**WEB TO FORU BANK**

**SOFTWARE ENGINEERING PROJECT REPORT**

[Submitted in Partial Fulfilment]

As a part of the curriculum of

**B.SC (H) COMPUTER SCIENCE**

From



Mata Sundri College for Women, New Delhi-110002

University of Delhi

SUBMITTED BY:

**Sakshi Rajput (17044570015)**

**Sakshi Sharma (17044570019)**

**B.SC (H) Computer science**

**IV Semester**

Mata Sundri College for Women

Mata Sundri Lane, New Delhi-110002

**CERTIFICATE**

This is to certify that the project entitled, **“WEB TO FORU BANK”**, an online bank management system is submitted by Sakshi Rajput(17044570015) and Sakshi Sharma(17044570019) of Mata Sundri College for Women in partial fulfilment of the requirements of BSc (H) Computer Science, IV semester.

It has been completed under the supervision of Ms Ashema Hasti from Mata Sundri College for Women, Delhi University.

Ms Ashema Hasti

(Supervisor)

**ACKNOWLEDGEMENT**

We are using this opportunity to express our gratitude to everyone who supported us throughout the course of this project entitled “Web to FORU Bank” on Online Bank Management System.

We are grateful to Ms Ashema Hasti, for her constant support, guidance and for providing us the necessary resources for the project. We would also like to thank our classmates for their valuable suggestions for the betterment of this project and everybody who has helped us directly or indirectly in completion of this project.

We are immensely grateful to everyone involved in this project as without their inspiration and valuable suggestion it would not have been possible to develop the project within the prescribed time.

Sakshi Rajput - 17044570015

Sakshi Sharma - 17044570019

**ABSTRACT**

Through this project we have tried to change the traditional way of banking.

Traditionally, we had to go to bank to open our account, or to access it. This website allow users to open a bank account or to manage it online thus, reducing the wastage of time and effort.

“Web to FORU Bank” follows incremental process model because initial software requirements are reasonably well defined but the overall scope of development effort precludes a purely linear process. In addition, there may be a compelling need to provide a limited set of software functionality to user quickly and then refine and expand on that functionality in later software release. This website allows a user to open an account in the bank or to perform several activities required.

* Allow customers to open an account online
* Allow users to perform activities like pay the bills such as electricity bill, water bill or to recharge phone or DTH.
* Allow users to transfer money to another account in the same bank or in different bank.
* Allow users to ask for services.
* Allow users to view their account details.

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIG NO.** | **NAME** | **PAGE NO.** |
| 1.1 | INCREMENTAL MODEL | 2 |
| 3.1 | USE CASE DIAGRAM | 15 |
| 5.1 | CONTEXT LEVEL DIAGRAM | 31 |
| 5.2 | LEVEL-1 DFD | 32 |
| 5.3 | LOGIN | 33 |
| 5.4 | REGISTER | 33 |
| 5.5 | ADD/REMOVE EMPLOYEE | 34 |
| 5.6 | ADD/REMOVE CUSTOMER | 34 |
| 5.7 | VIEW USER DETAILS | 35 |
| 5.8 | VEW ACCOUNT DETAILS | 35 |
| 5.9 | MONEY TRANSFER | 36 |
| 5.10 | REQUESTS | 36 |
| 5.11 | BILL PAYMENTS | 37 |
| 5.12 | UPDATE CUSTOMER DETAILS | 37 |
| 5.13 | ER DIAGRAM | 40 |
| 9.1 | WELCOME PAGE | 56 |
| 9.2 | LOGIN | 57 |
| 9.3 | REGISTRATION | 58 |
| 9.4 | VIEW USER DETAILS | 59 |
| 9.5 | VIEW ACCOUNT DETAILS | 60 |
| 9.6 | MONEY TRANSFER | 61 |
| 9.7 | BILL PAYMENT | 62 |
| 9.8 | ADD CUSTOMER | 63 |
| 9.9 | ADD EMPLOYEE | 64 |
| 9.10 | SERVICES | 65 |
| 9.11 | REMOVE EMPLOYEE | 66 |
| 9.12 | REMOVE CUSTOMER | 67 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **NAME** | **PAGE NO.** |
| 2.1 | PROCESS SCHEDULING | 4 |
| 2.2 | TIMELINE CHART | 5 |
| 4.1 | RISK TABLE | 27 |
| 5. 1 | CUSTOMER | 41 |
| 5.2 | EMPLOYEE | 42 |
| 5.3 | ADMIN | 43 |
| 5.4 | DEPENDENT | 44 |
| 5.5 | ACCOUNT | 44 |
| 5.6 | BANK BRANCH | 44 |
| 6.1 | FUNCTION POINT WORKSHEET | 46 |
| 6.2 | VALUE ADJUSTMENT FACTORS(VAFs) | 47 |
| 6.3 | COMPLEXITY AND WEIGHING TABLE FOR OBJECT POINTS | 49 |
| 6.4 | PRODUCTIVITY RATE FOR OBJECT POINTS | 49 |
| 7.1 | TEST CASES | 52 |

**LIST OF ABBREVIATIONS**

1. SRS - Software Requirements Specification
2. DFD - Data Flow Diagram
3. DD - Data Dictionary
4. COCOMO - COnstructive COst Model
5. SMS - Short Message Service
6. A/C - Account

|  |  |
| --- | --- |
| **CHAPTER** | **PAGE NO.** |
| 1. **INTRODUCTION**    1. PROBLEM STATEMENT    2. PROCESS MODEL | 1  2 |
| 1. **PROJECT SCHEDULING**    1. PROCESS SCHEDULING    2. TIMELINE CHART | 4  5 |
| 1. **SOFTWARE REQUIREMENTS SPECIFICATION**    1. INTRODUCTION       1. PURPOSE AND SCOPE       2. OVERVIEW    2. PROJECT DESCRIPTION       1. PRODUCT PRESPECTIVE       2. PRODUCT FUNCTIONS       3. USER CHARACTERISTICS       4. GENERAL CONSTRAINTS       5. ASSUMPTIONS AND DEPENDENCIES    3. SPECIFIC REQUIREMENTS       1. EXTERNAL INTERFACES       2. FUNCTIONAL REQUIREMENTS       3. PERFORMANCE REQUIREMENTS       4. DESIGN CONSTRAINTS       5. USE CASE DIAGRAM AND DESCRIPTIONS | 6  7  11 |
| 1. **RISK MANAGEMENT**    1. INTRODUCTION       1. CHARACTERISTICS OF RISK       2. CATEGORIES OF RISK       3. COMPONENTS OF RISK    2. RISK IDENTIFICATION | 24  26 |
| 1. **DESIGN**    1. DATA FLOW DIAGRAM       1. CONTEXT LEVEL DIAGRAM       2. LEVEL-1 DFD       3. LEVEL-2 DFD    2. DATA DICTIONARY    3. LOGICAL DATABASE DESIGN       1. ER DIAGRAM       2. DATA DESIGN       3. COMPONENT LEVEL DESIGN | 31  38  40 |
| 1. **ESTIMATION**    1. SIZE ESTIMATION (FUNCTION BASED METRICS)    2. COST ESTIMATION (COCOMO-II MODEL) | 46  48 |
| 1. **TESTING**    1. PSEUDO CODE    2. WHITE BOX TESTING       1. FLOW GRAPH       2. INDEPENDENT PATHS       3. CYCLOMATIC COMPLEXITY       4. TEST CASES | 50  51 |
| 1. **CONCLUSION AND REFERENCES**    1. CONCLUSION    2. LIMITATIONS OF THE SYSTEM    3. REFERENCES | 53  54  55 |
| 1. **ANNEXURES** | 56 |

**TABLE OF CONTENTS**

**1.1 Problem Statement**

**1.1.1 PURPOSE**

“Web to FORU bank" is a software designed to keep account details of the customers over the internet and to provide banking facilities online to customers.

**1.1.2 SCOPE**

Online Banking System is specifically developed for internet banking for balance enquiry, funds transfer to another account, bill payments and request services like request for cheque book/change details, request for SMS, mini statement (viewing monthly and annual statement).

The traditional way of maintaining details of a user in a bank was to enter the details and record them. Every time the user needs to perform some transaction he has to go to bank and perform the necessary action, which may not be so feasible all the time. It may be a hard hitting task for the users and the bankers too. Here, we provide automation for banking system through internet. Internet banking system project captures-activities performed by different roles in real life banking which provides enhanced techniques for maintaining the required information up-to-date, which results in efficiency. The project gives real life understanding of internet banking and activities performed by various roles in the supply chain.

**1.1.3 OVERVIEW**

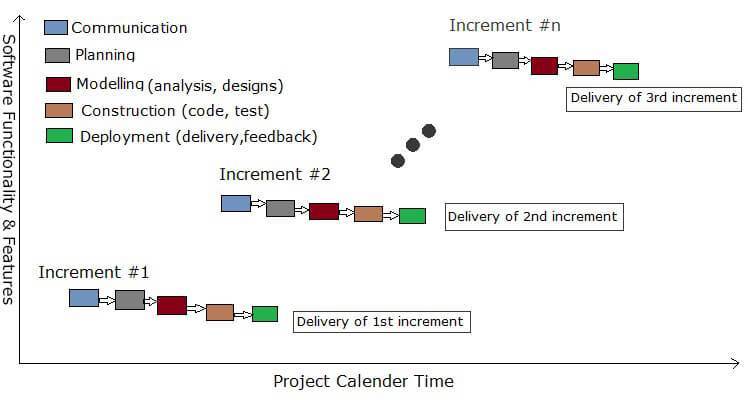
The project entitled “Web to FORU Bank " is an online bank management system of ‘FORU’ bank for maintaining customer's account details online. It enables the customer to access the bank website for viewing their bank balance, bill payments and money transfer. The customer can request past three months passbook details, SMS and email services to get updated with their account balance.

The main aim of this project is to facilitate customers with the additional functionalities like mobile recharge, DTH recharge, electricity bill, etc. to provide customers/users with the best experience of the software.

**1.2 INCREMENTAL PROCESS MODEL**

The **incremental model** combines elements of linear and parallel process flows, the incremental model applies linear sequences in a staggered fashion as calendar time progresses. Each linear sequence produces deliverable “increments” of the software in a manner that is similar to the increments produced by an evolutionary process flow. It should be noted that the process flow for any increment can incorporate the prototyping paradigm.

“Web to FORU Bank” follows incremental process model because initial software requirements are reasonably well defined but the overall scope of development effort precludes a purely linear process. In addition, there may be a compelling need to provide a limited set of software functionality to user quickly and then refine and expand on that functionality in later software release.



**Figure No. 1.1: Incremental Model**

The incremental process model focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and also provide a platform for evaluation by the user. Incremental development is particularly useful when staffing is unavailable for a complete implementation by the business deadline that has been established for the project. Early increments can be implemented with fewer people. If the core product is well received, then additional staff (if required) can be added to implement the next increment. In addition, increments can be planned to manage technical risks.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TASKS** | **PLANNED START** | **ACTUAL START** | **PLANNED COMPLETE** | **ACTUAL COMPLETE** | **ASSIGNED PERSON** | **EFFORT ALLOCATED** |
| PROBLEM STATEMENT | JAN W1,D1 | JAN W1,D1 | JAN W1,D7 | JAN W1,D7 | SAKSHI SHARMA | 1 PM |
| SOFTWARE PROCESS MODEL | JAN W1,D1 | JAN W1,D1 | JAN W2,D7 | JAN W2,D7 | SAKSHI RAJPUT | 1 PM |
| PROJECT SCHEDULING AND TIMELINE CHART | JAN W2,D1 | JAN W2,D1 | JAN W2,D7 | JAN W2,D7 | SAKSHI SHARMA | 1PM |
| CONTEXT LEVEL DIAGRAM | JAN W3,D1 | JAN W3,D1 | JAN W3,D7 | JAN W3,D7 | SAKSHI SHARMA | 1PW |
| DFD LEVEL-1 | JAN W3,D1 | JAN W3,D1 | JAN W3,D7 | JAN W3,D7 | SAKSHI RAJPUT, SAKSHI SHARMA | 2 PW |
| DFD LEVEL-2 | JAN W4,D1 | JAN W4,D2 | FEB W2,D7 | FEB W2,D7 | SAKSHI SHARMA | 1 PW |
| DATA DICTIONARY | JAN W4,D1 | JAN W4,D1 | FEB W1,D1 | FEB W1,D1 | SAKSHI SHARMA | 1 PW |
| SRS | JAN W4,D1 | JAN W4,D1 | FEB W3,D7 | FEB W3,D7 | SAKSHI RAJPUT | 1 PW |
| SOFTWARE METRICS | FEB W3,D1 | FEB W3,D1 | FEB W4,D7 | FEB W4,D7 | SAKSHI SHARMA | 1 PW |
| COCOMO MODEL | MAR W1,D1 | MAR W1,D2 | MAR W2,D7 | MAR W2,D7 | SAKSHI SHARMA | 1 PW |
| RISK ANALYSIS | MAR W1,D1 | MAR W1,D1 | APR W1,D7 | APR W1,D7 | SAKSHI RAJPUT | 1 PM |
| ER DIAGRAM | MAR W3,D1 | MAR W3,D3 | MAR W4,D7 | MAR W4,D7 | SAKSHI RAJPUT | 1 PW |
| DATA DESIGN | MAR W4,D1 | MAR W4,D1 | MAR W4,D7 | MAR W4,D7 | SAKSHI RAJPUT | 1 PW |
| COMPONENT LEVEL DESIGN | MAR W2,D6 | MAR W2,D7 | MAR W4 ,D7 | MAR W4,D7 | SAKSHI SHARMA | 1 PW |
| USE CASE DIAGRAM | APR W1,D1 | APR W1,D1 | APR W1,D7 | APR W1,D7 | SAKSHI RAJPUT | 1 PW |
| USE CASE DESCRIPTION | APR W2,D1 | APR W2,D1 | APR W2,D7 | APR W2,D7 | SAKSHI RAJPUT | 1 PW |
| TEST CASES | APR W3,D1 | APR W3,D1 | APR W4,D7 | APR W4,D2 | SAKSHI RAJPUT | 1 PW |

**TABLE 2.1 PROCESS SCHEDULING**

**TABLE 2.2 TIMELINE CHART**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MONTH** | **JANUARY** | **FEBRUARY** | **MARCH** | **APRIL** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **WEEK** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** |
| PROBLEM STATEMENT |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| SOFTWARE PROCESS MODEL |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| PROJECT SCHEDULING AND TIMELINE CHART |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| CONTEXT LEVEL DIAGRAM |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFD LEVEL-1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DFD LEVEL-2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DATA DICTIONARY |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| SRS |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| SOFTWARE METRICS |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| COCOMO MODEL |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| RISK ANALYSIS |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ER DIAGRAM |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| DATA DESIGN |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| COMPONENT LEVEL DESIGN |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| USE CASE DIAGRAM |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| USE CASE DESCRIPTION |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| TEST CASES |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**3. SOFTWARE REQUIREMENS SPECIFICATION**

**3.1. INTRODUCTION**

This document gives detailed functional and non-functional requirements for the Bank Management System. This software will support online banking transactions. The purpose of this document is that the requirements mentioned in it should be utilized by software developer to implement the system.

**3.1.1 PURPOSE & SCOPE**

Online Banking System is specifically developed for internet banking for Balance Enquiry, Funds transfer to another account, bill payments and request services like request for cheque book / change details, request for SMS, mini statement (viewing monthly and annual statement).

The traditional way of maintaining details of a user in a bank was to enter the details and record them. Every time the user needs to perform some transaction he has to go to bank and perform the necessary action, which may not be so feasible all the time. It may be a hard hitting task for the users and the bankers too .The project gives real life understanding of internet banking and activities performed by various roles in the supply chain. Here, we provide automation for banking system through internet. Internet banking system project captures-

Activities performed by different roles in real life banking which provides enhanced techniques for maintaining the required information up-to-date, which results in efficiency. The project gives real life understanding of internet banking and activities performed by various roles in the supply chain.

**3.1.2 OVERVIEW**

The project entitled “Web to FORU Bank " is an online bank management system of ‘FORU’ bank for maintaining customer's account details online. It enables the customer to access the bank website for viewing their bank balance, bill payments and money transfer. The customer can request past three months passbook details, SMS and email services to get updated with their account balance.

The main aim of this project is to facilitate customers with the additional functionalities like mobile recharge, DTH recharge, electricity bill, etc. to provide customers/users with the best experience of the software.

**3.2. PROJECT DESCRIPTION**

**3.2.1 PRODUCT PRESPECTIVE**

The client will have client interface in which he can interact with the banking system. It is a web based interface which will be the web page of the banking application. On starting the application, a page is displayed asking the type of user, whether he is admin, bank employee or customer, in this login page user can enter login details. If the login particulars are valid then the user is taken to a home page where he has the entire transaction list that he can perform with the bank. All the above activities come under the client interface.

The administrator will have an administrative interface which is a GUI so that he can view the entire system. He will have the same login page where he can enter the login particulars so that he can perform all his actions. This administrative interface provides different environment such that he can maintain employees’ database. He can add new employees and can remove the employee details, if the employee has left.

The employee will have an interface which is a GUI so that he can view the entire system. He will have the same login page where he can enter the login particulars so that he can perform all his actions. This interface provides different environment such that he can maintain customer database. He can add new customers and can remove the customer details if the customer has left. Employee is responsible for keeping customer’s data up-to-date.

**3.2.1.1** **SYSTEM INTERFACES**

This system does not have any interfaces with any of the existing systems.

**3.2.1.2 SYSTEM SPECIFICATIONS**

**3.2.1.2.1** **HARDWARE REQUIREMENTS**

No additional specific hardware Interfaces are needed during the operation of the “Web to FORU bank”.

**3.2.1.2.2 SOFTWARE REQUIREMENTS**

* **Operating system:** We have chosen windows operating system (all versions of windows 8 and 10) for its best support and user friendliness.
* **Database Management Software:** to save the records of the customers and bank employees and their details, MY SQL database is used.

**3.2.2 PRODUCT FUNCTIONS**

**Admin's Functionality**

* **Login/Logout** – Admin has to login to access the software using Id and password.
* **Add /Remove employees** – Admin has to specify the users who can manage the software.
* **View employee details** – Admin can view its employee details.

**Bank Employees functionality**

* **Login/Logout –** Employee has to login to gain access to the software using his/her Id and password.
* **Add/Remove end users** –They can add the new customers and remove the customer who have closed their account.
* **View and Update end users account details** –They has the access to view their customer details and update it.
* **Generate end users details** –They can print customer details.
* **Service customer requests** – Employee has to provide services demanded by the customers.

**End Users Functionality**

* **Register**–User who is new to the software should register themselves.
* **Login/Logout** –Registered users can simply login to the software using their Customer ID and password.
* **View Account balance**- They can view their current account balance.
* **Print transaction history** – They can request and print past three months transaction history of their account.
* **Request email/SMS services**– They can request email and SMS service for their transactions.
* **Fund transfer**– Services like fund transfer is also available online.
* **Bill payments**- They can pay mobile phone bill, electricity bill, water bill, gas bill etc. directly through their bank account.
* **Request check book** – They can request for a new check book online.
* **Contact us –** If they have any problem related to the software or their account they can contact to the bank.

**3.2.3 USER CHARACTERISTICS**

**USER1**:Admin: - Admin can access the details of all the bank employees of the branch. Admin can add new employees and can remove existing employees. Admin is also responsible for keeping the personal details of working employees up-to-date. Admin need to have a brief knowledge of the functionalities and internal processing of the system.

**USER2**: Bank employee:-Bank employee can access the details of all the customers. Employee can add new customers and remove existing customers. Employee is also responsible for keeping the personal details of customers up-to-date. Employee need to have a complete knowledge of the functionalities and internal processing of the system.

**USER 3**: General user:-General user will login to the system to know more about their bank account. The user does not need to have complete understanding of the complex functionalities and internal processing of the system.

**3.2.4 GENERAL CONSTRAINTS**

* The information of all the users must be stored in a database that is accessible by the Online Banking System.
* The Online Banking System is connected to the computer and is running all 24 hours a day.
* The users access the Online Banking System from any computer that has internet browsing capabilities and an internet connection.
* The user must have their correct user ID and password to enter into the Online Banking System.

**3.2.5 ASSUMPTIONS AND DEPENDENCIES**.

It is assumed that the general user, bank employee and admin are familiar with operating computer. It is assumed that the user is familiar with using a web application. No other assumptions or dependencies are identified.

**3.3. SPECIFIC REQUIREMENTS**.

**3.3.1 EXTERNAL INTERFACES**

**3.3.1.1 USER INTERFACES**

**GUI**-This interface must be highly interactive because there will not be an assistance for the user who is operating the system. The screens appearing should be designed in such a manner that it can draw user attention towards the new plan for the customer. Also the pin and password confidentiality should be maintained. This can be done using asterisks at the password panel. Proper security messages should be displayed at most of the places.

**3.3.1.2 HARDWARE INTERFACE**

Various interfaces for the product could be

1. Touch screen/Monitor
2. Keypad
3. Continuous battery backup

**3.3.1.3 SOFTWARE INTERFACE**

Any windows operating system.

**3.3.2 FUNCTIONAL REQUIREMENTS**

**Admin's Functionality**

* **Login/Logout** – Admin has to login to access the software using Id and password.
* **Add /Remove employees** – Admin has to specify the users who can manage the software.
* **View employee details** – Admin can view its employee details.

**Bank Employees functionality**

* **Login/Logout –** Employee has to login to gain access to the software using his/her Id and password.
* **Add/Remove end users** –They can add the new customers and remove the customer who have closed their account.
* **View and Update end users account details** –They has the access to view their customer details and update it.
* **Generate end users details** –They can print customer details.
* **Service customer requests** – Employee has to provide services demanded by the customers.

**End Users Functionality**

* **Register**–User who is new to the software should register themselves.
* **Login/Logout** –Registered users can simply login to the software using their Customer ID and password.
* **View Account balance**- They can view their current account balance.
* **Print transaction history** – They can request and print past three months transaction history of their account.
* **Request email/SMS services**– They can request email and SMS service for their transactions.
* **Fund transfer**– Services like fund transfer is also available online.
* **Bill payments**- They can pay mobile phone bill, electricity bill, water bill, gas bill etc. directly through their bank account.
* **Request check book** – They can request for a new check book online.
* **Contact us –** If they have any problem related to the software or their account they can contact to the bank.

**3.3.3 PERFORMANCE REQUREMENTS**

**3.3.3.1 SECURITY:-**

The Banking System must be fully accessible to only authentic user. It should require pin for entry to a new environment.

**3.3.3.2 RELIABILITY:-**

The application should be highly reliable and it should generate all the updated information in correct order.

**3.3.3.3 AVAILABILITY:-**

Any information about the account should be quickly available from any computer to the authorized user. The previously visited customer’s data must be cleared.

**3.3.3.4 MAINTAINABILITY:-**

The application should be maintainable in such a manner that if any new requirements occurs then it should be easily incorporated in an individual module.

**3.3.3.5 PORTABILITY:-**

The application should be portable on any Window based system. It should not be machine specific.

**3.3.4 DESIGN CONSTRAINTS:**

* **Software Language Used-**
  + The languages that can be used for coding Online Banking System are c, c++, java and HTML.
* **Database Design-**
  + In our database design, we give names to data flows, processes and data stores. Although the names are descriptive of data, they do not give details. Our interest is to build some details of the contents of data flow, processes and data stores. A data dictionary is a structured repository of data about data. It is a set of rigorous definitions of all DFD data elements and structures.

**3.3.4.1 STANDARD COMPLIANCE**

**Report format**: All the reports produced for this project are in compliance with the standard templates in accordance with the bank guidelines and policy.

**Naming Conventions:** All the documents will be named using the standard naming conventions.

* + 1. **USE CASE DIAGRAM AND DESCRIPTION**

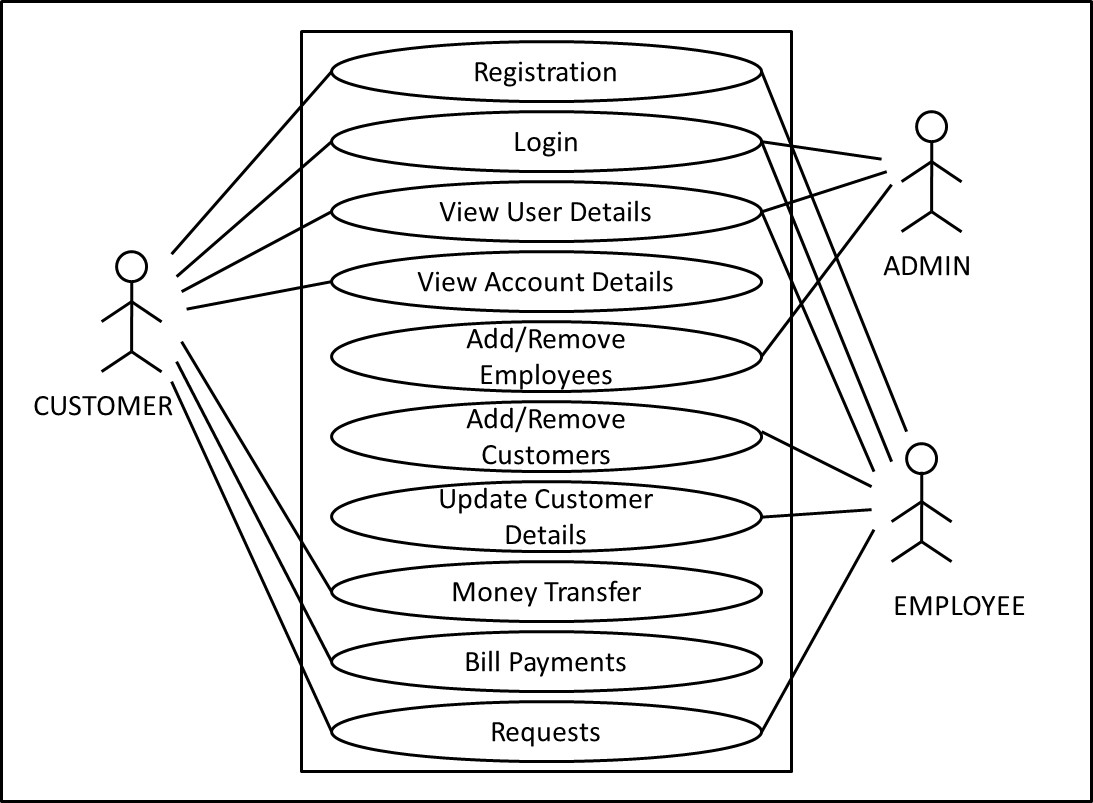


Fig. 3.1 USE CASE DIAGRAM

**USE CASE DESCRIPTION**

* **Register**

**Purpose:** This Use case registers a new customer.

**Actors:** Customer, Employee.

**Preconditions:** Customer must not be having an account in any bank branch.

**Basic flow of events:**

1. This use case starts when user wants to open an account in the bank.

2. The user is asked to enter his details like Name, Gender, Age, Address, Phone number, UIDAI, Email Id, Branch, Account Type and upload his photograph.

3. The user enter his details.

4. The user scan and upload Id proof and address proof.

5. Employee verify that details entered by the customer are correct, the customer is registered.

6. Customer Id is provided to the customer and he can set his Password.

**Alternate flows:**

1. Either the user is a customer of the bank or the entered details are incorrect.

2. The customer is notified that registration error has occurred through a pop up message.

**Post condition:**

1. The user is registered successfully, the customer database is changed.

2. If not, the customer database remains unchanged.

* **Login**

**Purpose:** This use case describes how a User login to bank website.

**Actors:** Admin, Customer or Employee.

**Preconditions:** The actors must have their User IDs and Passwords.

**Basic flow of events:**

1. This use case starts when user wants to access the bank website.

2. The user is asked to enter User Id, Password and the Type of user.

3. The user enters his User ID, Password and Type of user.

4. The user details are verified from respective records.

5. If the details entered are correct, the customer is login successfully.

6. The home page is displayed.

**Alternate flow:**

1. The details entered by the user are incorrect.
2. The user is notified that login is unsuccessful through a popup message.

**Post condition:**

The system is login successfully. If not, the system state remains unchanged.

* **View User Details**

**Purpose:** This use case allow user to check his personal details.

**Actors:** Admin, Employee or customer.

**Preconditions:**

The Actor must have his User id or Password.

**Basic flow of events:**

1. The user is asked to enter the Password.

2. The user enter his password.

3. The bank database verify user i.e. ensure that user exist.

4. The details of the user are displayed from the database.

5. The user can print his details, if required.

**Alternate flow:** There is no alternate flow.

**Post condition:** The user can verify his details. The system state remains unchanged.

* **View Account Details**

**Purpose:** This use case allow customer to check his personal details and account details.

**Actors:** Customer.

**Preconditions:** Customer must have User ID and Password.

**Basic flow of events:**

1. The customer enter his password.

2. The customer record verify the customer password.

3. The details of the customer are retrieved and displayed.

4. The customer can view all of his personal details like Name, Gender, Age, Address, Phone number, UIDAI, Email Id, Branch, Account Type as well as account details like Account number, balance.

5. The customer can print his details, if required.

**Alternate flow:**

Customer enters wrong password, he is notified through a pop up message.

**Post condition:** The system state remains unchanged.

* **Add/Remove Employee**

**Purpose:** This use case allows Admin to add a new employee or remove the existing employee.

**Actors:** Admin.

**Preconditions:** Admin must have Admin Id and Admin Password.

**Basic flow of events:**

1. A menu page is displayed.

2. The Admin selects whether to add a new employee or to remove the existing employee.

3. To add employee, the Admin enter employee details.

4. The Admin scan and upload Identity proof.

5. The Employee database is updated.

6. Employee Id and password are assigned.

7. To remove an employee details, the Admin enters Employee Id.

8. The Admin confirms removal.

9. The Employee database is updated.

**Alternate flow:** There is no alternate flow.

**Post condition:** The Employee database is updated.

* **Add/Remove Customer**

**Purpose:** This use case allows employee to add a new customer or remove the existing customer.

**Actors:** Employee.

**Preconditions:**

1. Employee must have Employee Id and Employee Password.

2. Customer is not registered online.

**Basic flow of events:**

1. The menu is displayed.

2. The employee chooses whether to add a new customer or to remove existing customer.

3. To add a customer who is registering at the bank, the employee get the details of the customer.

4. The employee verify the details.

5. If correct, Employee add the details to bank database.

6. Employee scan and upload the identity proof of customer.

7. The customer database is updated.

8. The customer is assigned the Customer ID and customer can set his Password.

9. Employee get the Customer ID of customer who has requested to close their account.

10. To remove customer, employee enters Customer ID.

11. The details of the customer are displayed.

12. The employee confirms deletion.

13. The customer database is updated.

**Alternate flow:** There is no alternate flow.

**Post condition:** The customer database is updated.

* **Update Customer Details**

**Purpose:** This use case allows Employee to update the details of customer.

**Actors:** Employee.

**Preconditions:**

Employee must have Employee ID and Password.

**Basic flow of events:**

1. The employee get the details of customers who has requested to update their bank account or who has performed a transaction offline.

2. The employee check for the details.

3. Employee update the details of customer if needed.

**Alternate flow:** There is no alternate flow.

**Post condition:** The database is updated.

* **Money Transfer**

**Purpose:** This use case allow customer to transfer money to another account in same or different bank.

**Actors:** Customer.

**Preconditions:** Customer must have customer ID and Password.

**Basic flow of events:**

1. The customer is asked to enter the receiver details: Bank name, Account holder’s name, Account number, Account type, IFSC code, Amount.

2. The customer enters password and OTP.

3. The password and OTP are verified from the customer database.

4. The customer is asked to proceed or cancel the request.

5. If user clicks proceed, the amount is checked and balance is updated accordingly.

6. Transaction is confirmed.

**Alternate flow:**

1. If the user enters wrong password or OTP, the message is delivered to customer.

2. If there is not enough balance, the user is notified through popup message.

**Post condition:** Money is transferred to another account .The account balance is updated.

* **Bill Payments**

**Purpose:** This use case allow customer to pay the bills.

**Actors:** Customer.

**Preconditions:** Customer must have user ID and Password.

**Basic flow of events:**

1. The customer enters his Id and password.

2. The customer selects the type of bill.

3. The customer enters the bill Id.

4. The customer fills the amount to be paid.

5. The customer balance is updated.

**Alternate flow:**

1. If there is not enough balance, the user is notified through popup message.

**Post condition:**

1. The bill amount is paid.

2. The user pays bill amount more than his balance, the system state remains unchanged.

* **Requests**

**Purpose:** This use case allows customer to request for services.

**Actors:** Employee and customer.

**Preconditions:** The Customer and Employees must have Id and password.

**Basic flow of events:**

1. The customer enters the required service details.

2. The request is sent to employee.

3. The employee fulfil the request.

4. The Customer can view the status of requests whether they are being processed or not.

**Alternate flow:** A pop up message ‘Request cannot be fulfilled’ will be displayed.

**Post condition:** The services are granted to the customer. If not, the system state remains unchanged.

**4.1 INTRODUCTION**

**4.1.1 CHARACTERISTICS OF RISK**

* **Uncertainty ­­-** The risk may or may not happen; that is, there are no 100 percent probable risks.
* **Loss -** If the risk becomes a reality, unwanted consequences or losses will occur.

When risks are analysed, it is important to qualify the level of uncertainty and the degree of loss associated with each risk.

**4.1.2 CATEGORIES OF RISK**

* **Project Risks –**These risks threaten the project plan. That is, if the project risks become real, it is likely that the project schedule will slip and that costs will increase. Project risks identify potential problems which might occur in budget, schedule and staffing (human resource). It also includes project complexity, project size and degree of structural uncertainty.
* **Technical Risks -** These risks threaten the quality and timeliness of the software to be produced. If a technical risk becomes a reality, implementation may become difficult or impossible. Technical risks identify potential design, implementation, interface, verification and maintenance problems.
* **Business Risks –** There are 5 types of Business risks :

1. Market Risk – Building an excellent product or system that no one really wants.

2. Strategic Risk - Building a product that no longer fits into the overall business strategy for the company.

3. Sales Risk – Building a product that the sales force doesn’t understand how to sell.

4. Management Risk – losing the support of senior management due to a change in focus or a change in people.

5. Budget Risk – losing budgetary or personnel commitment.

**Another categorization of risks proposed by Charette are:**

* **Known Risks -** These are risks that can be uncovered after careful evaluation of the project plan, business and technical environment in which the project is being developed.
* **Predictable Risks -** These are extrapolated from past project experience.
* **Unpredictable Risk -** They can and do occur, but they are extremely difficult to identify in advance.
  + 1. **COMPONENTS OF RISK**

The Risk Components are as follows:

* **Performance Risk –** The degree of uncertainty that the product will meet its requirements and be fit for its intended use.
* **Cost Risk –** The degree of uncertainty that the project budget will be maintained.
* **Support Risk –** The degree of uncertainty that the resultant software will be easy to correct, adapt, and enhance.
* **Schedule Risk -** The degree of uncertainty that the project schedule will be maintained and that the product will be delivered on time.

**4.2 RISK IDENTIFICATION**

The checklist can be used for risk identification:

* **Product size –** Risks associated with the overall size of the software to be built or modified.
* **Business impact –** Risks associated with constraints imposed by management or the marketplace.
* **Stake holder characteristics –** Risks associated with the sophistication of the stakeholders and the developer’s ability to communicate with stakeholders in a timely manner.
* **Process definition –** Risks associated with the degree to which the software process has been defined and is followed by the development organization.
* **Development environment –** Risks associated with the availability and quality of the tools to be used to build the product.
* **Technology to be built –** Risks associated with the complexity of the system to be built and the newness of the technology that is packaged by the system.
* **Staff size and experience -**  Risks associated with the overall technical and project experience of the software engineers who will do the work.

**ASSESSING OVERALL PROJECT RISK**

1. Have top software and customer mangers formally committed to support the project? **YES**

2. Are end users enthusiastically committed to the project and the system / product to be built? **YES**

3. Are requirements fully understood by the software engineering team and its customers? **YES**

4. Have customers been involved fully in the definition of requirements? **YES**

5. Do end users have realistic expectations? **YES**

6. Is the project scope stable? **YES**

7. Does the software engineering team have the right mix of skills? **YES**

8. Are project requirements stable? **YES**

9. Does the project team have experience with the technology to be implemented? **YES**

10. Is the number of people on the project team adequate to do the job? **YES**

11. Do all customer / user constituencies agree on the importance of the project and on the requirements for the system / product to be built? **YES**

**TABLE 4.1 RISK TABLE**

|  |  |  |  |
| --- | --- | --- | --- |
| **RISKS** | **CATEGORY** | **PROBABILITY** | **IMPACT** |
| Staff turnover will be high | ST | 70% | Catastrophic |
| Larger number of users than planned | PS | 20% | Marginal |
| Less reuse than planned | PS | 30% | Marginal |
| Delivery deadline will be tightened | BU | 70% | Critical |
| Funding will be lost | CU | 40% | Critical |
| Customer will change requirements | PS | 50% | Catastrophic |
| Lack of training on tools | DE | 60% | Critical |
| Staff inexperienced | ST | 20% | Marginal |

**RISK 1: Customer will change requirements**

**MITIGATION**

The cost of the project would rise if the requirements are changed after the subsequent steps have commenced. To mitigate this risk we would want to:

* Make sure that data collected is correct.
* Put forward a deadline for proposing changes after which changes proposed would be chargeable.

**MONITORING**

While working on the SRS, we would conduct multiple reviews to make sure the requirements are understood well and does not have to be changed much later.

**MANAGEMENT**

In case there is no other option but to make a change in the SRS, the development team must cease their work until the change in the requirements is done. Since prototype model is used in the initial stage, this change may be flexible and thus the project could be continued without much chaos.

**RISK 2: Staff inexperienced**

**MITIGATION**

The project will not be able to achieve desired quality in the given budget within deadline if this risk becomes the reality. To make sure this risk do not happen we will:

* Check the qualifications of the qualifications of all the staff members.
* Personally talk to them regarding their experience.

**MONITORING**

While hiring the staff, we will ensure that person is suitable for our project. He is having proper experience of handling such projects.

**MANAGEMENT**

In case this risk occurs, we will give proper training to the staff before handling the work to them.

Also, we will closely monitor all the actions, so as to ensure that errors are not carried forward.

**RISK 3: Funding will be lost**

**MITIGATION**

The project can lose the budget because of many factors .to mitigate this risk we will:

* Ensure that requirements gathered are correct.
* Ensure that staff has proper experience.
* Ensure that we have backup plan for staff turnover.

**MONITORING**

While working on the project we will ensure that all the tasks are done correctly, and more and more errors are detected so that they are not transferred to the last step, which will ultimately lead to rise in project cost.

**MANAGEMENT**

In case, at any step, if we find that we could lose our budget we will try to plan our rest of the project in such a way that expenditure could be managed.

**RISK 4: Delivery deadline will be tightened**

**MITIGATION**

The delivery of product can be delayed due to various factors:

* Due to high staff turnover.
* Due to various changes in requirements.
* Due to lack of appropriate technology.

**MONITORING**

To ensure that this risk does not happen, we ensure that everything in our project is according to our planning.

**MANAGEMENT**

To manage this risk, we continuously trace our schedules .When we find ourselves lacking we can fasten our rest of the steps to finish our project on time.

**5.1 DFDs**

**5.1.1 CONTEXT LEVEL DIAGRAM**

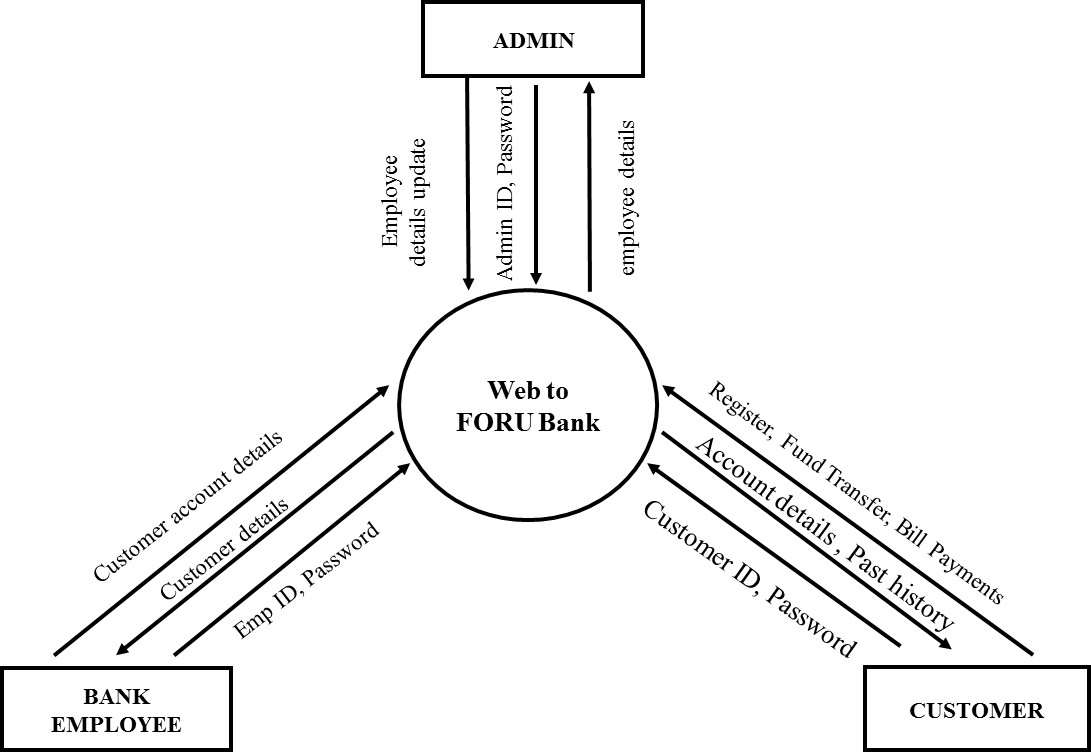


Fig. 5.1 CONTEXT LEVEL DIAGRAM

**5.1.2 LEVEL 1 DFD**

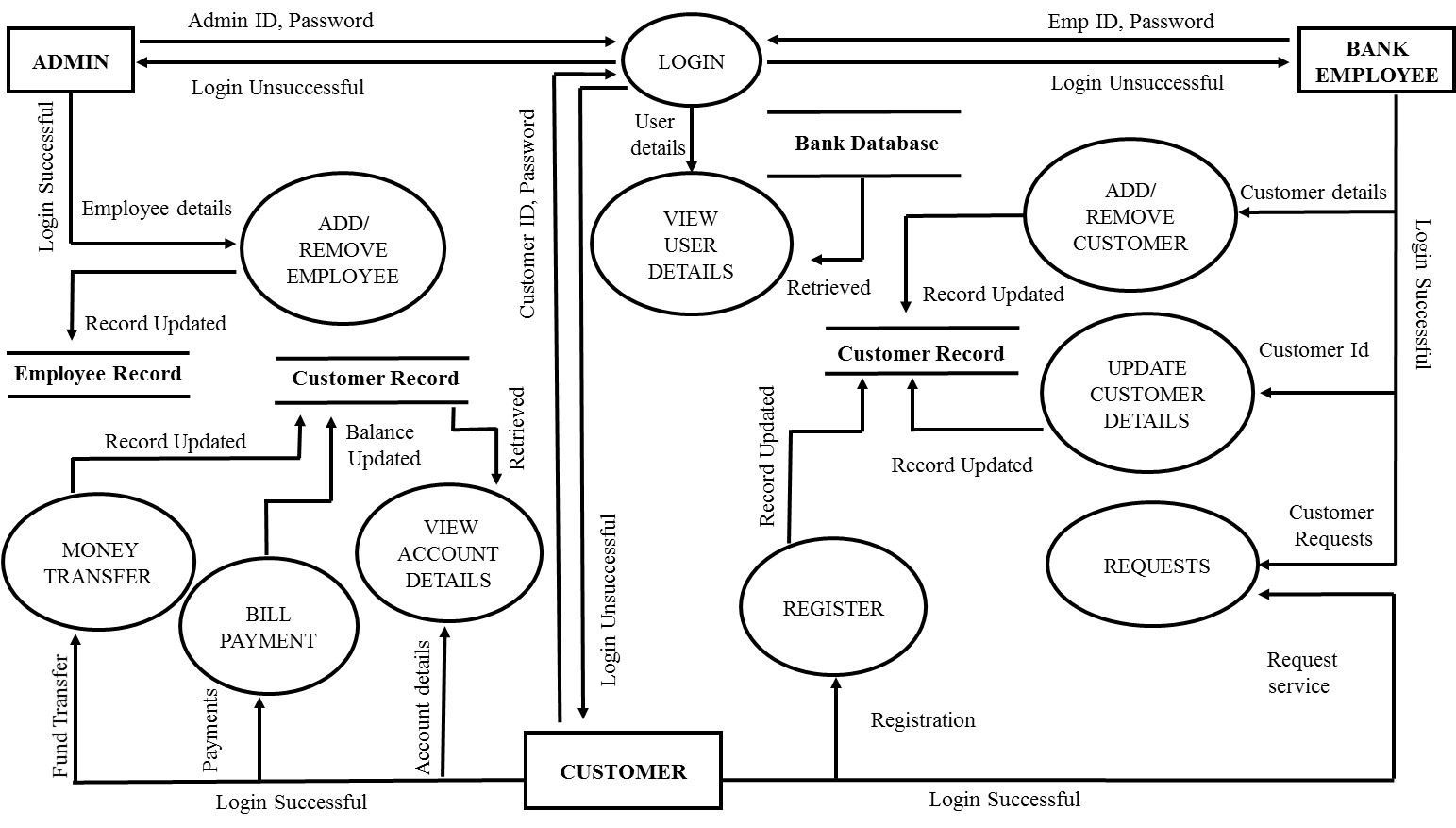


Fig. 5.2 LEVEL-1 DFD

**5.1.3 LEVEL 2 DFD**

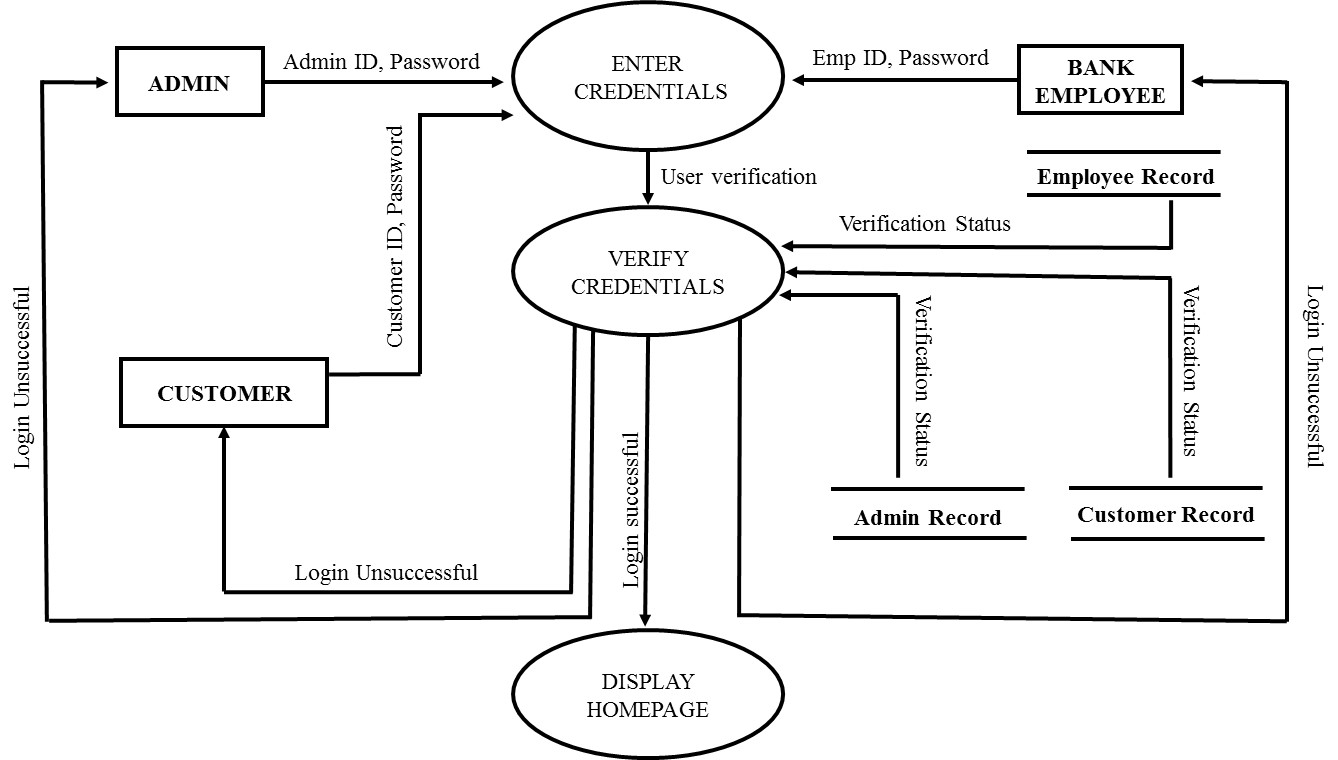


Fig. 5.3 LOGIN

**REGISTER**

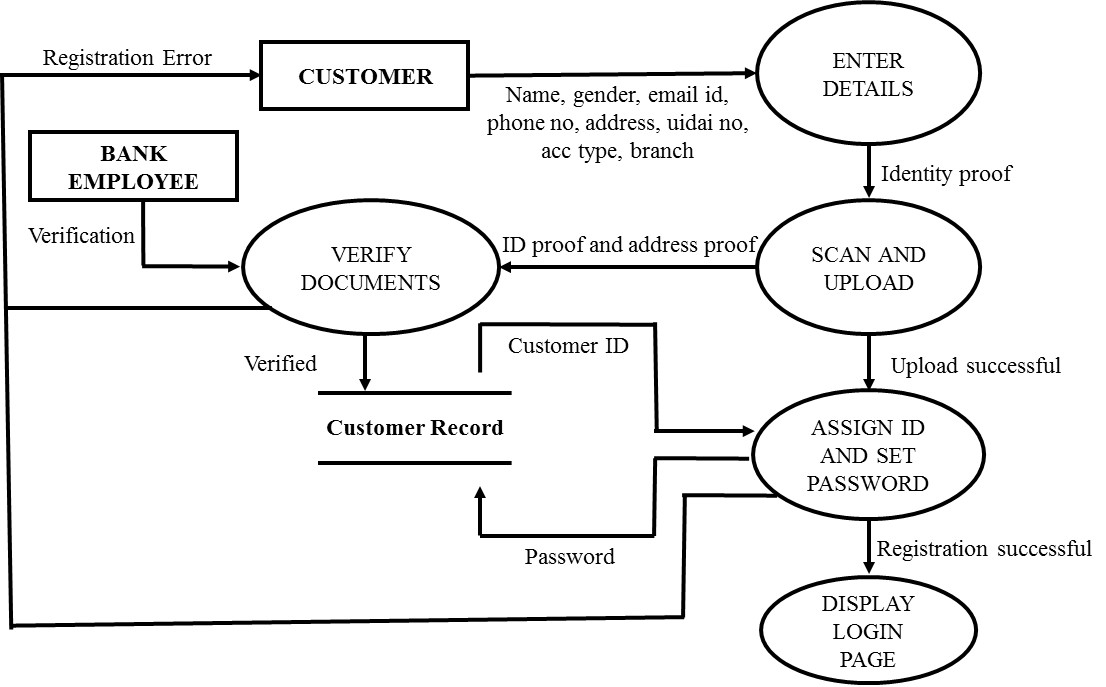


Fig. 5.4 REGISTER

**ADD/REMOVE EMPLOYEE**

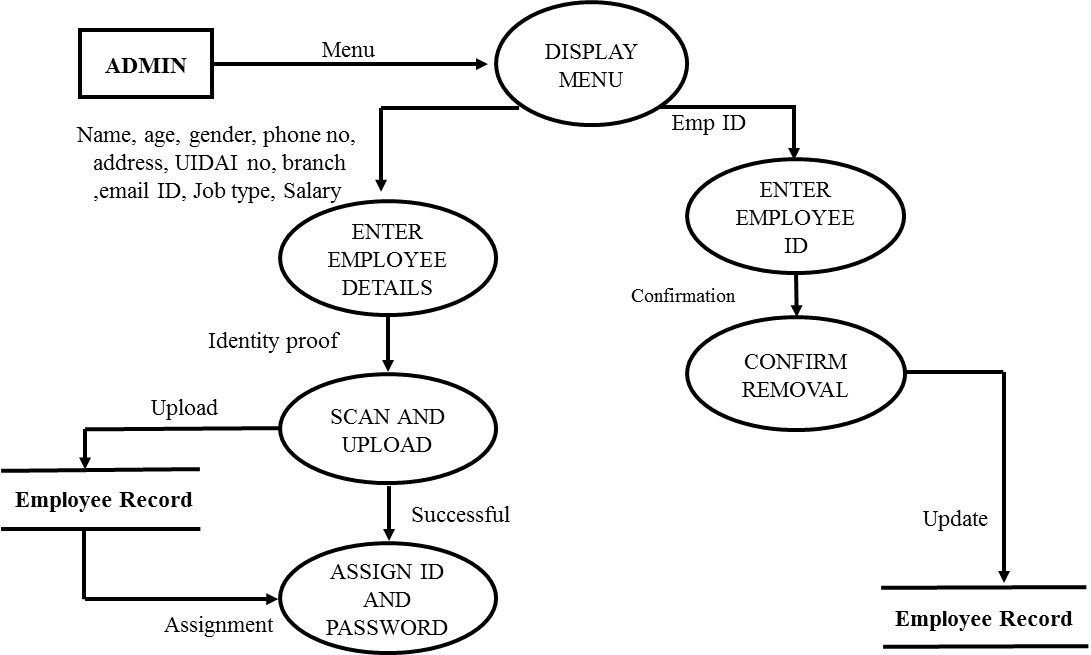


Fig. 5.5 ADD/REMOVE EMPLOYEE

**ADD/REMOVE CUSTOMER**

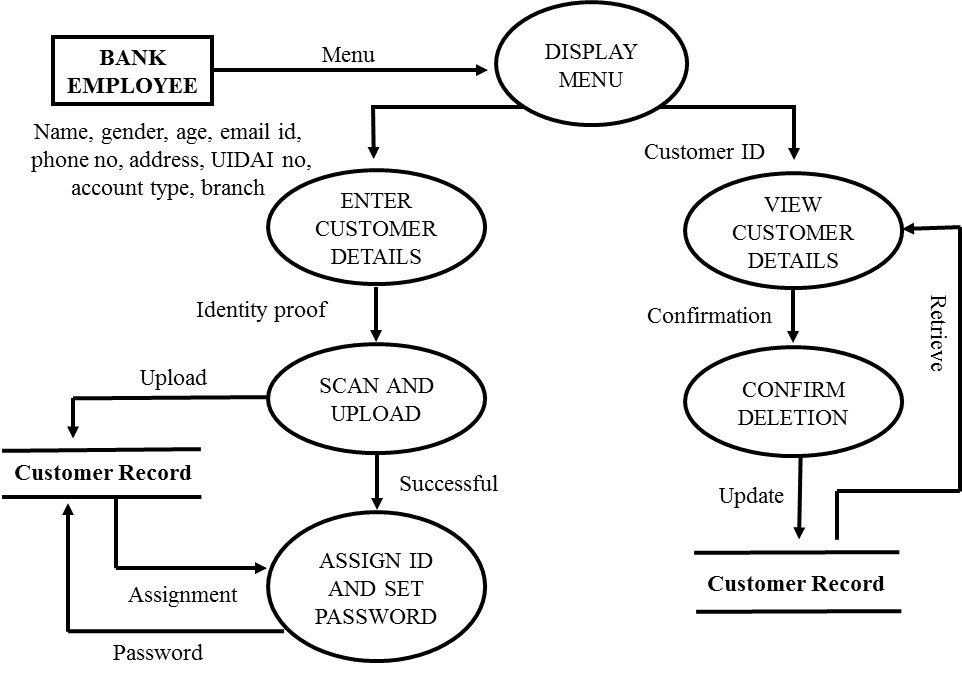


Fig. 5.6 ADD/REMOVE CUSTOMER

**VIEW USER DETAILS**

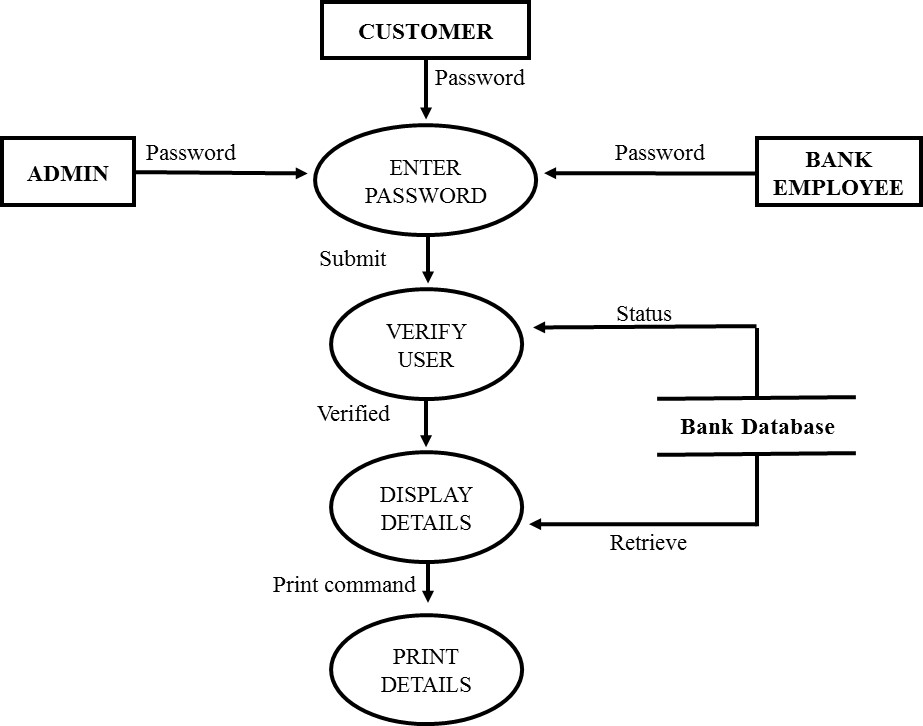


Fig. 5.7 VIEW USER DETAILS

**VIEW ACCOUNT DETAILS**

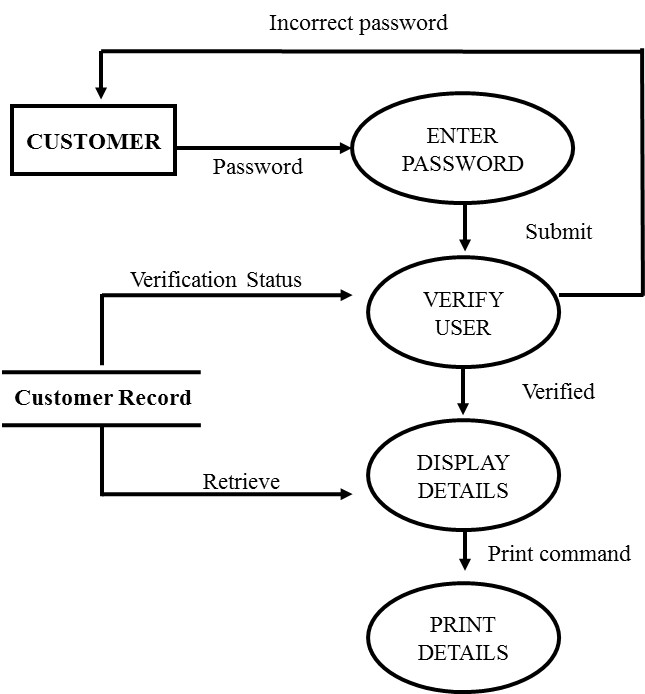


Fig. 5.8 VIEW ACCOUNT DETAILS

**MONEY TRANSFER**

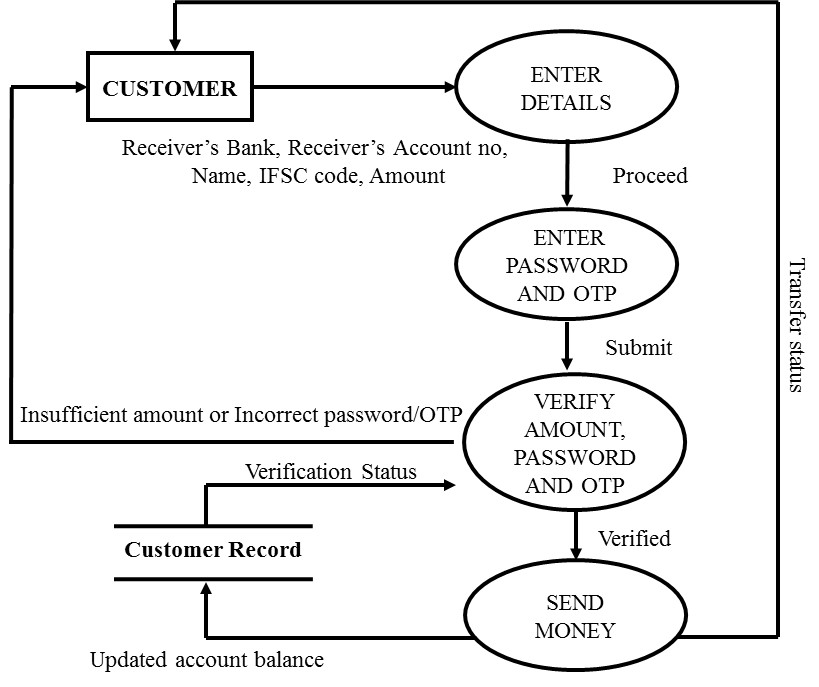


Fig. 5.9 MONEY TRANSFER

**REQUESTS**

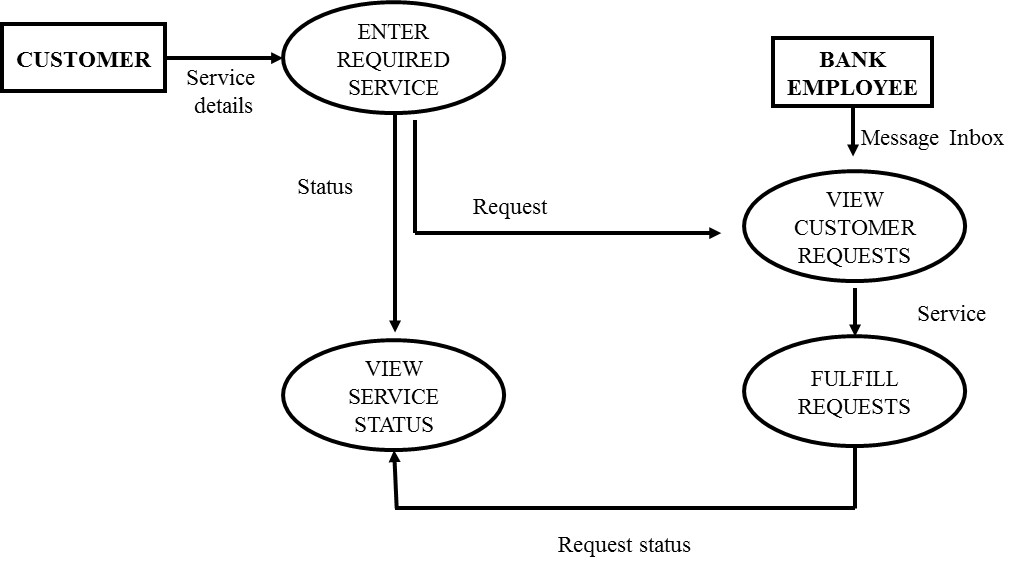


Fig. 5.10 REQUESTS

**BILL PAYMENTS**

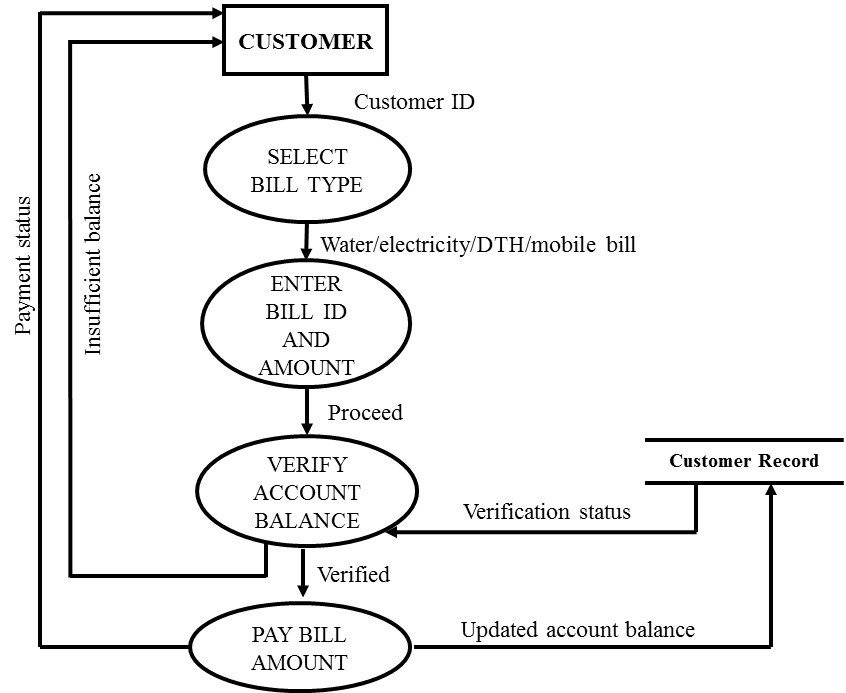


Fig. 5.11 BILL PAYMENTS

**UPDATE CUSTOMER DETAILS**

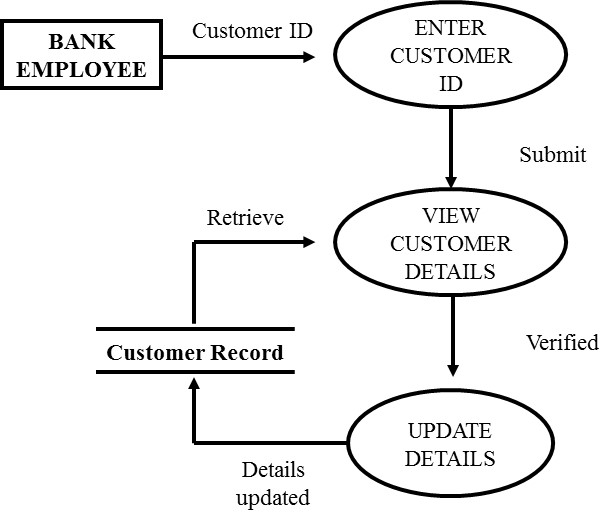


Fig. 5.12 UPDATE CUSTOMER DETAILS

**5.2 DATA DICTIONARY**

Data dictionary (DD) is used to describe the precise structure of data which is not specified by Data Flow Diagrams (DFDs). A DD can specify – data elements, data structures, data flows and data stores.

* **Admin ID** = [ Legal characters | Digits | Special characters ]\*

Legal characters = [A-Z | a-z]

Digits = [0-9]

Special characters = [. | \_ | @ ]

* **Customer ID** = { Digits }\*

Digits = [0-9]

* **Emp ID** = [ Legal characters | Special characters ]\*

Legal characters = [A-Z | a-z]

Special characters = [. | \_ | @ ]

* **Password** = [ Legal characters | Digits | Special characters ]\*

Legal characters = [A-Z | a-z]

Digits = [0-9]

Special characters = [# | @]

* **Name** = First name + ( Middle name ) + Last name

First name = {Legal characters}\*

Middle name = {Legal characters}\*

Last name = {Legal characters}\*

* **UIDAI NO** = digit+ digit+ digit+ digit+ digit+ digit+ digit+ digit+ digit+ digit+ digit+ digit

digit = [0-9]

* **Phone no** = digit+ digit+ digit+ digit+ digit+ digit+ digit+ digit+ digit+ digit

digit = [0-9]

* **Email ID** = [ legal characters + Digits + Special characters ] \*

Legal characters = [A-Z | a-z]

Digits = [0-9]

Special characters = [. || @ | \_ ]

* **Branch** = {Legal characters}\*

Legal characters = [A-Z | a-z]

* **Gender** = [Male | Female | Others]
* **Account Type** = [ Saving | Current ]
* **Age** = digit + digit

digit= [0-9]

* **Address** = {Legal characters | Digits | -}\*

Legal characters = [A-Z | a-z]

Digits = [0-9]

* 1. **LOGICAL DATABASE DESIGN**

**5.3.1 ER DIAGRAM**

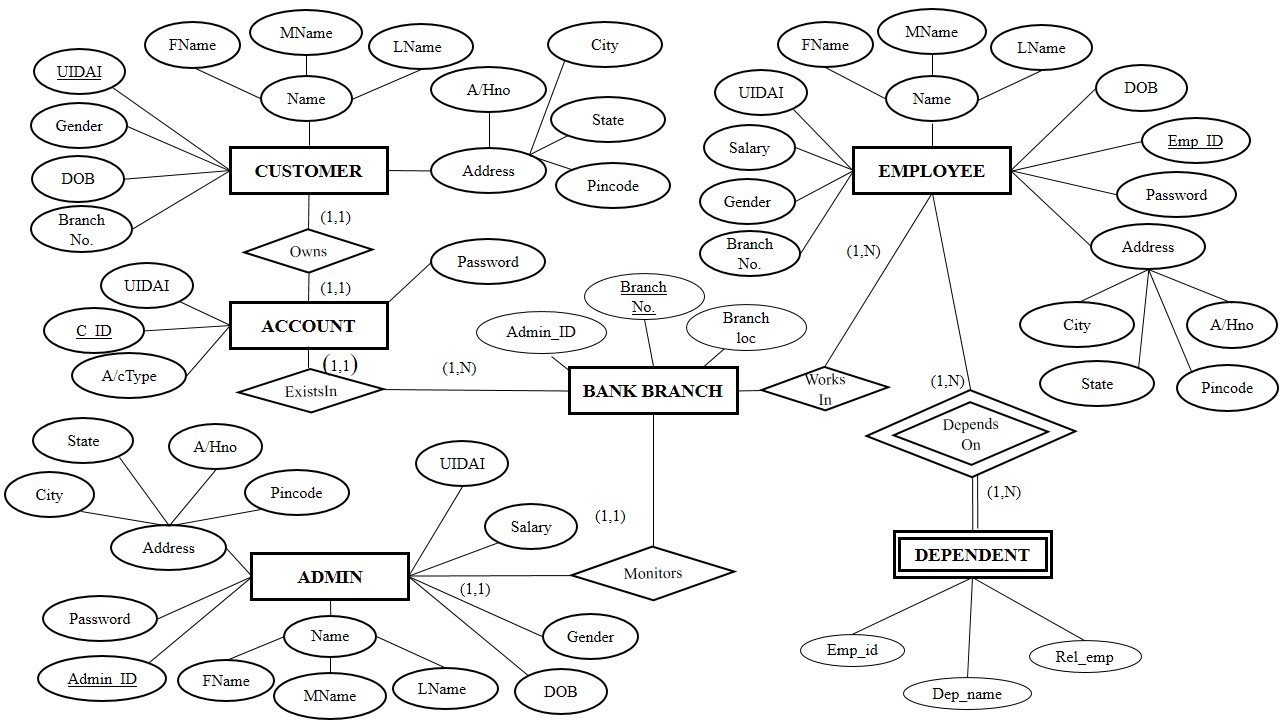
****

Fig. 5.13 ER DIAGRAM

**5.3.2 DATA DESIGN**

A DATA DESIGN is a central store house of information about the system’s data .An Analyst uses Data Design to Collect, document and organize specific facts about the system, including the contents of data flow, data stores, entities and process.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Specifications** | **Constraint** | **Unique** | **Description** |
| First Name | Character | 10 alphabetic characters | Not Null | No | Customer’s First Name |
| Middle Name | Character | 10 alphabetic characters | Null | No | Customer’s Middle Name |
| Last Name | Character | 10 alphabetic characters | Not Null | No | Customer’s Last Name |
| A/H Number | Character | 10 alpha-numeric characters | Null | No | Customer’s Apartment/House No. |
| City | Character | 10 alphabetic characters | Null | No | Customer’s City |
| State | Character | 10 alphabetic characters | Null | No | Customer’s State |
| Pincode | Integer | 10 integer characters | Null | No | Pin code of Customer’s city |
| DOB | Date | ‘MM/DD/YYYY’ format | Null | No | Customer’s Date of Birth |
| Gender | Character | 10 alphabetic characters | Null | No | Customer’s Gender |
| UIDAI | Integer | 10 integer characters | Primary key | Yes | Customer’s Unique identity No. |
| BranchNo. | Integer | 10 integer characters | Not Null | No | Customer’s Branch No. |

**Table No.5.1**:  **Customer** This table contains information about the client who is having an account in any branch of ‘FORU’ Bank.

**Table No. 5.2**: **Employee** This table contains information about the Employee who is working in any branch of ‘FORU’ Bank.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Specifications** | **Constraint** | **Unique** | **Description** |
| First Name | Character | 10 alphabetic characters | Not Null | No | Employee’s First Name |
| Middle Name | Character | 10 alphabetic characters | Null | No | Employee’s Middle Name |
| Last Name | Character | 10 alphabetic characters | Not Null | No | Employee’s Last Name |
| A/H Number | Character | 10 alpha-numeric characters | Null | No | Employee’s A/H No. |
| City | Character | 10 alphabetic characters | Null | No | Employee’s City |
| State | Character | 10 alphabetic characters | Null | No | Employee’s State |
| Pincode | Integer | 10 integer characters | Null | No | Pincode of Employee’s city |
| DOB | Date | ‘MM/DD/YYYY’ format | Null | No | Employee’s Date of Birth |
| Gender | Character | 10 alphabetic characters | Null | No | Employee’s Gender |
| UIDAI | Integer | 10 integer characters | Not Null | Yes | Employee’s Unique identity No. |
| Salary | Integer | 10 integer characters | Not Null | No | Employee’s salary |
| Emp\_id | Character | 10 alpha-numeric characters | Primary key | Yes | Employee’s Login Id |
| Emp\_password | Character | 10 alpha-numeric characters | Not Null | Yes | Employee’s login password |
| BranchNo. | Integer | 10 integer characters | Foreign key | Yes | Employee’s Branch No. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Specifications** | **Constraint** | **Unique** | **Description** |
| First Name | Character | 10 alphabetic characters | Not Null | No | Admin ’s First Name |
| Middle Name | Character | 10 alphabetic characters | Null | No | Admin ’s Middle Name |
| Last Name | Character | 10 alphabetic characters | Not Null | No | Admin ’s Last Name |
| A/H Number | Integer | 10 alpha-numeric characters | Null | No | Admin ’s A/H No. |
| City | Character | 10 alphabetic characters | Null | No | Admin ’s City |
| State | Character | 10 alphabetic characters | Null | No | Admin ’s State |
| Pincode | Integer | 10 integer characters | Null | No | Pincode of Admin ’s city |
| DOB | Date | ‘MM/DD/YYYY’ format | Null | No | Admin ’s Date of Birth |
| Gender | Character | 10 alphabetic characters | Null | No | Admin ’s Gender |
| UIDAI No. | Integer | 10 integer characters | Not Null | Yes | Admin’s Unique identity No. |
| Salary | Integer | 10 integer characters | Not Null | No | Admin ’s salary |
| Admin\_id | Character | 10 alpha-numeric characters | Primary Key | Yes | Admin ’s Login Id |
| Admin\_  password | Character | 10 alpha-numeric characters | Not Null | Yes | Admin ’s Login password |

**Table No. 5.3**: **Admin** This table contains information about the admin who is managing the branch of ‘FORU’ bank.

**Table No. 5.4: Dependent** This table contains information about the Dependents of Employee who is working in any branch of ‘FORU’ Bank.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Specifications** | **Constraints** | **Unique** | **Description** |
| Dep\_name | Character | 10 alphabetic characters | Not Null | Yes | Name of Dependent of Employee |
| Rel\_Emp | Character | 10 alphabetic characters | Not Null | No | Relation of Dependent |
| Emp\_id | Character | 10 alphabetic characters | Partial key | Yes | Emp\_id of Employee |

**Table No. 5.5: Account** This table contains information about the Account of Customer in any branch of ‘FORU Bank.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Specifications** | **Constraints** | **Unique** | **Description** |
| C\_id | Character | 10 alphabetic characters | Primary Key | Yes | Customer’s Id |
| C\_password | Character | 10 alphabetic characters | Not Null | Yes | Customer’s Password |
| Type\_of\_Account | Character | 10 alphabetic characters | Not Null | No | Customer’s Type of Account |

**Table No. 5.6: Bank Branch** This table contains information about the Branches of the bank.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Specifications** | **Constraints** | **Unique** | **Description** |
| BranchNo. | Integer | 10 Integer characters | Primary Key | Yes | Branch No. |
| Branchloc | Characer | 20 alphanumeric characters | Not Null | No | Location of the Branch |
| Admin\_id | Character | 10 alpha-numeric characters | Not Null | Yes | Admin’s Login Id |

**5.3.3 COMPONENT LEVEL DESIGN**

**MONEY TRANSFER**

char password[20], rname[50], IFSC[10], bname[20];

int raccno[12];

long amount, balance, OTP;

take the input from the customer for rname, raccno, IFSC, bname and amount

if amount is less than balance

then

Take password and OTP from the customer

If password and OTP are correct

then

balance = balance - amount

else

Report error in password or OTP

Endif

else

Show an error message “Insufficient balance”

Endif

**6.1 SIZE ESTIMATION (FUNCTION POINT METRICS)**

Project Metrics are used to control and coordinate software engineering process and to improve quality of the software to be produced. Project specific metrics provide indication of productivity and insight into the technical activities. Project specific metrics provide indication of productivity and insight into the technical activities.

Function Oriented Metrics use function point as normalization value. Function points are derived using an empirical relationship based on countable (direct) measure of software`s information domain and assessments of software complexity.

**Table 6.1 FUNCTION POINT WORKSHEET**

|  |  |  |  |
| --- | --- | --- | --- |
| **Information Domain Value** | **Count** | **Weighing Factor**  **(Simple)** | **Weighing Count** |
| Number of inputs | 70 | 3 | 210 |
| Number of outputs | 22 | 4 | 88 |
| Number of user queries | 0 | 3 | 0 |
| Number of internal files | 12 | 7 | 84 |
| Number of external Interfaces | 0 | 5 | 0 |
| TOTAL 382 | | | |

Count total = 382

**TABLE 6.2 VALUE ADJUSTMENT FACTORS (VAFs)**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **FACTORS** | **VALUE** |
| 1 | Does the system require reliable backup and recovery? | 5 |
| 2 | Are specialized data communications required to transfer the information to or from the application? | 1 |
| 3 | Are there distributed processing functions? | 3 |
| 4 | Is performance critical? | 3 |
| 5 | Will the system run in an existing, heavily utilized operational environment? | 3 |
| 6 | Does the system require online data entry? | 4 |
| 7 | Does the online data entry require the input transaction to be built over multiple screens or operations? | 2 |
| 8 | Are the ILFs updated online? | 4 |
| 9 | Are the input/output files, or inquiries complex? | 2 |
| 10 | Is the internal processing complex? | 3 |
| 11 | Is the code designed to be reusable? | 1 |
| 12 | Are conversion and installation included in the design? | 2 |
| 13 | Is the system designed for multiple installations in different organizations? | 4 |
| 14 | Is the application designed to facilitate change and ease of use by the user? | 3 |
|  | Σ(fi) | 40 |

Function Point = Count Total \* [0.65 + [0.01 x Σ (Fi)]

= 382 \* [0.65 + (0.01 \* 40)]

= 382 \* [0.65 + 0.4]

= 382 \* [1.05]

= 401.1 Function point per person month.

**6.2 COCOMO- II MODEL**

Barry Boehm gave a hierarchy of software estimation models called COCOMO i.e., Construction Cost Model (CCM).

The original COCOMO model was widely used in the industry and was later evolved into a comprehensive estimation model called COCOMO-II model.

COCOMO II is a hierarchy of estimation models that address the following areas:

•  **Application composition model** - It is used during the early stages of software engineering, when prototyping of user interfaces assessment of performance and software system interaction are important.

•  **Early design stage model** – It is used once the requirements have been stabilized and basic software architecture has been established.

•  **Post-architecture-stage model** – This model is used during the construction of the software.

These models use sizing information for which three options are available which are:-

1. Object Points
2. Function Points
3. Lines of source code

**OBJECT POINTS** – It is an indirect software measure computed using counts of number of screens at the user interface, number of reusable components called number of reusable components required to build the application. Each object instance is classified into one of the 3 complexity levels- simple, medium or difficult. Complexity is a function of number of source of client and server, data table required to generate screen or report and number of views or sections presented as part of screen/report.

**TABLE 6.3 Complexity and weighing table for object types**

|  |  |  |  |
| --- | --- | --- | --- |
| **Object type** | **Simple** | **Medium** | **Difficult** |
| **Screen** | 1 | 2 | 3 |
| **Report** | 2 | 5 | 8 |
| **3GL components** | - | - | 10 |

NUMBER OF SCREENS = 12

NUMBER OF REPORTS = NIL

NUMBER OF 3GL COMPONENTS USED = 1

NOP (NEW OBJECT POINT) = OBJECT POINT COUNT \* [(100 - %REUSE)/100]

= (12\*1 + 0\*2 + 1\*10) \* [(100-100)/100]

= 22

**TABLE 6.4 PRODUCTIVITY RATE FOR OBJECT POINTS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Developer’s experience or capability** | **Very low** | **Low** | **Normal** | **High** | **Very**  **High** |
| **Environment maturity or capability** | 4 | 7 | 13 | 25 | 50 |

PROD (PRODUCTIVITY RATE) = 7 (assumed)

ESTIMATED EFFORT = NOP / PROD

= 22/7 = 3.1428

= 4 PERSON MONTH

**7.1 PSEUDO CODE**

**MONEY TRANSFER**

char password[20];

char rname[50];

int raccno[12];

char IFSC[10];

char Bname[20];

long amount , balance,OTP;

Money Transfer()

1. Take the input from the customer

2. Input amount, rname, raccno, IFSC, bname

3. if(amount<balance) then

4. Input password and OTP;

5,6 if(Password is correct and OTP is correct) then

7. balance = balance – amount;

Display “Money transferred”

8. else

9. Report error “Incorrect password or OTP”

10. endif

11. else

12. Report error “Amount is greater than account balance “

13. endif

//End money transfer

**7.2 WHITE BOX TESTING**

**7.2.1 FLOW GRAPH**

****

Fig. 7.1 FLOW GRAPH

**7.2.2 INDEPENDENT PATHS FROM THE FLOW GRAPH**

PATH 1: 1 – 2 – 3 – 4 – 5 – 6 – 7 – 10 – 13

PATH 2: 1 – 2 – 3 – 4 – 5 – 8 – 9 – 10 – 13

PATH 3: 1 – 2 – 3 – 4 – 5 – 6 – 8 – 9 – 10 – 13

PATH 4: 1 – 2 – 3 – 11 – 12 – 13

**7.2.3 CYCLOMATIC COMPLEXITY OF THE GRAPH**

1. V(G) = Predicate nodes + 1

=3+1=4

2. V(G ) = Number of regions

= 4

3. V(G) = Edges – Nodes + 2

=14 – 12 +2 = 4

* + 1. **TEST CASES**

**TABLE 7.1 TEST CASES**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE ID** | **INPUT VALUE(S)** | **ACTUAL OUTPUT** | **EXPECTED OUTPUT** |
| 1. | Amount > Balance | Error “Amount is greater than balance”. | Error “Amount is greater than balance”. |
| 2. | Amount < Balance  , OTP is correct and Password is correct | Money transferred | Money transferred |
| 3. | Amount < Balance  , OTP is incorrect and Password is incorrect | Error “Incorrect password or/and OTP” | Error “Incorrect password or/and OTP” |
| 4. | Amount < Balance  , OTP is correct and Password is incorrect | Error “Incorrect password or/and OTP” | Error “Incorrect password or/and OTP” |
| 5. | Amount < Balance  , OTP is incorrect and Password is correct | Error “Incorrect password or/and OTP” | Error “Incorrect password or/and OTP” |

**8.1 CONCLUSION**

Through this project we have tried to change the traditional way of banking. Traditionally, we had to go to bank to open our account, or to access it. This website allow users to open a bank account or to manage it online thus, reducing the wastage of time and effort.

This website allows a user to open an account in the bank or to perform several activities required.

**Advantages:**

* **Allow customers to open an account online**

To open an account user have to fill the required registration form and upload the required documents. The documents are verified by the employee and customer is added.

* **Allow users to perform activities like pay the bills such as electricity bill, water bill or to recharge phone or DTH.**

The customer chooses the activity to be performed from various options, fulfill the required formalities and the task is completed.

* **Allow users to transfer money to another account in the same bank or in different bank.**

The user is asked to fill the details of the account holder and provide OTP and password, if the details provided are correct, the money is transferred.

* **Allow users to ask for services.**

The user can ask for services required like check book, the SMS service (If any transaction happens customer is notified through message on the phone) or E – mail service ( customer is notified through mail for every transaction).

The services are processed by the employees .The customer can view the status of the request whether it has been fulfilled by the employee or not.

* **Allow users to view their account details.**

The customers can view their details which includes all the personal details like Name, Address, Phone number etc. and Account details like Account number, Type of account and balance.

User can print the details if required.

* **Allow Customers to apply online for closing their account.**

In the request section, user can send a request to close the account. The request can be processed by the employee and account can be closed.

**8.2 LIMITATIONS OF THE SYSTEM**

Although many of the activities can be performed online, but many of the activities are still there that cannot be performed by our website .Some are:

* **User cannot apply for loan**

Although user is a permanent customer of the bank, he cannot apply for loan online. He will have to go to bank and fulfil the requirements offline in order to get the loan.

* **User cannot pay for Airways or Railways reservations**

The website could have reduced very much load if there could have been an option of making reservations directly from bank account but user is not provided with any such option.

* **User cannot pay several government taxes**

It could have been an additional feature of the bank website where we could pay for several taxes like house tax, income tax or LPG pipeline bills directly through bank account. We will try to add these features in next releases.

* **User cannot pay for movie tickets, restaurants etc.**

We should have provided an option for paying for paying for movie tickets and restaurants making the user less dependent on cash.

* **Admin cannot add employees online to the bank**

An employee can add a customer online to the bank but an admin can only add employee to website. To add an employee to website means he has been selected to be an employee and now admin will grant the rights and duties to be performed online by the employee. The procedure of selecting an employee is offline. However adding an employee online would have made things easier for the candidates.

**8.3 REFERENCES**

1. R.S. Pressman, Software Engineering: A Practitioner’s Approach (7th Edition), McGraw-Hill, 2009.
2. P. Jalote, an Integrated Approach to Software Engineering (2nd Edition), Narosa Publishing House, 2003.
3. **ANNEXURES**



Fig. 9.1 WELCOME PAGE

Number of inputs = 0

Number of outputs = 1

Number of user queries = 0

Number of internal files = 0

Number of external Interfaces = 0



Fig. 9.2 LOGIN

Number of inputs = 6

Number of outputs = 4

Number of user queries = 0

Number of internal files = 3

Number of external Interfaces = 0

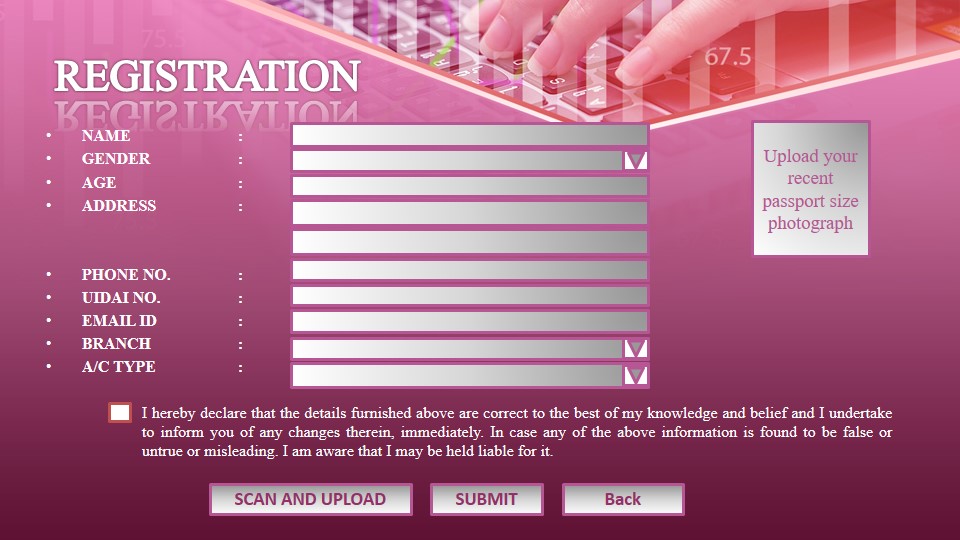


Fig. 9.3 REGISTRATION

Number of inputs = 13

Number of outputs = 4

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0

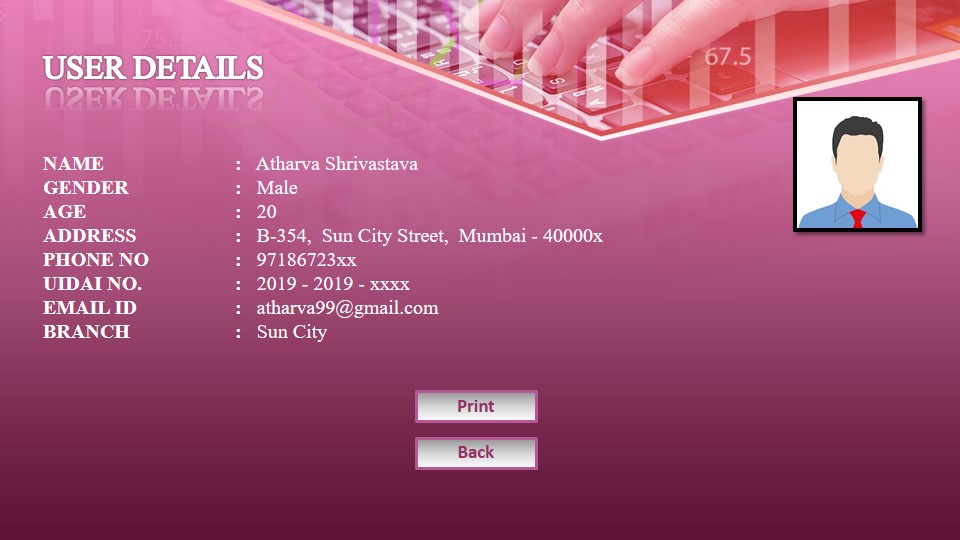


Fig. 9.4 VIEW USER DETAILS

Number of inputs = 2

Number of outputs = 1

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0



Fig. 9.5 VIEW ACCOUNT DETAILS

Number of inputs = 2

Number of outputs = 1

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0



Fig. 9.6 MONEY TRANSFER

Number of inputs = 8

Number of outputs = 1

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0

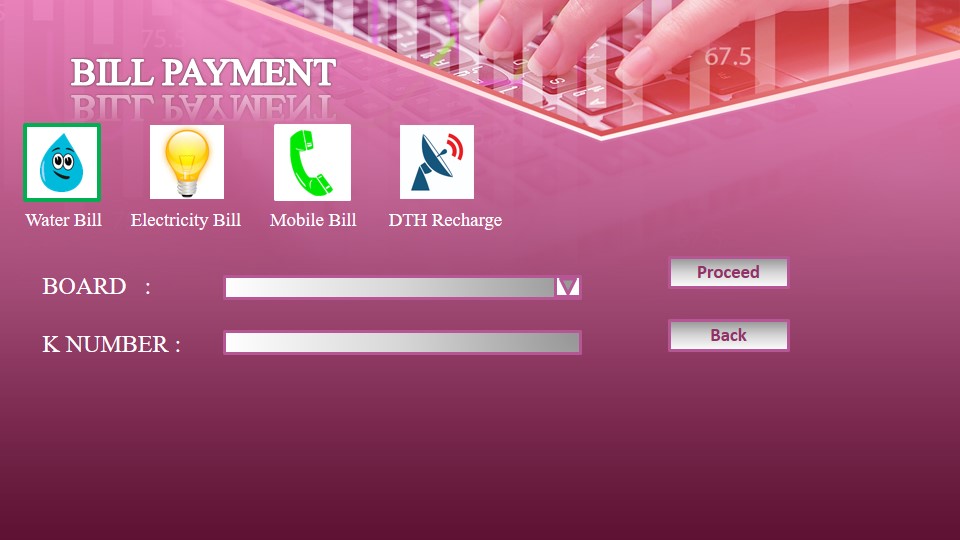


Fig. 9.7 BILL PAYMENT

Number of inputs = 5

Number of outputs = 2

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0



Fig. 9.8 ADD CUSTOMER

Number of inputs = 12

Number of outputs = 3

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0



Fig. 9.9 ADD EMPLOYEE

Number of inputs = 13

Number of outputs = 3

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0

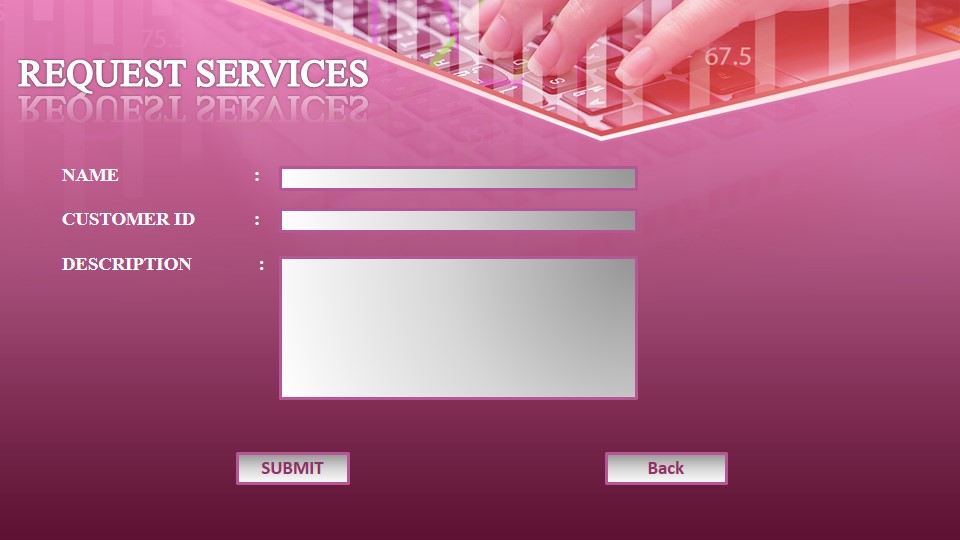


Fig. 9.10 SERVICES

Number of inputs = 5

Number of outputs = 0

Number of user queries = 0

Number of internal files = 0

Number of external Interfaces = 0

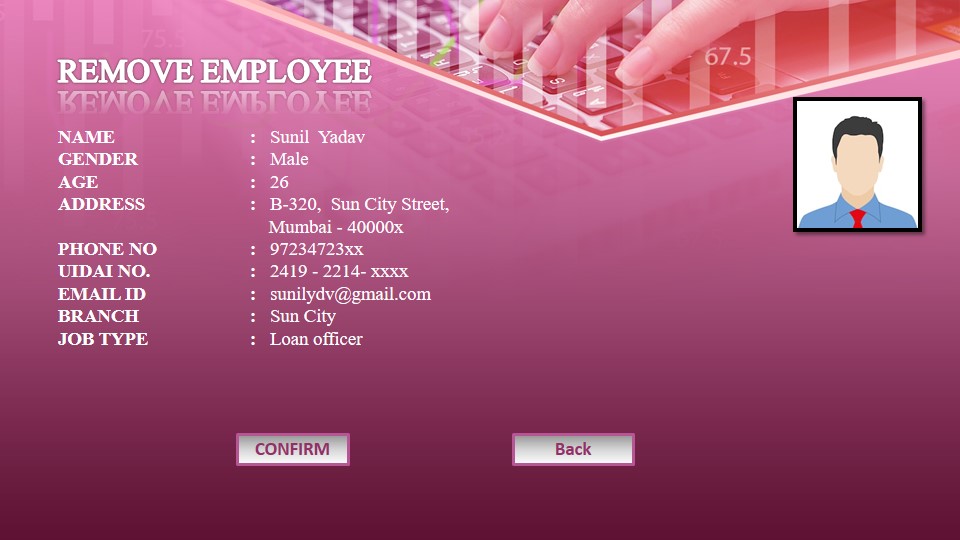


Fig. 9.11 REMOVE EMPLOYEE

Number of inputs = 2

Number of outputs = 1

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0

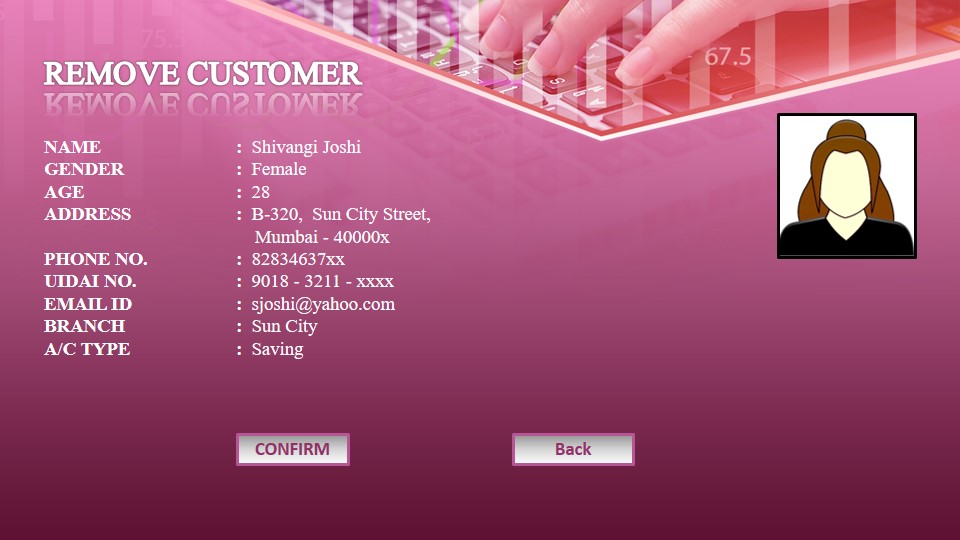


Fig. 9.12 REMOVE CUSTOMER

Number of inputs = 2

Number of outputs = 1

Number of user queries = 0

Number of internal files = 1

Number of external Interfaces = 0