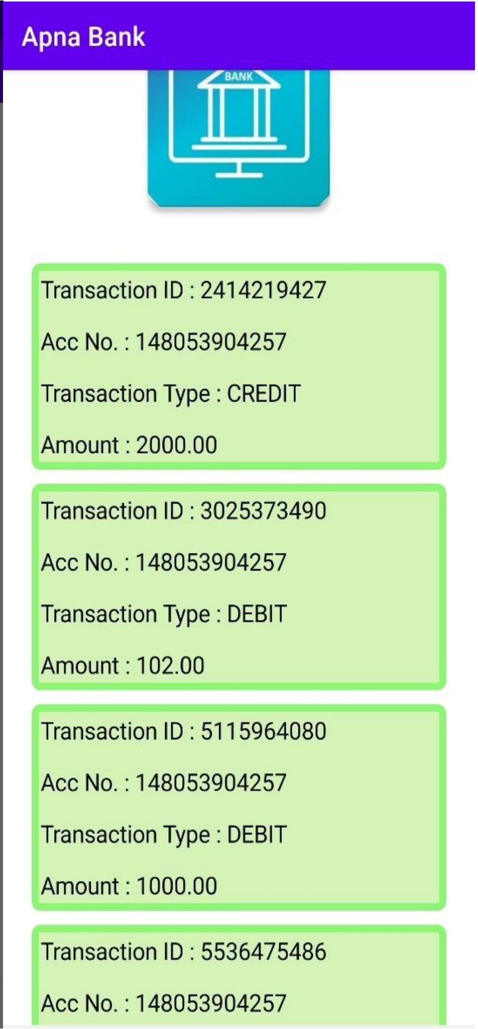


Your code for withdrawal is: 926387

1)Go to the nearest APNA BANK ATM
2)Select Mobile Banking
3)Enter above code and otp received for withdrawal

148053904257

300



Transaction ID : 2414219427
Acc No. : 148053904257
Transaction Type : CREDIT
Amount : 2000.00

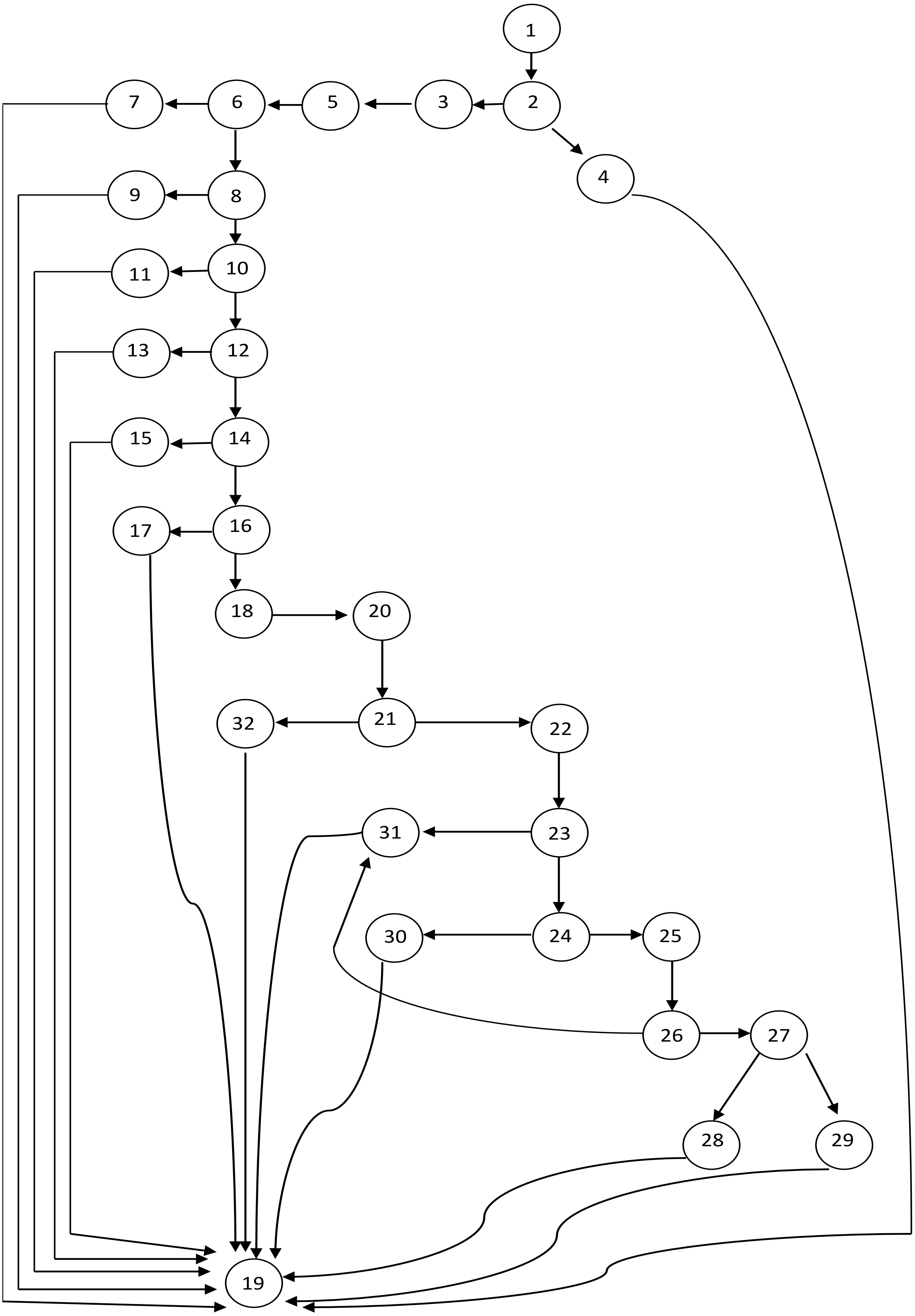
Transaction ID : 3025373490
Acc No. : 148053904257
Transaction Type : DEBIT
Amount : 102.00

Transaction ID : 5115964080
Acc No. : 148053904257
Transaction Type : DEBIT
Amount : 1000.00

Transaction ID : 5536475486
Acc No. : 148053904257

6 Testing

6.1 Control Flow Graph



6.1.1 Calculating Cyclomatic Complexity

No. of Edges = 43
No. of Nodes = 32
No. of Predicate Nodes = 12

We know, Cyclomatic Complexity = E-N+2

= 43 – 32 + 2
= 13

OR

Cyclomatic Complexity = P+1

=12+1
=13

6.1.2 Independent Paths

Path 1 – 1,2,4,19
Path 2 – 1,2,3,5,6,7,19
Path 3 – 1,2,3,5,6,8,9,19
Path 4 – 1,2,3,5,6,8,10,11,19
Path 5 – 1,2,3,5,6,8,10,12,13,19
Path 6 – 1,2,3,5,6,8,10,12,14,15,19
Path 7 – 1,2,3,5,6,8,10,12,14,16,17,19
Path 8 – 1,2,3,5,6,8,10,12,14,16,18,20,21,32,19
Path 9 - 1,2,3,5,6,8,10,12,14,16,18,20,21,22,23,31,19
Path 10 - 1,2,3,5,6,8,10,12,14,16,18,20,21,22,23,24,30,19
Path 11 - 1,2,3,5,6,8,10,12,14,16,18,20,21,22,23,24,25,26,31,19
Path 12 - 1,2,3,5,6,8,10,12,14,16,18,20,21,22,23,24,25,26,27,28,19
Path 13 - 1,2,3,5,6,8,10,12,14,16,18,20,21,22,23,24,25,26,27,29,19

6.1.3 Test Cases

Path No.	Case	Outcome	Remarks
1	Internet not connected	Login Failed	Fail
2	No account no. selected	Select Valid Account to Deposit	Fail
3	No amount entered	Enter Valid amount	Fail
4	Entered Amount less than 100	Enter valid amount, more than 99	Fail
5	Entered Amount more than 100000	Enter valid amount, less than 100000	Fail
6	Invalid card no.	Enter correct card number	Fail
7	Invalid CVV no.	Enter correct CVV number	Fail
8	Wrong OTP entered	Transaction Failed	Fail
9	Server Down	Transaction Failed	Fail
10	Server Not Responding	Transaction Failed	Fail
11	Server Not Responding	Transaction Failed	Fail
12	Every user details are correct	Transaction Successful	Pass
13	Server Not Responding	Transaction Failed	Fail

6.2 Equivalence Class Testing

Valid class- (100<=x<=1,00,000)

Invalid class- (x<=99), (x>=1,00,001), where x the amount of cash you are depositing in Bank.

INPUT	EXPEXTED OUTPUT	OBSERVED OUTPUT	MATCH
50	Enter amount more than or equal to 100	Enter amount more than or equal to 100	Yes
2,00,000	Enter amount less than or equal to 1 lakh	Enter amount less than or equal to 1 lakh	Yes
1000	Deposited successfully	Deposited successfully	Yes

6.3 Boundary Value Testing

Here boundary value are - 99,100,10000,100000,100001

INPUT	EXPEXTED OUTPUT	OBSERVED OUTPUT	MATCH
99	Enter amount more than or equal to 100	Enter amount more than or equal to 100	Yes
100	Deposited successfully	Deposited successfully	Yes
10000	Deposited successfully	Deposited successfully	Yes
1,00,000	Deposited successfully	Deposited successfully	Yes
1,00,001	Enter amount less than or equal to 1 lakh	Enter amount less than or equal to 1 lakh	Yes

7 Conclusion

“Banking Management System" keeps the day by day tally record as a complete banking. It can keep the information of Account, account opening form, Deposit, Withdrawal, and the transaction, Transaction report . The exciting part of this project is; it displays Transaction reports, Statistical Summary of Account and Interest Information.

8 References

Books –

1. R.S. Pressman, Software Engineering: A Practitioner’s Approach, McGraw-Hill
2. P. Jalote, An Integrated Approach to Software Engineering
3. R. Mall, Fundamentals of Software Engineering, Prentice-Hall of India
4. I. Sommerville, Software Engineering, Addison Wesley

Websites –

1. https://en.wikipedia.org/
2. https://www.geeksforgeeks.org/
3. https://codingtimes.in/