VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi: 590 018

A Database Management Systems Mini Project report on

"Bank Management System"

Submitted in partial fulfillment of the requirement for the award of Degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

By

S Pavan Reddy 1AY18CS098

Under the guidance of

Prof. Ancy Thomas

Prof. Rashmi Kulkarni,

Prof. Chidananda T

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
ACHARYA INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)

2020-2021

ACHARYA INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)
Soladevanahalli, Bangalore – 560090

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Certificate

Certified that the Database Management Systems mini project entitled **“BANK DATABASE MANAGEMENT SYSTEM”** is a bonafide work carried out by **S PAVAN REDDY (1AY18CS098)** of 5th semester in partial fulfillment for the award of degree of **Bachelor of Engineering in Computer Science & Engineering of the Visvesvaraya Technological University**, **Belagavi**, during the year **2020-2021.** It is certified that all corrections/ suggestions indicated for internal assessments have been incorporated in the Report deposited in the departmental library. The Mini Project report has been approved as it satisfies the academic requirements in respect of Mini Project work prescribed for the **Bachelor of Engineering Degree**.

Signature of Guides **Signature of H.O.D**

Name of the examiners **Signature with date**

1.

2.

ABSTRACT

This is a Bank Database Management System designed to be accessed through User login, Cashier Login and Manager Login. In this project with the help of MySQL server (where the whole database resides) seven different tables are created (login, branch, feedback, useraccounts, transaction, notice, deletelogs). Every table has an option of inserting a new entry of data, deleting an old inserted data and updating an existing one. Framework used for the project are and Microsoft MySQL Community Server and Apache server using XAMPP. For Scripting language PHP has been used. For Front end Technology HTML, CSS, JavaScript, Bootstrap has been used. This project can be used in banks after enhancements to make to suit the real world scenarios.

ACKNOWLEDGEMENT

I express my gratitude to our institution and management for providing us with good infrastructure, laboratory, facilities and inspiring staff whose gratitude was of immense help in completion of this seminar successfully.

I express my sincere gratitude to our principal, **Dr. Prakash M R** for providing required environment and valuable support for developing this mini project.

My sincere thanks to **Dr. Prashanth C M,** Head of the Department, Computer Science and Engineering, Acharya Institute of Technology for his valuable support and for rendering us resources for this mini project work.

I express my gratitude to **Prof. Ancy Thomas, Prof. Rashmi Kulkarni, Prof. Chidananda T,** Assistant Professors, Dept. Computer Science and Engineering, Acharya Institute of Technology who guided me with valuable suggestions in completing this mini-project at every stage.

My gratitude thanks should be rendered to many people who helped me in all possible ways.

S Pavan Reddy

(1AY18CS098)

Contents

Chapter 1: Introduction 1
Chapter 2: System Requirements 2
2.1 Hardware Requirements 2
2.2 Software Requirements 2
2.3 Functional Requirements 2
2.4 Non-Functional Requirements 3
Chapter 3: Design 4
3.1 Entity-Relationship Diagram 4
3.2 Relation Cardinality 5
3.3 Schema Diagram 5
Chapter 4: Implementation 8
4.1 Relations 8
4.1.1 Login 8
4.1.2 Branch 8
4.1.3 Feedback 9
4.1.4 UserAccounts 10
4.1.5 Transaction 11
4.1.6 Notice 12
4.1.7 Screenshots 13
4.2 Triggers 17
4.3 Usage of database 19
Chapter 5: Results and Discussion 21
Chapter 6: Conclusion and Future enhancements 37
6.1 Conclusion 37
6.2 Future Enhancements 37
Chapter 7: References… 38

Table of Figures

Figure 3.1: E-R diagram of Bank Management System 4
Figure 3.2: Schema Diagram of Bank Management System 6
Figure 3.3: Enhanced E-R Diagram of Bank Management System 7
Figure 4.1: Attributes of the relation ‘Login’ 8
Figure 4.2: Attributes of the relation ‘Branch’ 9
Figure 4.4: Attributes of the relation ‘UserAccounts’ 10
Figure 4.5: Attributes of the relation ‘Transaction’ 12
Figure 4.6: Attributes of the relation ‘Notice’ 12
Figure 4.7: Login Table 13
Figure 4.8: Branch Table 13
Figure 4.9: Feedback Table 14
Figure 4.10: UserAccounts Table 15
Figure 4.11: Transaction Table 16
Figure 4.12 Notice Table 16
Figure 4.13 Deletelogs Table 18
Figure 4.14: Deletelogs table after deleting an user account. 19
Figure 4.15: Normal execution of database. 20
Figure 4.16: Normal execution of database (continued...) 20
Figure 5.1: Home/Login Page 21
Figure 5.2: User Login 21
Figure 5.3: User Home Page 22
Figure 5.4: Accounts Page 22
Figure 5.5: Statement Page 23
Figure 5.6: Fund Transfer Page 23
Figure 5.7: Fund Transfer Page (Continued…) 24
Figure 5.8: Fund Transfer Page (After successful fund transfer) 24

Figure 5.9: Page when user enters their own account number as beneficiary account number..25
Figure 5.10: Bank Notice/Notification Page..25
Figure 5.11: Help/Feedback Page 26
Figure 5.12: User giving feedback..26
Figure 5.13: Page after user give feedback 27
Figure 5.14: Manager Login 27
Figure 5.15: Manager Home Page and User Account Control Page. 28
Figure 5.16: User account information page for user with account #10001...............28
Figure 5.17: Manager sending notice to a user in response to feedback. 29
Figure 5.18: Page after manager sends notice. 29
Figure 5.19: Page after Manager deletes an account. 30
Figure 5.20: Page for Manager to add or remove cashiers. 30
Figure 5.21: Page after manager deletes a cashier...31
Figure 5.21: Pop-up box for adding new cashier...31
Figure 5.22: Manager adding new cashier details. 32
Figure 5.23 Page showing new cashier added. 32
Figure 5.24: Add new User account page 33
Figure 5.25: Manager adding new User account. 33
Figure 5.26: Feedback page where manager can view user feedbacks......................34
Figure 5.27: Feedback page after deleting a feedback...34
Figure 5.28: Cashier Login Page 35
Figure 5.29: Cashier Home page 35
Figure 5.30: Cashier Transaction Page (After entering account #). 36
Figure 5.31: Page shown after any kind of successful transaction. 36

Chapter 1

Introduction

The title of the project is “Bank Management system”. Bank management systems play an essential role in the current banking system. Bank authorities all over the world are engaged in a lot of day to day administrative and banking activities to manage and provide a better banking experience to customers effectively. However, maintaining and keeping track of bank transactions is not an easy process in the fast-growing world. It requires hard work and it often is time consuming.

To better perform the bank transactions and administrative activities of a bank and to assure security of the money by means of authentication and authorization, banks utilize Bank Management Systems nowadays. Such applications often offer many features that helps to enhance the performances of banks with minimum effort. A bank management software does it by avoiding the manual paper work and automating many banking and administrative activities involved in banking.

The main aim of this project is to develop software for Bank Account Management System. This project has been developed to carry out the processes easily and quickly, which is not possible with the manuals systems, which are overcome by this software.

Chapter 2

System Requirements

2.1 Hardware Requirements

The hardware requirements for this project are as follows:

- **Processor:** Any x86 instruction set processor
- **RAM:** 64MB or more
- **Storage:** 1GB or more
- **GPU:** AMD Radeon Graphics Processor (0x68E0) or above

2.2 Software Requirements

The software requirements for this project are as follows:

- **Operating System:** Any operating system that supports the x86 software used here.
- **Front-end:** HTML 5, CSS 3, JavaScript ES6, Bootstrap 4
- **Back-end:** MYSQL, PHP
- **Framework:** XAMPP stack solution for web development v3.2.2 or above
  - Apache 2.4.29
  - MariaDB 10.1.30
  - PHP 7.2.1
  - phpMyAdmin 4.7.4

2.3 Functional Requirements

The functional requirements for the project are as follows:

- User should be able to login to his/her account.
- User should be able view his/her account balance.
- User should be able to transfer money to any bank account.
- User should be able to view notices.

- User should be able to give feedback.
- User should be able to request help.
- User should be able to view past transaction history.
- Cashier should be able to login to his/her account.
- Cashier should be able to withdraw and deposit users money.
- Manager should be able to login to his/her account.
- Manager should be able to view all accounts in bank.
- Manager should be able to add new cashiers and remove existing ones when required.
- Manager should be able to add new user accounts and remove existing ones if required.
- Manager should be able to view feedback and respond to them.
- Manager should be able to send notices to us.

2.4 Non-Functional Requirements

Some of the non-functional requirements of our proposed software are:

- **Speed:** The software must be quick enough to deliver data as quickly as possible, and ensure a constant frame rate of at least 60 frames per second.
- **Security:** The software must be protected against malicious users attempting to attack it using nefarious techniques like timing attacks, SQL injection, etc.
- **Reliability:** The software must be reliable enough to be expected to run 24x7 with minimal supervision.
- **Data protection:** The software must ensure that the data it stores is adequately secured and cannot be tampered with.
- **Data consistency:** The software must try to maintain only the bare minimum of redundant data. Wherever such redundancies are present, the software must ensure that the data is consistent.

Chapter 3

Design

3.1 Entity-Relationship Diagram

The entity-relationship diagram, also known as the ER diagram, is a high-level database design, which shows the database in a diagrammatic approach. It consists of entities, relationships, attributes, and associations. The ER-diagram depicts the various relationships among entities, considering each object as entity. Entity is represented as rectangle shape and relationship represented as diamond shape. The ER diagram for the project is shown in Figure 3.1 below:

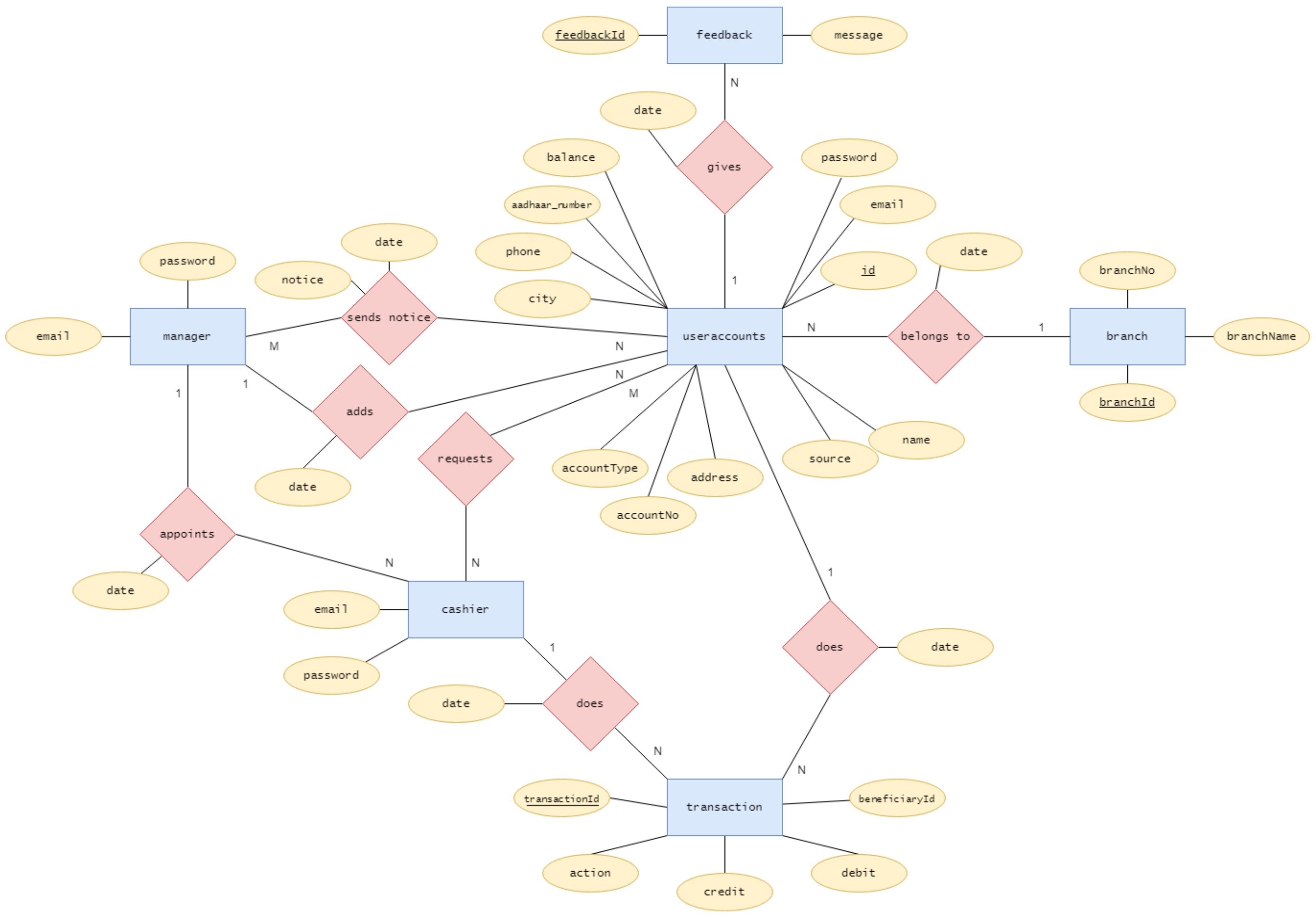


Figure 3.1: E-R diagram of Bank Management System

3.2 Relation Cardinality

- **1:N**
    - A branch can have many accounts but a particular account belongs to a unique branch.
    - A user can do many transactions but a particular transaction can’t be initiated by different accounts.
    - A user can give any number of feedbacks to the bank but a particular feedback can be generated by only one user.
    - A manager can appoint any number of cashiers but a specific cashier can be appointed by a single manager.
    - A manager can add any number of accounts of users but a particular account is added by a single manager.
    - A cashier can do any number of transactions but a specific transaction can be done by a single cashier.
- **M:N**
    - A manager can send notices to any number of users and also a user can receive notices from any number of managers.
    - A user can request any number of cashiers to do a transaction and also a cashier can receive requests from any number of users to do transactions.

3.3 Schema Diagram and Enhanced Entity-Relationship Diagram

A schema diagram is an illustrative display of most aspects of a database schema. A schema construct is a component of the schema, or an object within the schema. The schema diagram of our database system is illustrated on the next page, in Figure 3.2.

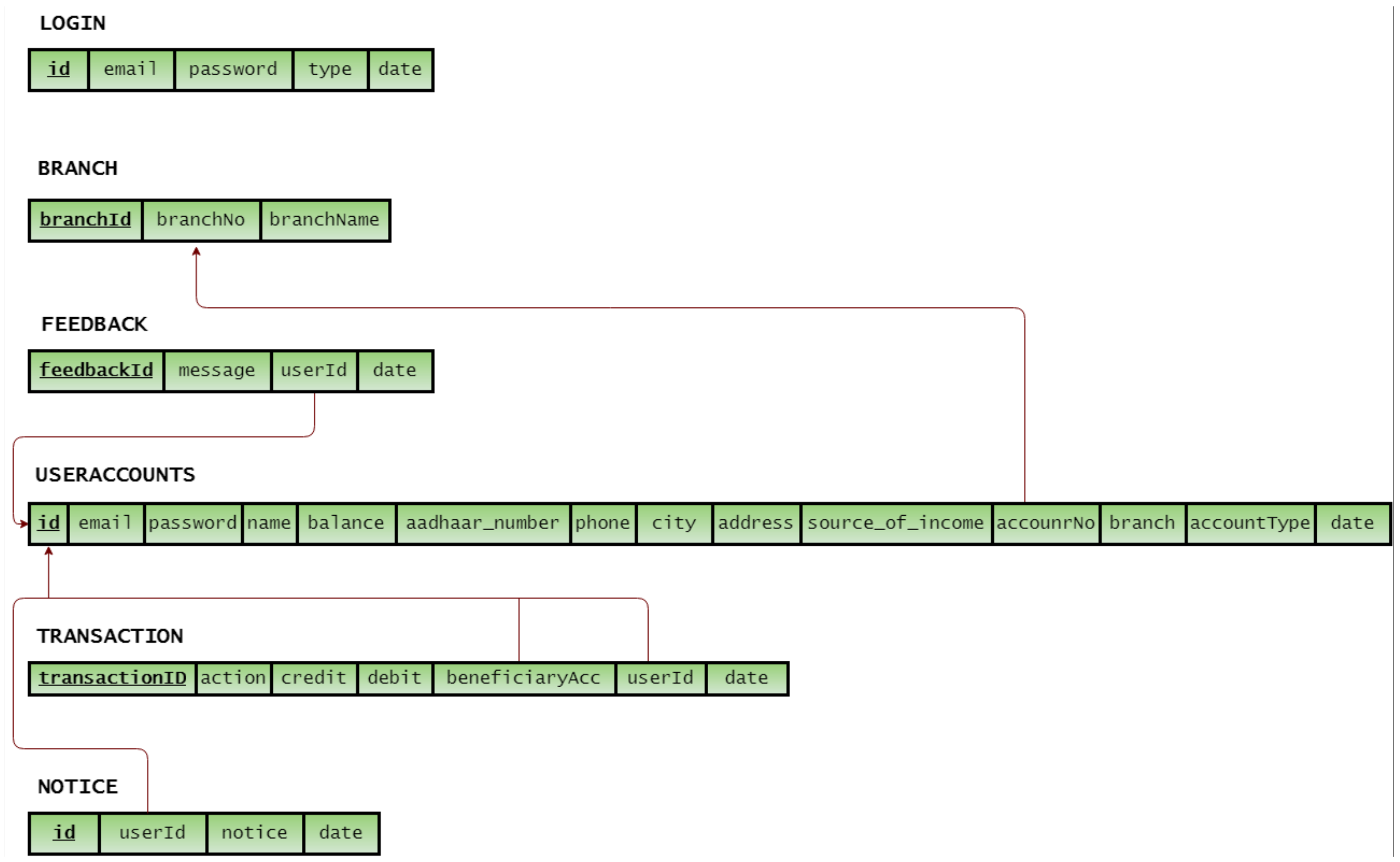


Figure 3.2: Schema Diagram of Bank Management System

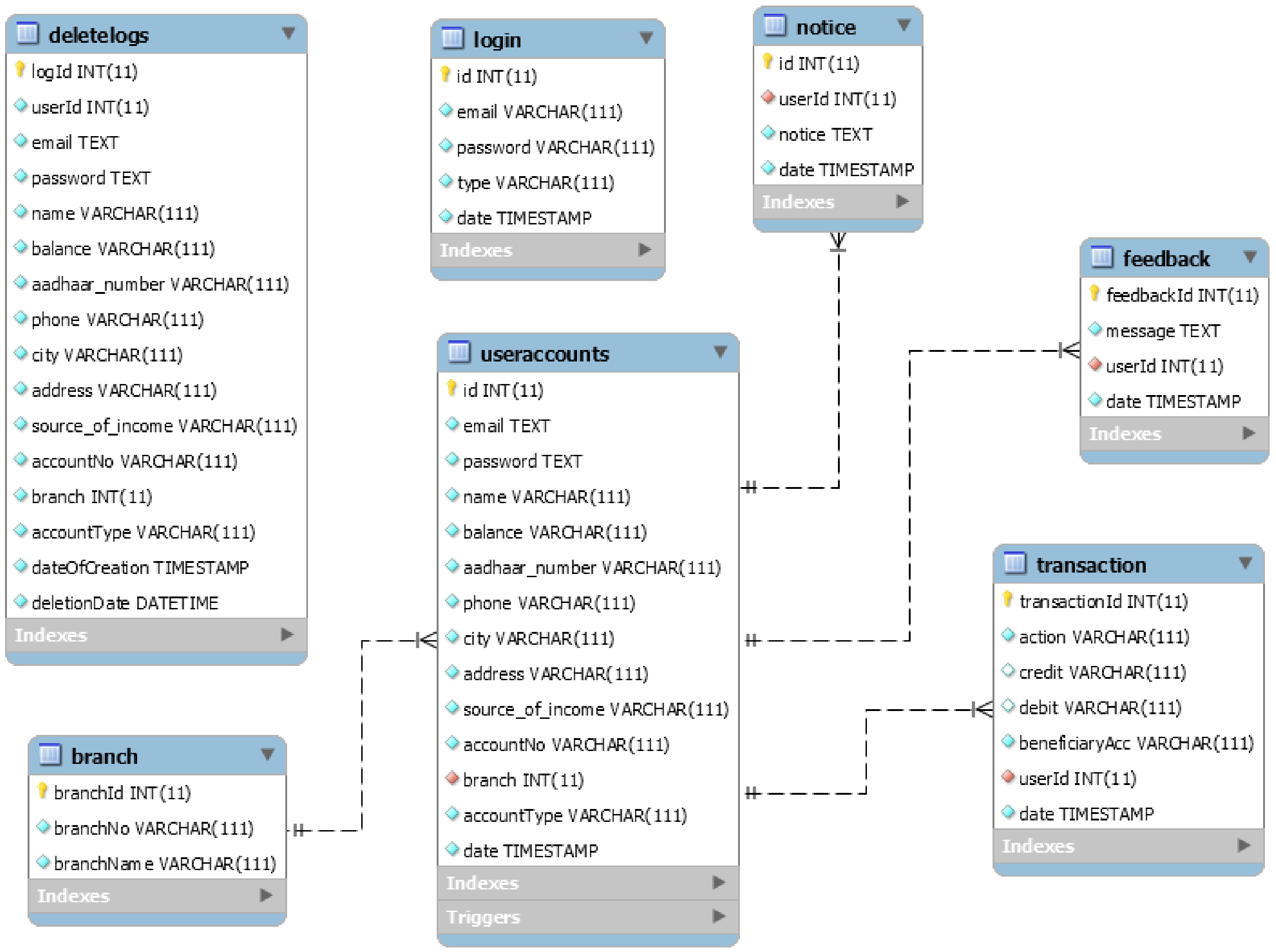


Figure 3.3: Enhanced E-R Diagram of Bank Management System

Chapter 4

Implementation

The whole database is created in MariaDB. All front-end related queries were executed using prepared statements in PHP to avoid the possibility of SQL injection.

4.1 Relations

4.1.1 Login

This table contains login information of the Bank staff.

Sl. No.	Name	Datatype	Description
1	id	int(11)	Auto-incrementing unique id for user login
2	email	varchar(111)	User e-mail address
3	password	varchar(111)	User password
4	type	varchar(111)	Users type (manager/cashier)
5	date	timestamp	Timestamp of insertion of record

Figure 4.1: Attributes of the relation 'Login'

```
CREATE TABLE `login` (
  `id` int(11) NOT NULL,
  `email` varchar(111) NOT NULL,
  `password` varchar(111) NOT NULL,
  `type` varchar(111) NOT NULL,
  `date` timestamp NOT NULL DEFAULT Current_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `login`
  ADD PRIMARY KEY (`id`);

ALTER TABLE `login`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;
```

4.1.2 Branch

This table stores branch information given below.

Sl. No.	Name	Datatype	Description
1	branchId	int(11)	Auto incrementing unique id for branch
2	branchNo	varchar(111)	Branch Number
3	branchName	varchar(111)	Branch Name

Figure 4.2: Attributes of the relation 'Branch'

```
CREATE TABLE `branch` (
  `branchId` int(11) NOT NULL,
  `branchNo` varchar(111) NOT NULL,
  `branchName` varchar(111) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `branch`
  ADD PRIMARY KEY (`branchId`);

ALTER TABLE `branch`
  MODIFY `branchId` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;
```

4.1.3 Feedback

Users/Customers can give feedback to bank whenever they want to and the information regarding the same is stored here.

Sl. No.	Name	Datatype	Description
1	feedbackId	int	Auto incrementing unique id for feedback
2	message	text	The feedback message
3	userId	int(11)	ID of the user giving feedback
4	date	timestamp	Timestamp of feedback

Figure 4.3: Attributes of the relation 'Feedback'

```
CREATE TABLE `feedback` (
  `feedbackId` int(11) NOT NULL,
  `message` text NOT NULL,
  `userId` int(11) NOT NULL,
  `date` timestamp NOT NULL DEFAULT Current_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `feedback`
  ADD PRIMARY KEY (`feedbackId`);

ALTER TABLE `feedback`
  MODIFY `feedbackId` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;
```

4.1.4 UserAccounts

The useraccounts relation contains data about users and their accounts. This data helps in keeping track of customer personal information and account information like balance, account number, etc.

Sl. No.	Name	Datatype	Description
1	id	int(11)	Auto incrementing unique id for useraccount
2	email	text	Registered email address of user
3	password	text	Password of bank management system for user email
4	name	varchar(111),	Name of account holder
5	balance	varchar(111),	Balance in bank
6	aadhaar_numbar	varchar(111)	Unique ID and proof of citizenship of user in India
7	phone	varchar(111)	Phone number of account holder
8	city	varchar(111)	City of residence of user
9	address	varchar(111)	Address of user
10	source_of_income	varchar(111)	Source of income of user
11	accountNo	varchar(111)	Account Number of user's bank account
12	branch	int(11)	Branch to which account belongs
13	accountType	varchar(111)	Type of bank account (Savings/Current)
14	date	timestamp	Timestamp of account creation

Figure 4.4: Attributes of the relation 'UserAccounts'

```
CREATE TABLE `useraccounts` (
  `id` int(11) NOT NULL,
  `email` text NOT NULL,
  `password` text NOT NULL,
  `name` varchar(111) NOT NULL,
  `balance` varchar(111) NOT NULL,
  `aadhaar_number` varchar(111) NOT NULL,
  `phone` varchar(111) NOT NULL,
  `city` varchar(111) NOT NULL,
  `address` varchar(111) NOT NULL,
  `source_of_income` varchar(111) NOT NULL,
  `accountNo` varchar(111) NOT NULL,
  `branch` int(11) NOT NULL,
  `accountType` varchar(111) NOT NULL,
  `date` timestamp NOT NULL DEFAULT Current_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `useraccounts`
  ADD PRIMARY KEY (`id`);
ALTER TABLE `useraccounts`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=8;
```

4.1.5 Transaction

The transaction table, as the name indicates stores all transactions' information such as the id of the useraccount which has initiated it, the beneficiary account number, type of transaction, amount credited or debited from an account and timestamp of transaction.

Sl. No.	Name	Datatype	Description
1	transactionId	int(11)	Auto incrementing unique id for a transaction
2	action	varchar(11)	Type of transaction
3	credit	varchar(11)	Amount credited
4	debit	varchar(11)	Amount debited
5	beneficiaryAcc	varchar(111)	Beneficiary account number
6	userid	int(11)	ID of useraccount which initiated the transaction
7	date	timestamp	Timestamp of transaction

Figure 4.5: Attributes of the relation 'Transaction'

```
CREATE TABLE `transaction` (
  `transactionId` int(11) NOT NULL,
  `action` varchar(111) NOT NULL,
  `credit` varchar(111) DEFAULT NULL,
  `debit` varchar(111) DEFAULT NULL,
  `beneficiaryAcc` varchar(111) NOT NULL,
  `userId` int(11) NOT NULL,
  `date` timestamp NOT NULL DEFAULT Current_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `transaction`
  ADD PRIMARY KEY (`transactionId`);
ALTER TABLE `transaction`
  MODIFY `transactionId` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=28;
```

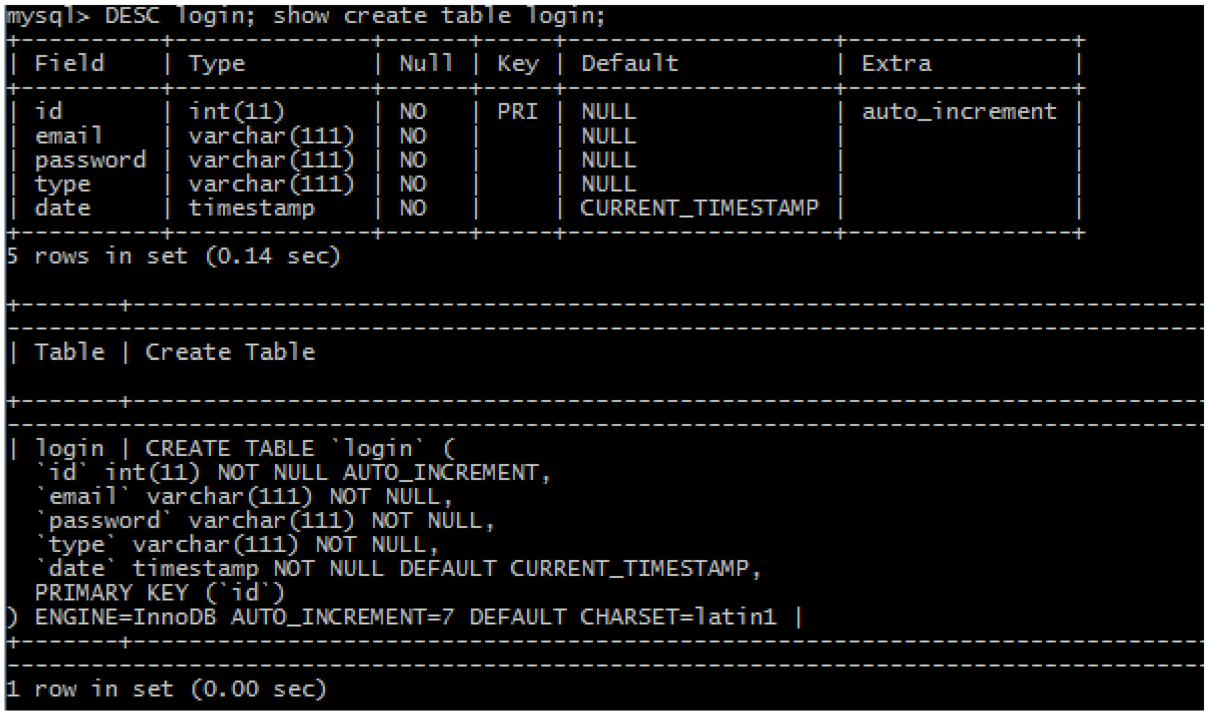
4.1.6 Notice

Users can receive notices from the bank via managers and the notice information is stored in this table.

Sl. No.	Name	Datatype	Description
1	id	int(11)	Auto incrementing unique id for notice
2	userId	int(11)	ID of recipient user
3	notice	text	Notice message
4	date	timestamp	Timestamp of Notice

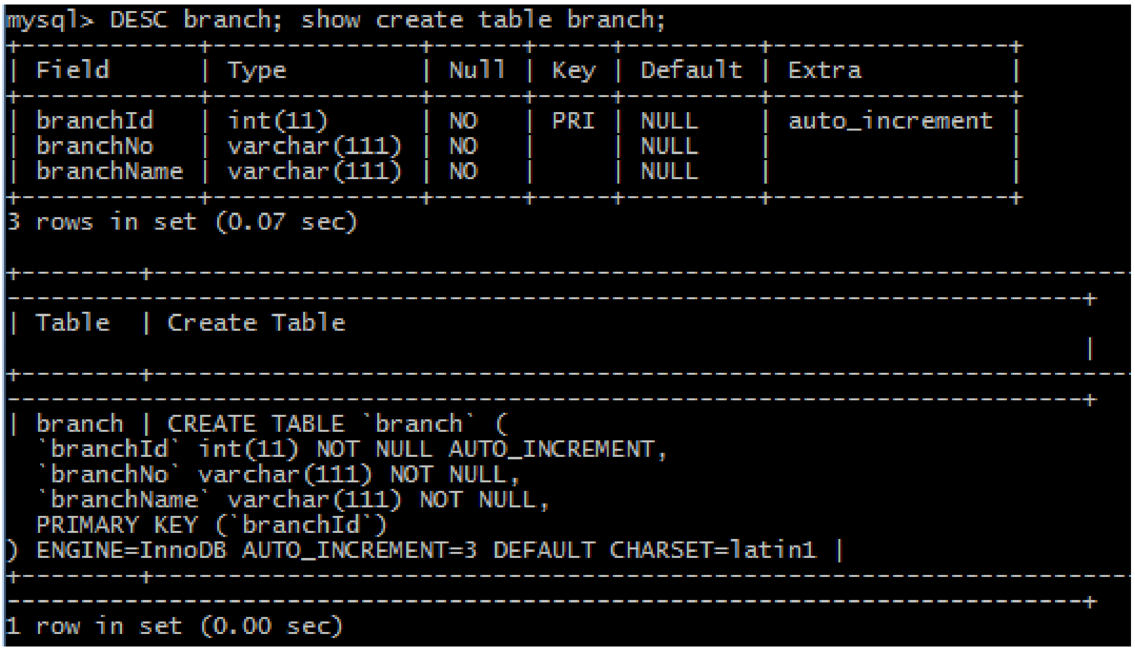
Figure 4.6: Attributes of the relation 'Notice'

4.1.7 Screenshots



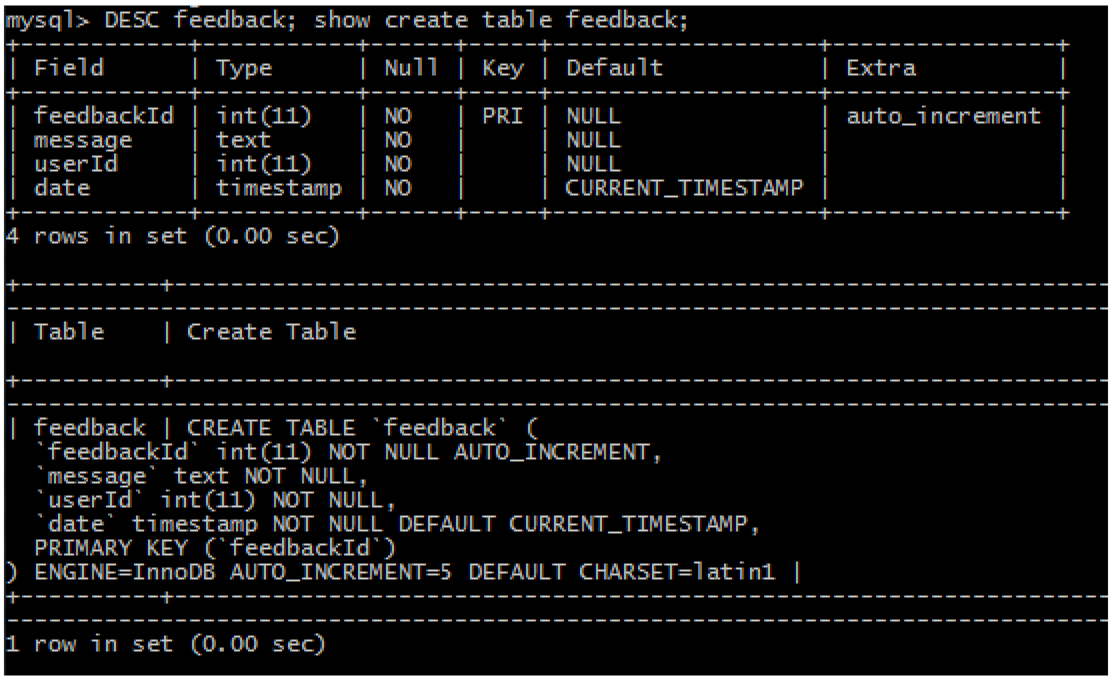
```
mysql> DESC login; show create table login;
+----------+--------------+------+-----+-------------------+----------------+
| Field    | Type         | Null | Key | Default           | Extra          |
+----------+--------------+------+-----+-------------------+----------------+
| id       | int(11)      | NO   | PRI | NULL              | auto_increment |
| email    | varchar(111) | NO   |     | NULL              |                |
| password | varchar(111) | NO   |     | NULL              |                |
| type     | varchar(111) | NO   |     | NULL              |                |
| date     | timestamp    | NO   |     | CURRENT_TIMESTAMP |                |
+----------+--------------+------+-----+-------------------+----------------+
5 rows in set (0.14 sec)

+-------+------------------------------------------------------------------------
---------------------------------------------------------------------------------
| Table | Create Table

+-------+------------------------------------------------------------------------
---------------------------------------------------------------------------------
| login | CREATE TABLE `login` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `email` varchar(111) NOT NULL,
  `password` varchar(111) NOT NULL,
  `type` varchar(111) NOT NULL,
  `date` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=latin1 |
+-------+------------------------------------------------------------------------
---------------------------------------------------------------------------------
1 row in set (0.00 sec)
```

Figure 4.7: Login Table



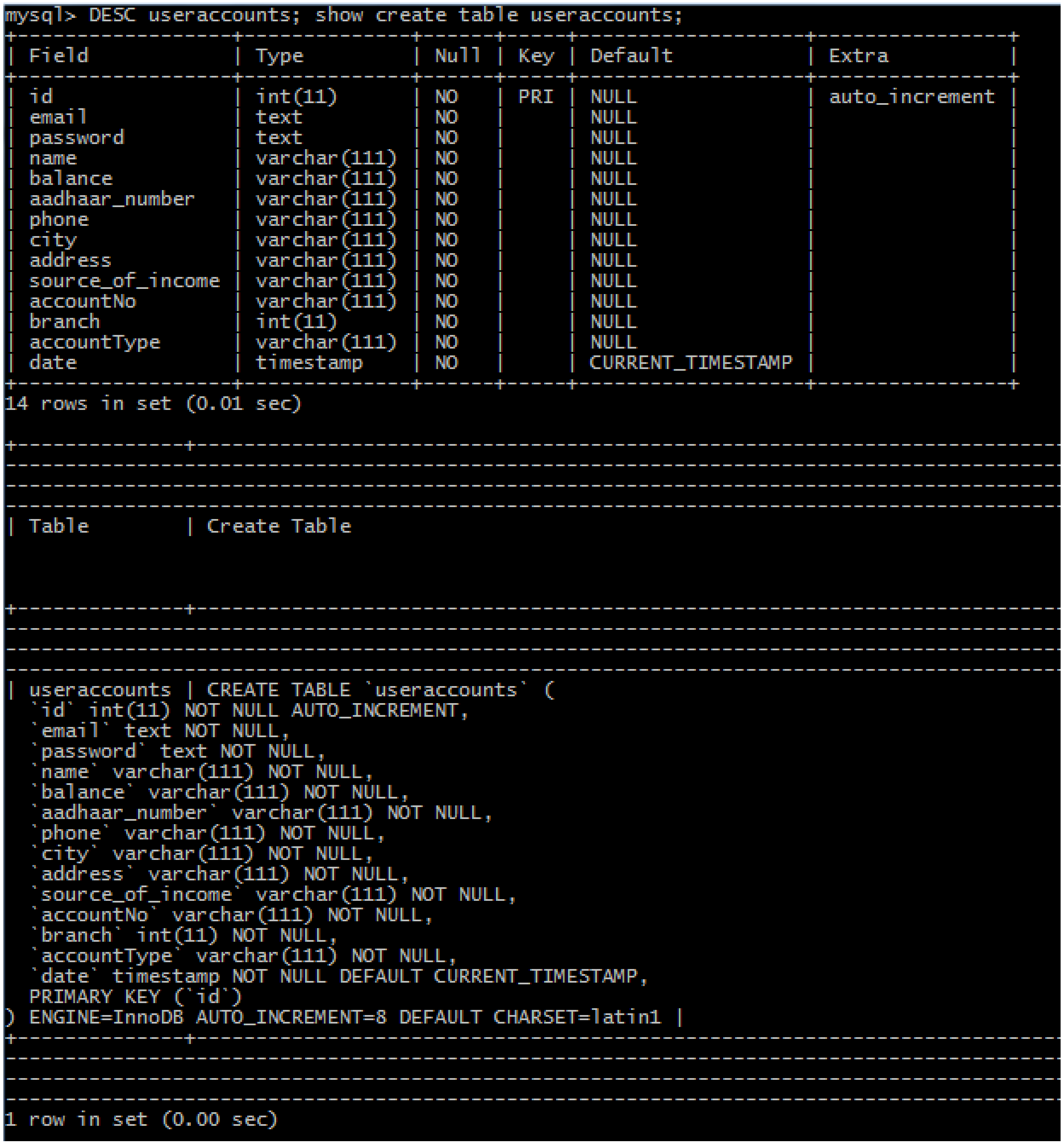
```
mysql> DESC branch; show create table branch;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| branchId   | int(11)      | NO   | PRI | NULL    | auto_increment |
| branchNo   | varchar(111) | NO   |     | NULL    |                |
| branchName | varchar(111) | NO   |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
3 rows in set (0.07 sec)

+--------+----------------------------------------------------------------------
---------------------------------------------------------------------------+
| Table  | Create Table
                                                                           |
+--------+----------------------------------------------------------------------
---------------------------------------------------------------------------+
| branch | CREATE TABLE `branch` (
  `branchId` int(11) NOT NULL AUTO_INCREMENT,
  `branchNo` varchar(111) NOT NULL,
  `branchName` varchar(111) NOT NULL,
  PRIMARY KEY (`branchId`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=latin1 |
+--------+----------------------------------------------------------------------
---------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

Figure 4.8: Branch Table



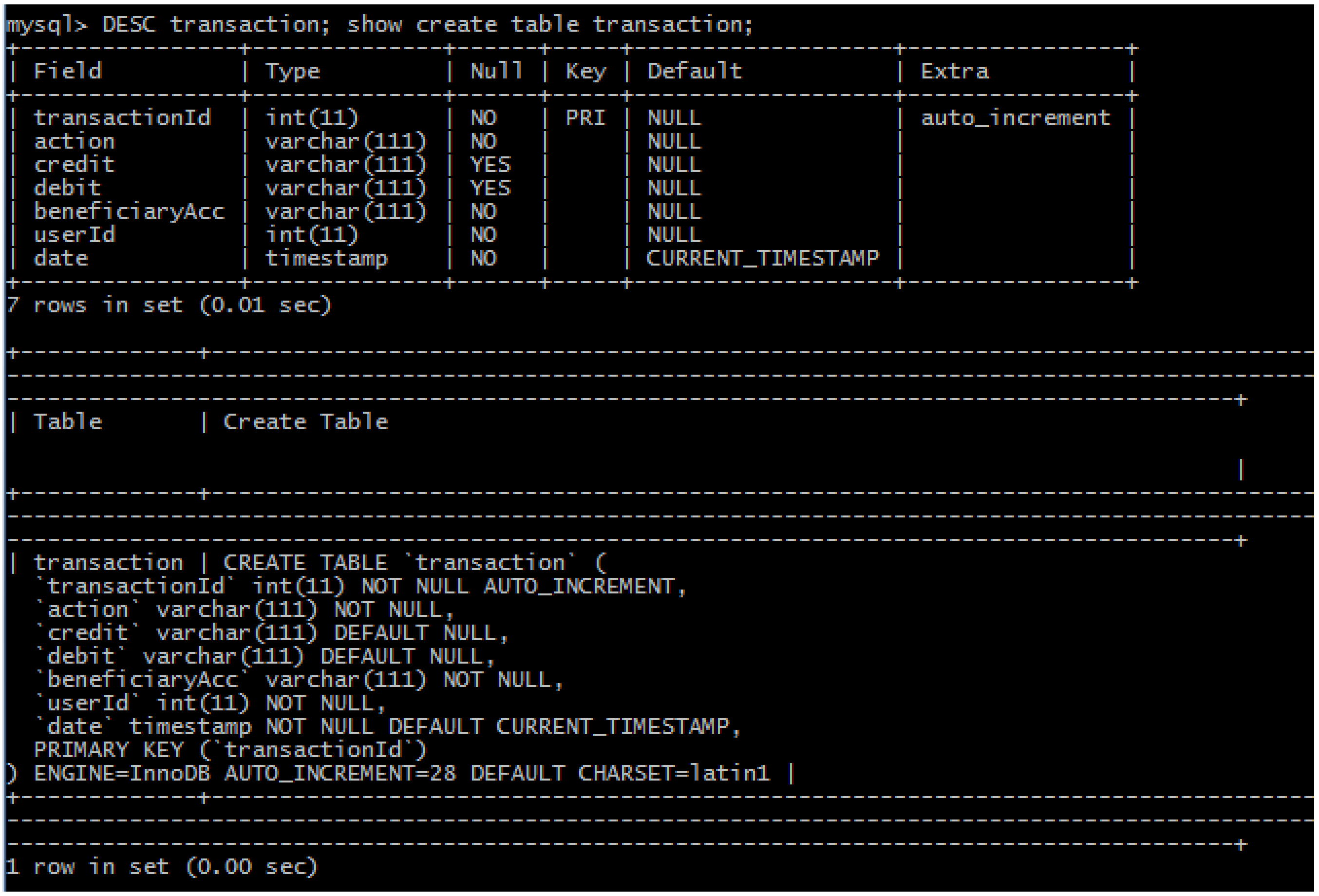
```
mysql> DESC feedback; show create table feedback;
+------------+-----------+------+-----+-------------------+----------------+
| Field      | Type      | Null | Key | Default           | Extra          |
+------------+-----------+------+-----+-------------------+----------------+
| feedbackId | int(11)   | NO   | PRI | NULL              | auto_increment |
| message    | text      | NO   |     | NULL              |                |
| userId     | int(11)   | NO   |     | NULL              |                |
| date       | timestamp | NO   |     | CURRENT_TIMESTAMP |                |
+------------+-----------+------+-----+-------------------+----------------+
4 rows in set (0.00 sec)

+----------+------------------------------------------------------------------
------------------------------------------------------------------------------
| Table    | Create Table

+----------+------------------------------------------------------------------
------------------------------------------------------------------------------
| feedback | CREATE TABLE `feedback` (
  `feedbackId` int(11) NOT NULL AUTO_INCREMENT,
  `message` text NOT NULL,
  `userId` int(11) NOT NULL,
  `date` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`feedbackId`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=latin1 |
+----------+------------------------------------------------------------------
------------------------------------------------------------------------------
1 row in set (0.00 sec)
```

Figure 4.9: Feedback Table



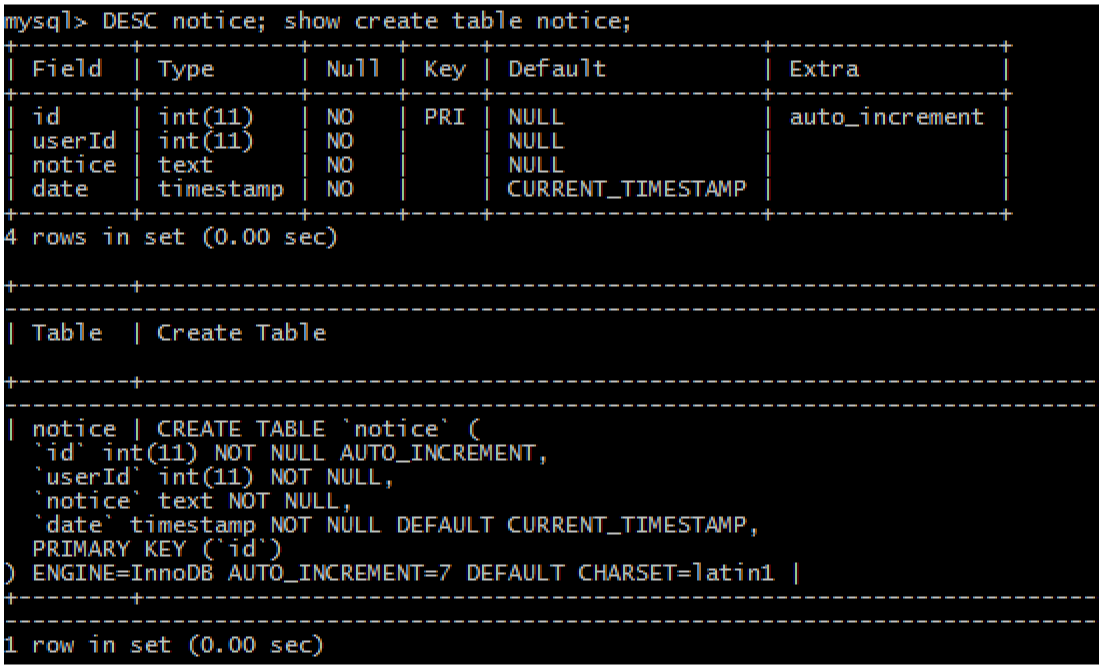
```
mysql> DESC useraccounts; show create table useraccounts;
+-----------------+--------------+------+-----+-------------------+----------------+
| Field           | Type         | Null | Key | Default           | Extra          |
+-----------------+--------------+------+-----+-------------------+----------------+
| id              | int(11)      | NO   | PRI | NULL              | auto_increment |
| email           | text         | NO   |     | NULL              |                |
| password        | text         | NO   |     | NULL              |                |
| name            | varchar(111) | NO   |     | NULL              |                |
| balance         | varchar(111) | NO   |     | NULL              |                |
| aadhaar_number  | varchar(111) | NO   |     | NULL              |                |
| phone           | varchar(111) | NO   |     | NULL              |                |
| city            | varchar(111) | NO   |     | NULL              |                |
| address         | varchar(111) | NO   |     | NULL              |                |
| source_of_income | varchar(111) | NO   |     | NULL              |                |
| accountNo       | varchar(111) | NO   |     | NULL              |                |
| branch          | int(11)      | NO   |     | NULL              |                |
| accountType     | varchar(111) | NO   |     | NULL              |                |
| date            | timestamp    | NO   |     | CURRENT_TIMESTAMP |                |
+-----------------+--------------+------+-----+-------------------+----------------+
14 rows in set (0.01 sec)

+--------------+-----------------------------------------------------------------
| Table        | Create Table

+--------------+-----------------------------------------------------------------
| useraccounts | CREATE TABLE `useraccounts` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `email` text NOT NULL,
  `password` text NOT NULL,
  `name` varchar(111) NOT NULL,
  `balance` varchar(111) NOT NULL,
  `aadhaar_number` varchar(111) NOT NULL,
  `phone` varchar(111) NOT NULL,
  `city` varchar(111) NOT NULL,
  `address` varchar(111) NOT NULL,
  `source_of_income` varchar(111) NOT NULL,
  `accountNo` varchar(111) NOT NULL,
  `branch` int(11) NOT NULL,
  `accountType` varchar(111) NOT NULL,
  `date` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=latin1 |
+--------------+-----------------------------------------------------------------
1 row in set (0.00 sec)
```

Figure 4.10: UserAccounts Table



```
mysql> DESC transaction; show create table transaction;
+----------------+--------------+------+-----+-------------------+----------------+
| Field          | Type         | Null | Key | Default           | Extra          |
+----------------+--------------+------+-----+-------------------+----------------+
| transactionId  | int(11)      | NO   | PRI | NULL              | auto_increment |
| action         | varchar(111) | NO   |     | NULL              |                |
| credit         | varchar(111) | YES  |     | NULL              |                |
| debit          | varchar(111) | YES  |     | NULL              |                |
| beneficiaryAcc | varchar(111) | NO   |     | NULL              |                |
| userId         | int(11)      | NO   |     | NULL              |                |
| date           | timestamp    | NO   |     | CURRENT_TIMESTAMP |                |
+----------------+--------------+------+-----+-------------------+----------------+
7 rows in set (0.01 sec)

+-------------+---------------------------------------------------------------------
| Table       | Create Table
                                                                                   |
+-------------+---------------------------------------------------------------------
| transaction | CREATE TABLE `transaction` (
  `transactionId` int(11) NOT NULL AUTO_INCREMENT,
  `action` varchar(111) NOT NULL,
  `credit` varchar(111) DEFAULT NULL,
  `debit` varchar(111) DEFAULT NULL,
  `beneficiaryAcc` varchar(111) NOT NULL,
  `userId` int(11) NOT NULL,
  `date` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`transactionId`)
) ENGINE=InnoDB AUTO_INCREMENT=28 DEFAULT CHARSET=latin1 |
+-------------+---------------------------------------------------------------------
1 row in set (0.00 sec)
```

Figure 4.11: Transaction Table



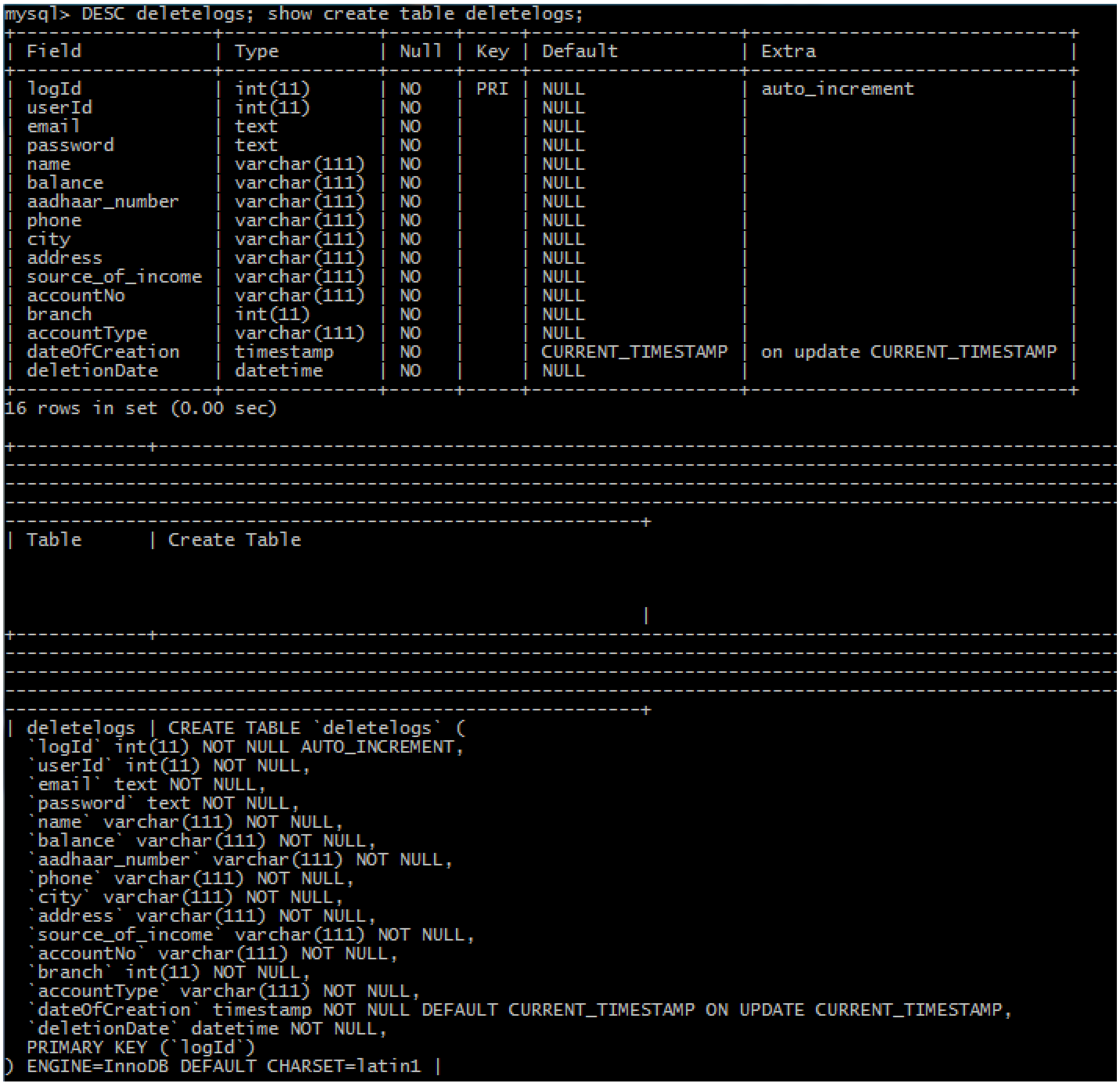
```
mysql> DESC notice; show create table notice;
+--------+-----------+------+-----+-------------------+----------------+
| Field  | Type      | Null | Key | Default           | Extra          |
+--------+-----------+------+-----+-------------------+----------------+
| id     | int(11)   | NO   | PRI | NULL              | auto_increment |
| userId | int(11)   | NO   |     | NULL              |                |
| notice | text      | NO   |     | NULL              |                |
| date   | timestamp | NO   |     | CURRENT_TIMESTAMP |                |
+--------+-----------+------+-----+-------------------+----------------+
4 rows in set (0.00 sec)

+--------+---------------------------------------------------------------
| Table  | Create Table

+--------+---------------------------------------------------------------
| notice | CREATE TABLE `notice` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `userId` int(11) NOT NULL,
  `notice` text NOT NULL,
  `date` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=latin1 |
+--------+---------------------------------------------------------------
1 row in set (0.00 sec)
```

Figure 4.12 Notice Table

4.2 Triggers

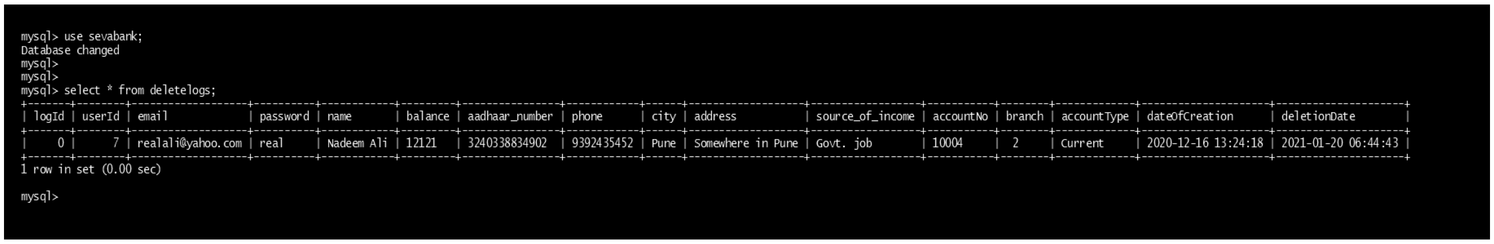
Triggers are stored programs, which are automatically executed or fired when some events occur. So here one extra table is created called “deletelogs” so whenever a record is deleted from useraccounts table then the ‘deletelogger’ trigger will be fired which will store the entire record that is deleted along with timestamp of deletion. Each record inserted into this table has an auto incrementing unique logId. This is mostly useful for auditing purposes.

```
CREATE TABLE `sevabank`.`deletelogs`
( `logId` INT NOT NULL AUTO_INCREMENT ,
  `userId` int(11) NOT NULL,
  `email` text NOT NULL,
  `password` text NOT NULL,
  `name` varchar(111) NOT NULL,
  `balance` varchar(111) NOT NULL,
  `aadhaar_number` varchar(111) NOT NULL,
  `phone` varchar(111) NOT NULL,
  `city` varchar(111) NOT NULL,
  `address` varchar(111) NOT NULL,
  `source_of_income` varchar(111) NOT NULL,
  `accountNo` varchar(111) NOT NULL,
  `branch` int(11) NOT NULL,
  `accountType` varchar(111) NOT NULL,
  `dateOfCreation` timestamp NOT NULL,
  `deletionDate` DATETIME NOT NULL ,
PRIMARY KEY (`id`)) ENGINE = InnoDB;
```



```
mysql> DESC deletelogs; show create table deletelogs;
+------------------+--------------+------+-----+-------------------+-----------------------------+
| Field            | Type         | Null | Key | Default           | Extra                       |
+------------------+--------------+------+-----+-------------------+-----------------------------+
| logId            | int(11)      | NO   | PRI | NULL              | auto_increment              |
| userId           | int(11)      | NO   |     | NULL              |                             |
| email            | text         | NO   |     | NULL              |                             |
| password         | text         | NO   |     | NULL              |                             |
| name             | varchar(111) | NO   |     | NULL              |                             |
| balance          | varchar(111) | NO   |     | NULL              |                             |
| aadhaar_number   | varchar(111) | NO   |     | NULL              |                             |
| phone            | varchar(111) | NO   |     | NULL              |                             |
| city             | varchar(111) | NO   |     | NULL              |                             |
| address          | varchar(111) | NO   |     | NULL              |                             |
| source_of_income | varchar(111) | NO   |     | NULL              |                             |
| accountNo        | varchar(111) | NO   |     | NULL              |                             |
| branch           | int(11)      | NO   |     | NULL              |                             |
| accountType      | varchar(111) | NO   |     | NULL              |                             |
| dateOfCreation   | timestamp    | NO   |     | CURRENT_TIMESTAMP | on update CURRENT_TIMESTAMP |
| deletionDate     | datetime     | NO   |     | NULL              |                             |
+------------------+--------------+------+-----+-------------------+-----------------------------+
16 rows in set (0.00 sec)

+------------+----------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
----------------------------------------------------------+
| Table      | Create Table


                                                          |
+------------+----------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
----------------------------------------------------------+
| deletelogs | CREATE TABLE `deletelogs` (
  `logId` int(11) NOT NULL AUTO_INCREMENT,
  `userId` int(11) NOT NULL,
  `email` text NOT NULL,
  `password` text NOT NULL,
  `name` varchar(111) NOT NULL,
  `balance` varchar(111) NOT NULL,
  `aadhaar_number` varchar(111) NOT NULL,
  `phone` varchar(111) NOT NULL,
  `city` varchar(111) NOT NULL,
  `address` varchar(111) NOT NULL,
  `source_of_income` varchar(111) NOT NULL,
  `accountNo` varchar(111) NOT NULL,
  `branch` int(11) NOT NULL,
  `accountType` varchar(111) NOT NULL,
  `dateOfCreation` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  `deletionDate` datetime NOT NULL,
  PRIMARY KEY (`logId`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 |
```

Figure 4.13 Deletelogs Table



```
mysql> use sevabank;
Database changed
mysql>
mysql>
mysql> select * from deletelogs;
+-------+--------+-------------------+----------+------------+---------+----------------+------------+------+-------------------+------------------+-----------+--------+-------------+---------------------+---------------------+
| logId | userId | email             | password | name       | balance | aadhaar_number | phone      | city | address           | source_of_income | accountNo | branch | accountType | dateOfCreation      | deletionDate        |
+-------+--------+-------------------+----------+------------+---------+----------------+------------+------+-------------------+------------------+-----------+--------+-------------+---------------------+---------------------+
|     0 |      7 | realali@yahoo.com | real     | Nadeem Ali | 12121   | 3240338834902  | 9392435452 | Pune | Somewhere in Pune | Govt. job        | 10004     |  2     | Current     | 2020-12-16 13:24:18 | 2021-01-20 06:44:43 |
+-------+--------+-------------------+----------+------------+---------+----------------+------------+------+-------------------+------------------+-----------+--------+-------------+---------------------+---------------------+
1 row in set (0.00 sec)

mysql>
```

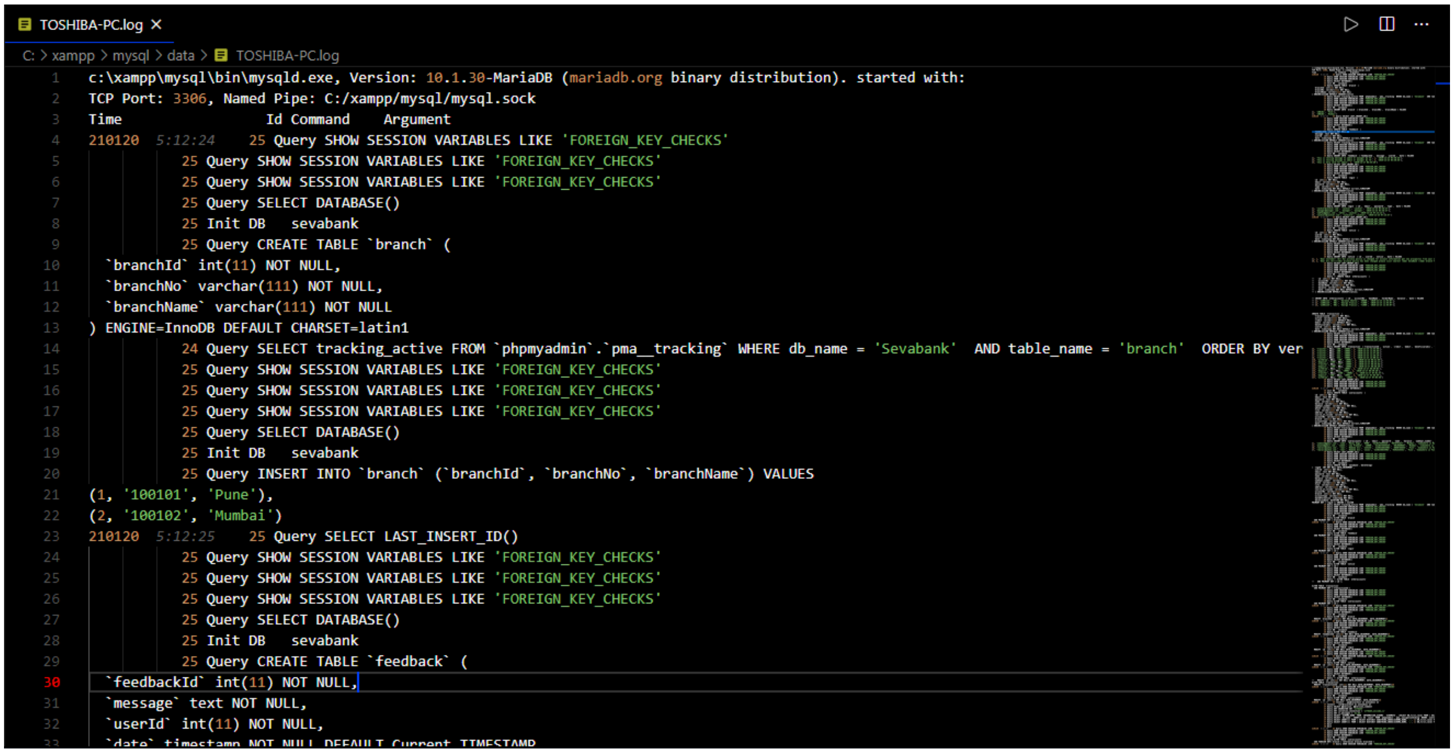
Figure 4.14: Deletelogs table after deleting an user account.

Creation of Trigger

```
CREATE TRIGGER `deletelogger`
AFTER delete ON `useraccounts`
FOR EACH ROW
INSERT INTO deletelogs
VALUES
(DEFAULT,
OLD.id,
OLD.email,
OLD.password,
OLD.name,
OLD.balance,
OLD.aadhaar_number,
OLD.phone,
OLD.city,
OLD.address,
OLD.source_of_income,
OLD.accountNo,
OLD.branch,
OLD.accountType,
OLD.date,
NOW());
```

4.3 Usage of Database

The following queries were not provided directly through the database, but through the PHP backend instead. A log file was opened and maintained to document all queries that may occur during everyday operation of the database.

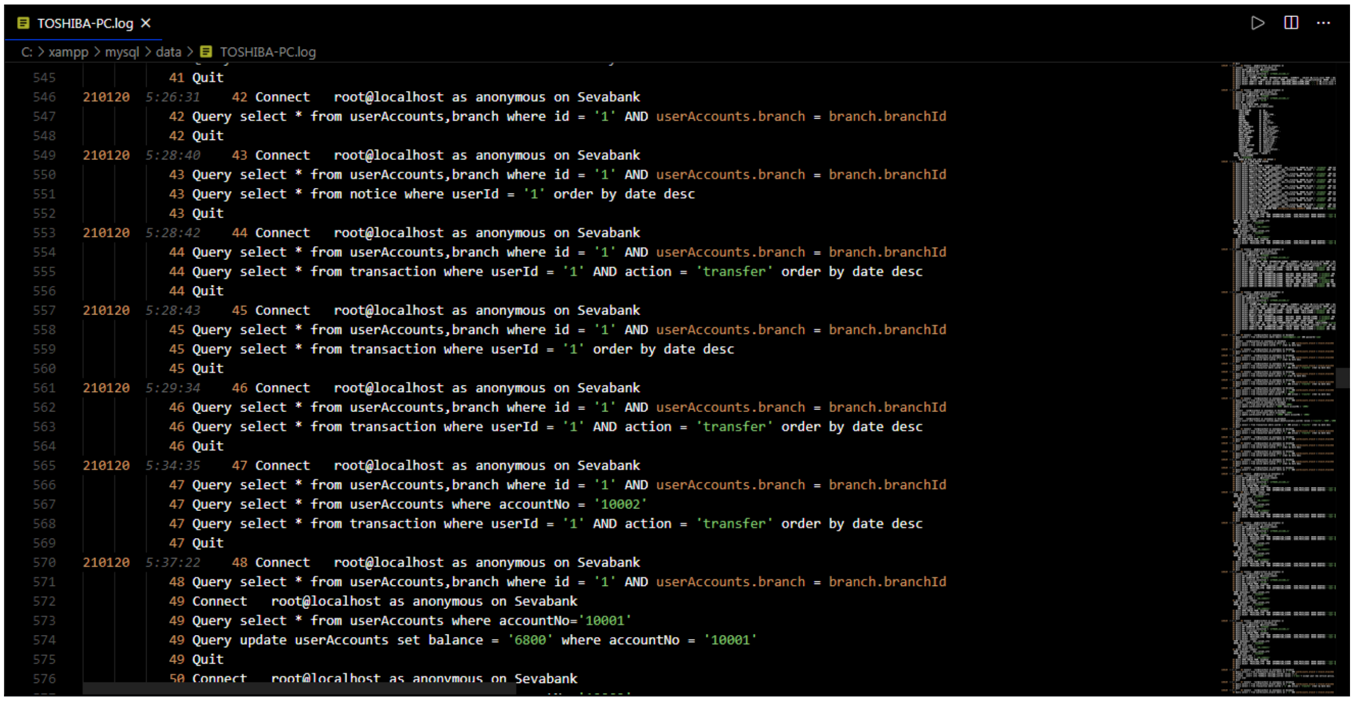


TOSHIBA-PC.log ×

C: > xampp > mysql > data > TOSHIBA-PC.log

```
c:\xampp\mysql\bin\mysqld.exe, Version: 10.1.30-MariaDB (mariadb.org binary distribution). started with:
TCP Port: 3306, Named Pipe: C:/xampp/mysql/mysql.sock
Time                 Id Command    Argument
210120  5:12:24     25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SELECT DATABASE()
                    25 Init DB   sevabank
                    25 Query CREATE TABLE `branch` (
  `branchId` int(11) NOT NULL,
  `branchNo` varchar(111) NOT NULL,
  `branchName` varchar(111) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1
                    24 Query SELECT tracking_active FROM `phpmyadmin`.`pma__tracking` WHERE db_name = 'Sevabank'  AND table_name = 'branch'  ORDER BY ver
                    25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SELECT DATABASE()
                    25 Init DB   sevabank
                    25 Query INSERT INTO `branch` (`branchId`, `branchNo`, `branchName`) VALUES
(1, '100101', 'Pune'),
(2, '100102', 'Mumbai')
210120  5:12:25     25 Query SELECT LAST_INSERT_ID()
                    25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SHOW SESSION VARIABLES LIKE 'FOREIGN_KEY_CHECKS'
                    25 Query SELECT DATABASE()
                    25 Init DB   sevabank
                    25 Query CREATE TABLE `feedback` (
  `feedbackId` int(11) NOT NULL,
  `message` text NOT NULL,
  `userId` int(11) NOT NULL,
  `date` timestamp NOT NULL DEFAULT Current_TIMESTAMP
```

Figure 4.15: Normal execution of database.



TOSHIBA-PC.log ×

C: > xampp > mysql > data > TOSHIBA-PC.log

```
                    41 Quit
210120  5:26:31     42 Connect   root@localhost as anonymous on Sevabank
                    42 Query select * from userAccounts,branch where id = '1' AND userAccounts.branch = branch.branchId
                    42 Quit
210120  5:28:40     43 Connect   root@localhost as anonymous on Sevabank
                    43 Query select * from userAccounts,branch where id = '1' AND userAccounts.branch = branch.branchId
                    43 Query select * from notice where userId = '1' order by date desc
                    43 Quit
210120  5:28:42     44 Connect   root@localhost as anonymous on Sevabank
                    44 Query select * from userAccounts,branch where id = '1' AND userAccounts.branch = branch.branchId
                    44 Query select * from transaction where userId = '1' AND action = 'transfer' order by date desc
                    44 Quit
210120  5:28:43     45 Connect   root@localhost as anonymous on Sevabank
                    45 Query select * from userAccounts,branch where id = '1' AND userAccounts.branch = branch.branchId
                    45 Query select * from transaction where userId = '1' order by date desc
                    45 Quit
210120  5:29:34     46 Connect   root@localhost as anonymous on Sevabank
                    46 Query select * from userAccounts,branch where id = '1' AND userAccounts.branch = branch.branchId
                    46 Query select * from transaction where userId = '1' AND action = 'transfer' order by date desc
                    46 Quit
210120  5:34:35     47 Connect   root@localhost as anonymous on Sevabank
                    47 Query select * from userAccounts,branch where id = '1' AND userAccounts.branch = branch.branchId
                    47 Query select * from userAccounts where accountNo = '10002'
                    47 Query select * from transaction where userId = '1' AND action = 'transfer' order by date desc
                    47 Quit
210120  5:37:22     48 Connect   root@localhost as anonymous on Sevabank
                    48 Query select * from userAccounts,branch where id = '1' AND userAccounts.branch = branch.branchId
                    49 Connect   root@localhost as anonymous on Sevabank
                    49 Query select * from userAccounts where accountNo='10001'
                    49 Query update userAccounts set balance = '6800' where accountNo = '10001'
                    49 Quit
                    50 Connect   root@localhost as anonymous on Sevabank
```

Figure 4.16: Normal execution of database (continued...)

As of now, the log file consists of more than 1000 lines of log data.

Chapter 5

Results and Discussion

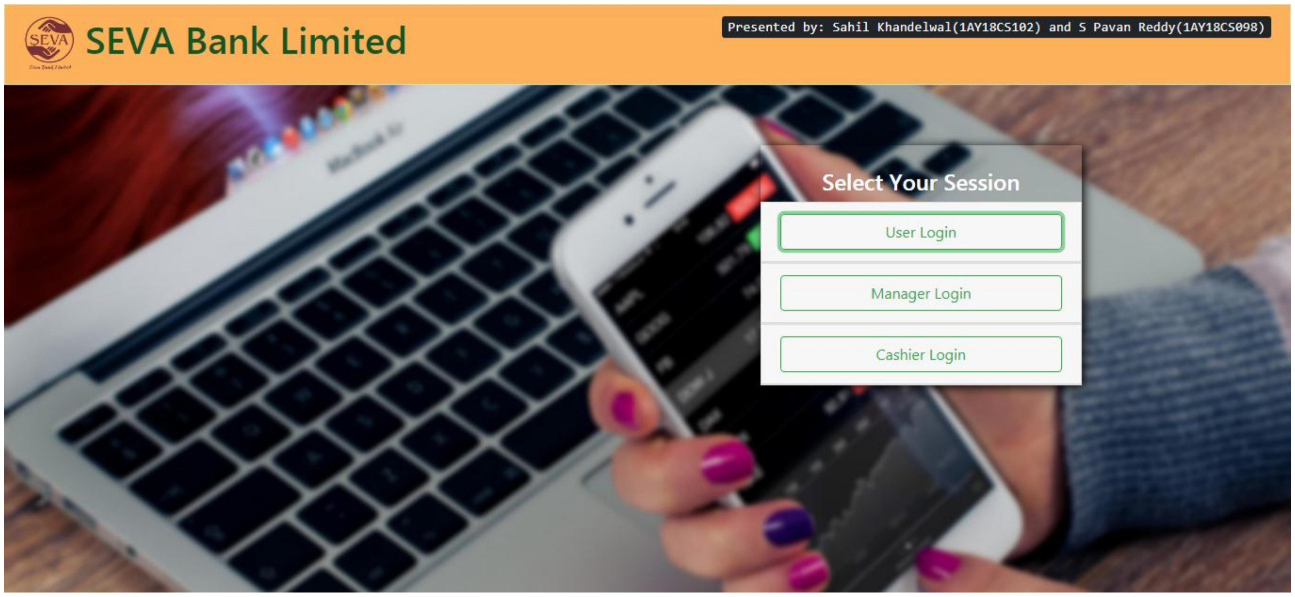

SEVA Bank Limited
Presented by: Sahil Khandelwal(1AY18CS102) and S Pavan Reddy(1AY18CS098)
Select Your Session
User Login
Manager Login
Cashier Login

Figure 5.1: Home/Login Page

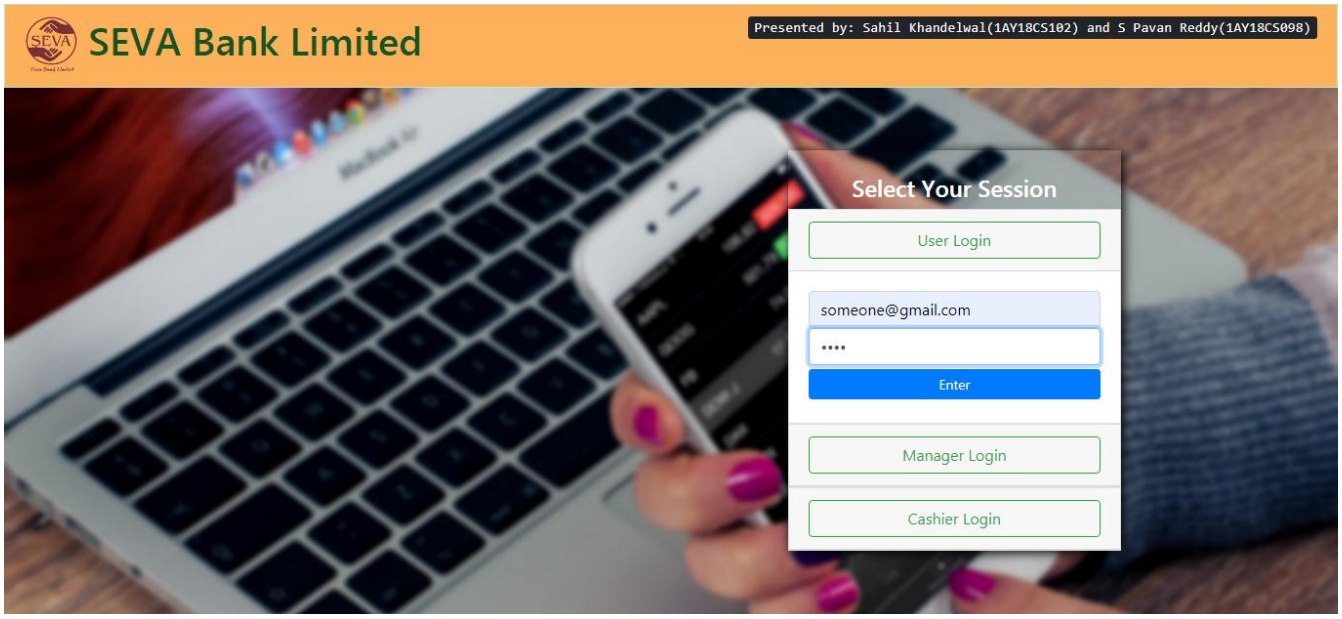
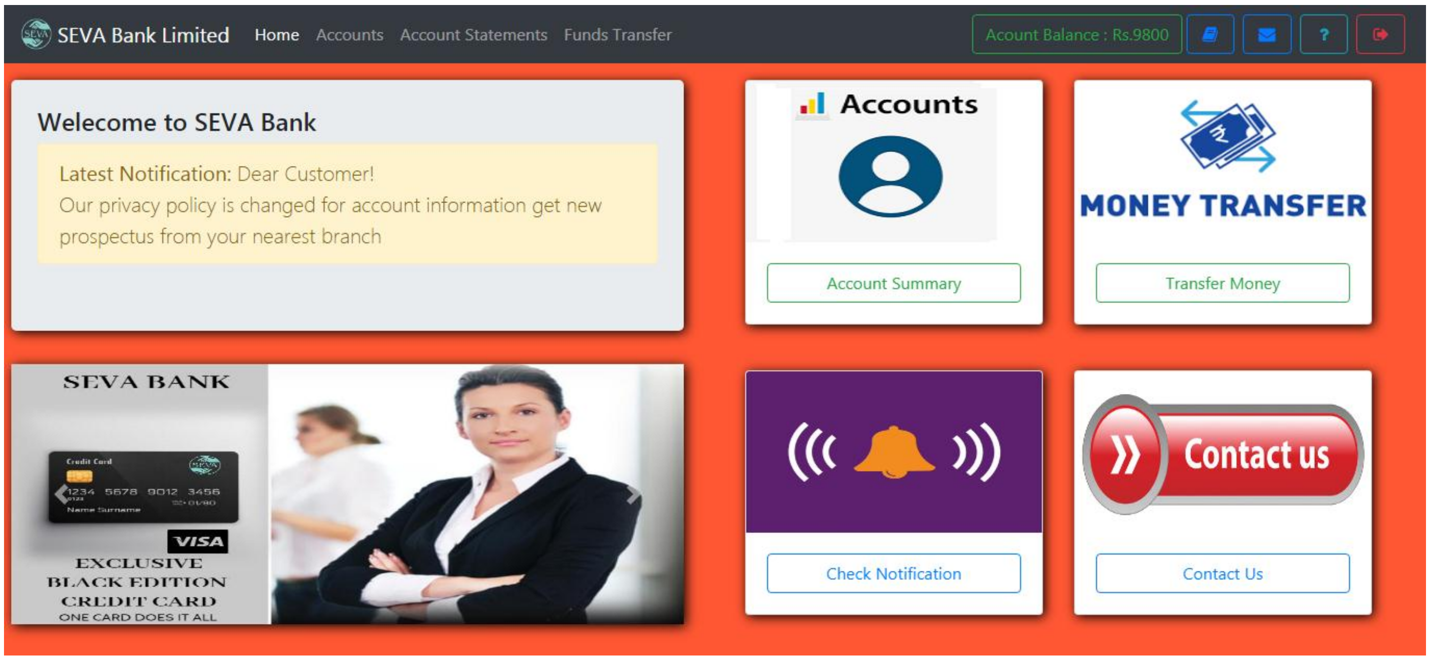


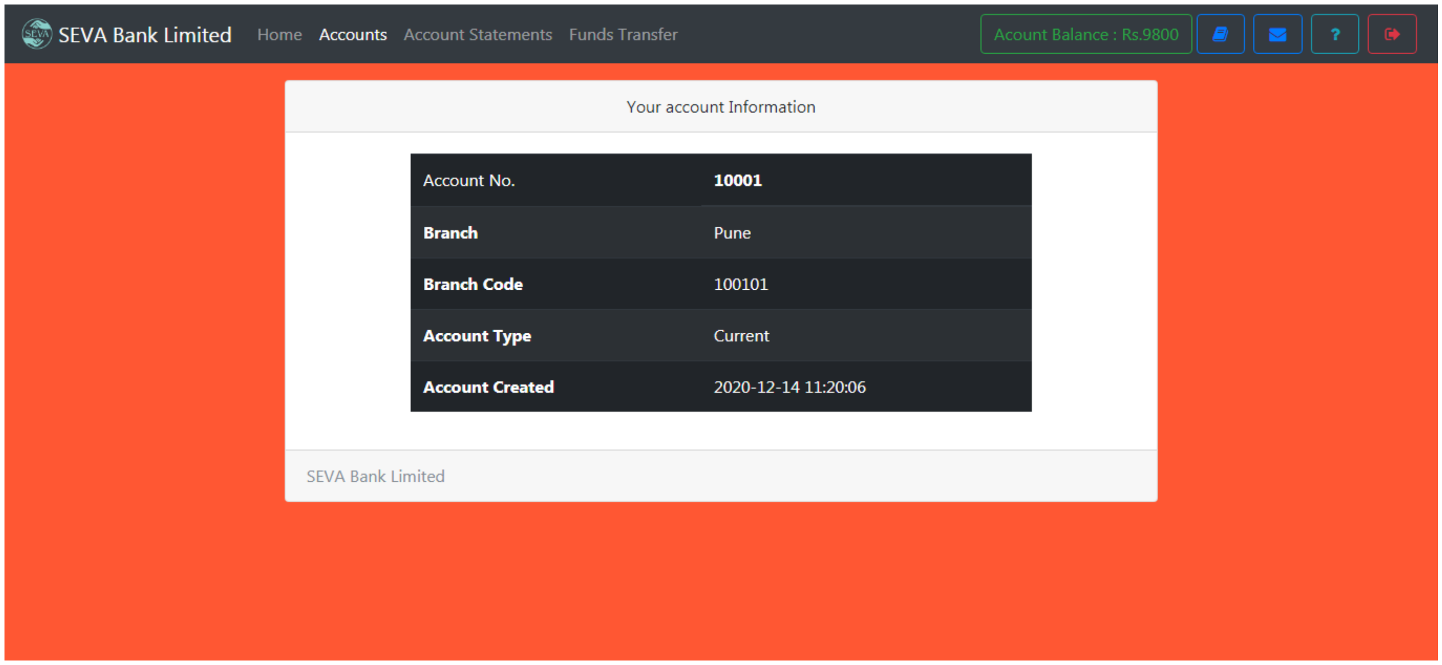
Figure 5.2: User Login



Welcome to SEVA Bank

Latest Notification: Dear Customer!
Our privacy policy is changed for account information get new prospectus from your nearest branch

Figure 5.3: User Home Page

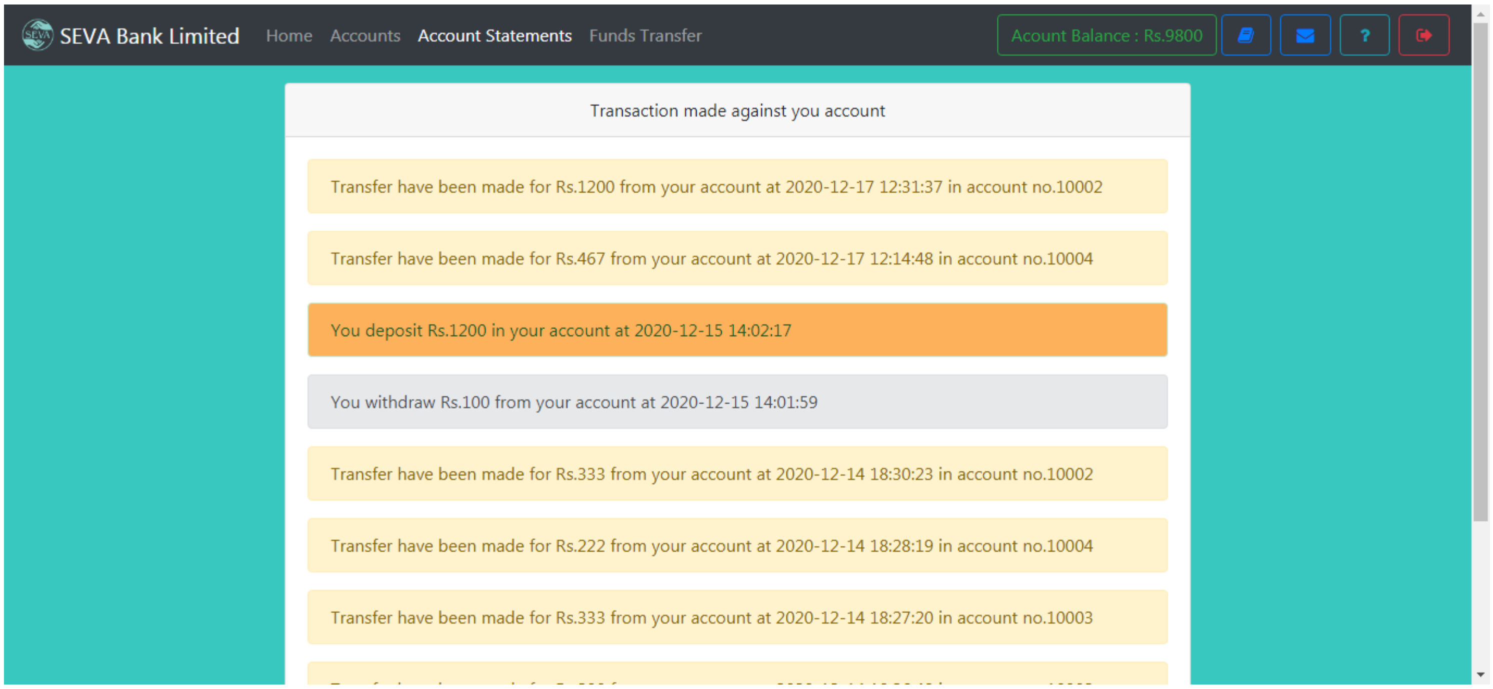


Your account Information

Account No.	**10001**
Branch	Pune
Branch Code	100101
Account Type	Current
Account Created	2020-12-14 11:20:06

SEVA Bank Limited

Figure 5.4: Accounts Page



Transaction made against you account

Transfer have been made for Rs.1200 from your account at 2020-12-17 12:31:37 in account no.10002

Transfer have been made for Rs.467 from your account at 2020-12-17 12:14:48 in account no.10004

You deposit Rs.1200 in your account at 2020-12-15 14:02:17

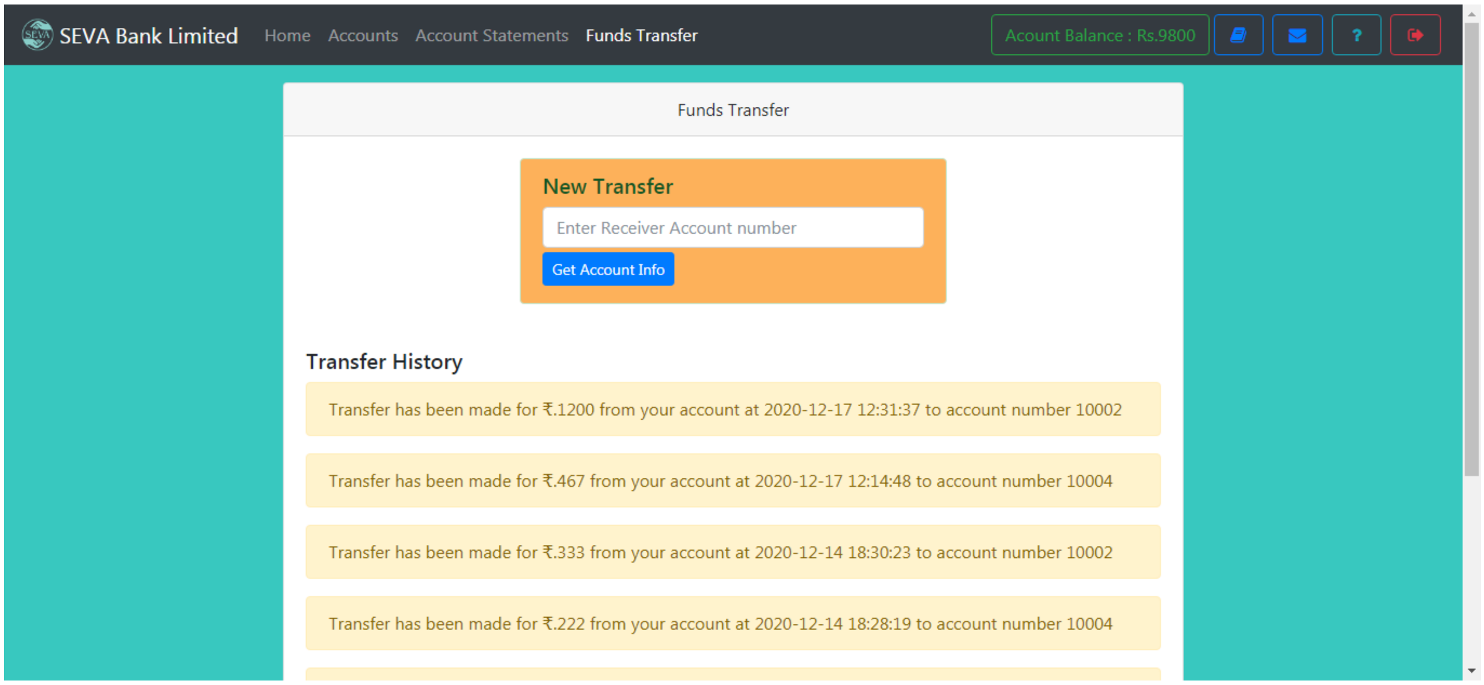
You withdraw Rs.100 from your account at 2020-12-15 14:01:59

Transfer have been made for Rs.333 from your account at 2020-12-14 18:30:23 in account no.10002

Transfer have been made for Rs.222 from your account at 2020-12-14 18:28:19 in account no.10004

Transfer have been made for Rs.333 from your account at 2020-12-14 18:27:20 in account no.10003

Figure 5.5: Statement Page



Transfer History

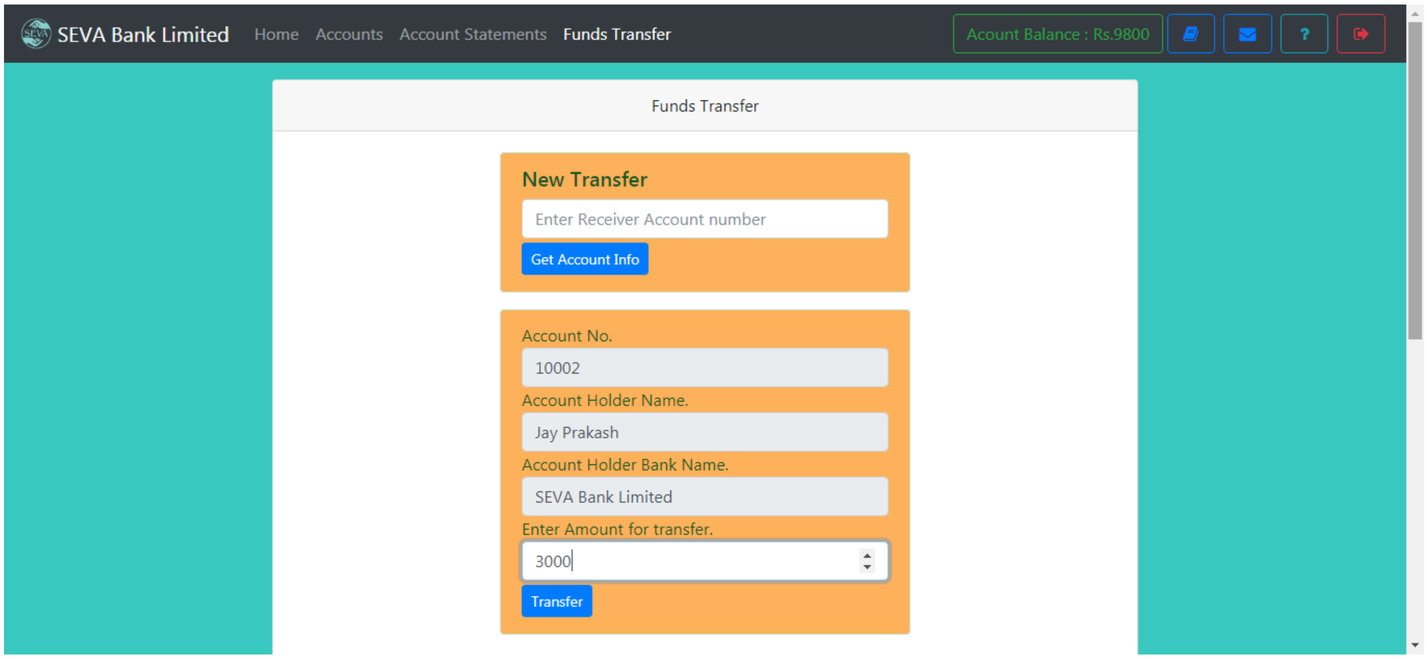
Transfer has been made for ₹.1200 from your account at 2020-12-17 12:31:37 to account number 10002

Transfer has been made for ₹.467 from your account at 2020-12-17 12:14:48 to account number 10004

Transfer has been made for ₹.333 from your account at 2020-12-14 18:30:23 to account number 10002

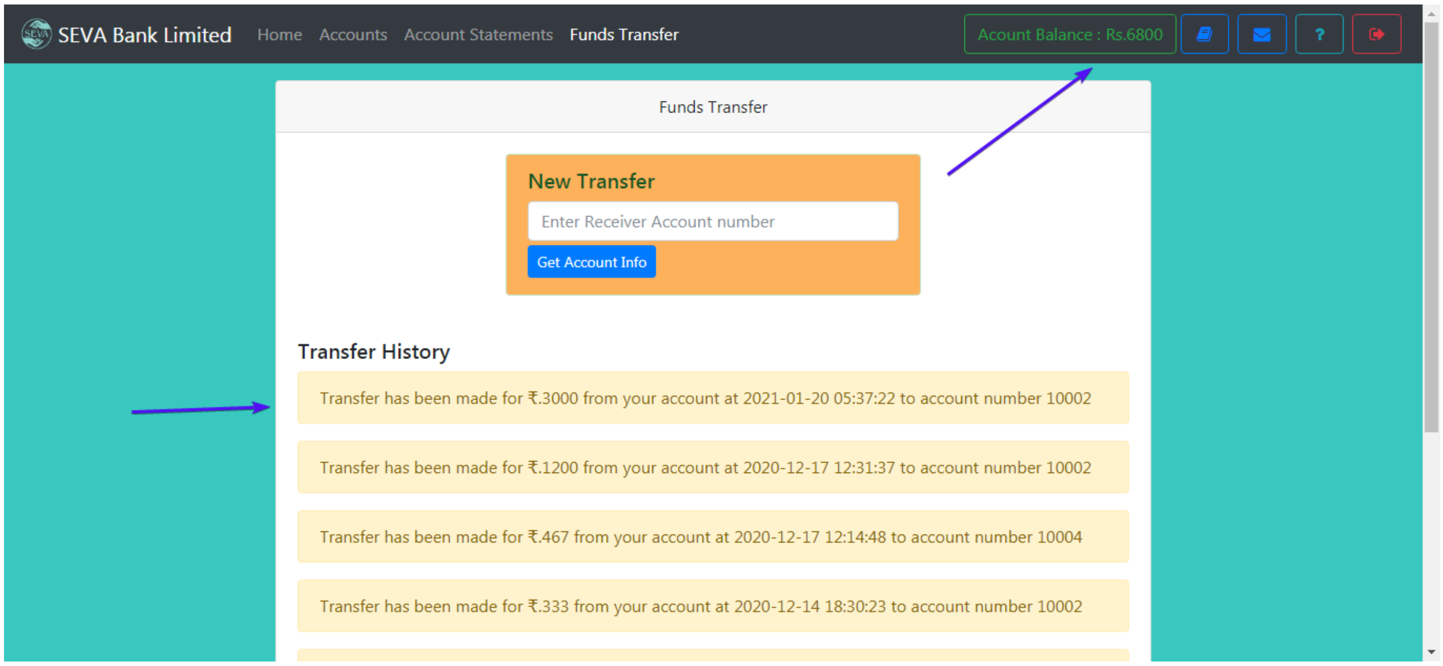
Transfer has been made for ₹.222 from your account at 2020-12-14 18:28:19 to account number 10004

Figure 5.6: Fund Transfer Page



Account No.
10002
Account Holder Name.
Jay Prakash
Account Holder Bank Name.
SEVA Bank Limited
Enter Amount for transfer.
3000

Figure 5.7: Fund Transfer Page (Continued…)



Transfer History

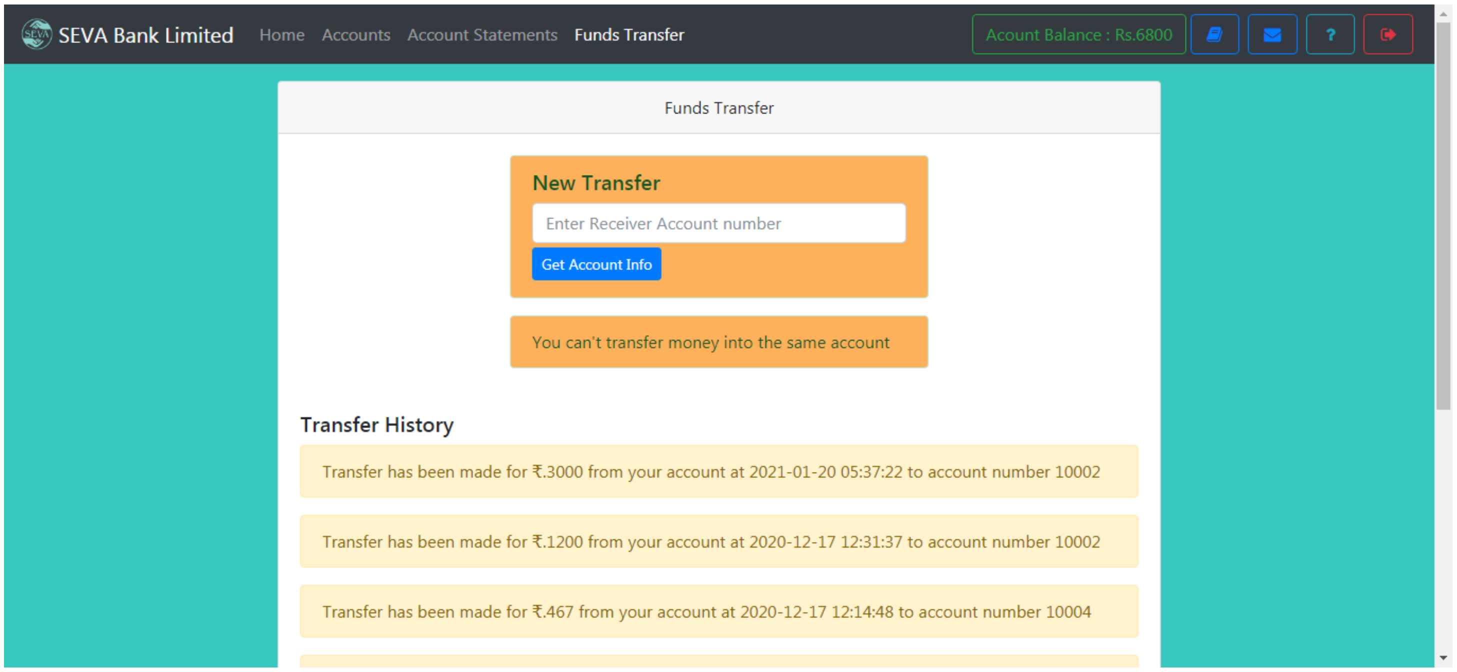
Transfer has been made for ₹.3000 from your account at 2021-01-20 05:37:22 to account number 10002

Transfer has been made for ₹.1200 from your account at 2020-12-17 12:31:37 to account number 10002

Transfer has been made for ₹.467 from your account at 2020-12-17 12:14:48 to account number 10004

Transfer has been made for ₹.333 from your account at 2020-12-14 18:30:23 to account number 10002

Figure 5.8: Fund Transfer Page (After successful fund transfer)



You can't transfer money into the same account

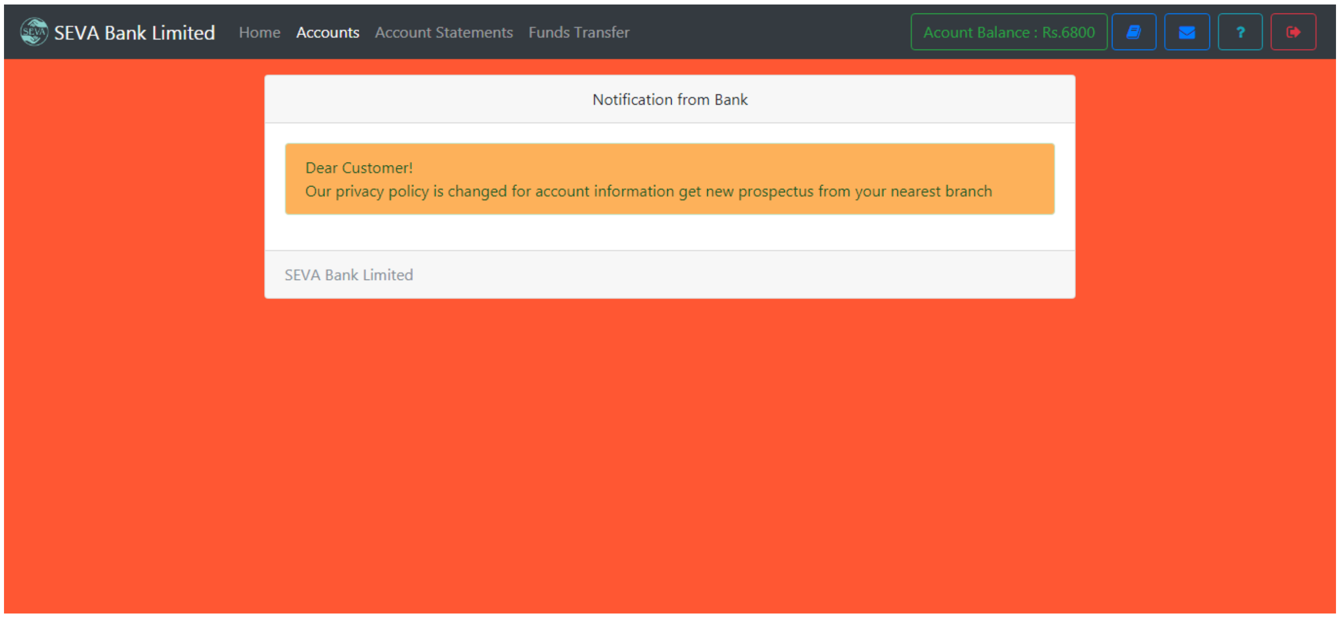
Transfer History

Transfer has been made for ₹.3000 from your account at 2021-01-20 05:37:22 to account number 10002

Transfer has been made for ₹.1200 from your account at 2020-12-17 12:31:37 to account number 10002

Transfer has been made for ₹.467 from your account at 2020-12-17 12:14:48 to account number 10004

Figure 5.9: Page when user enters their own account number as beneficiary account number.



Dear Customer!
Our privacy policy is changed for account information get new prospectus from your nearest branch

SEVA Bank Limited

Figure 5.10: Bank Notice/Notification Page

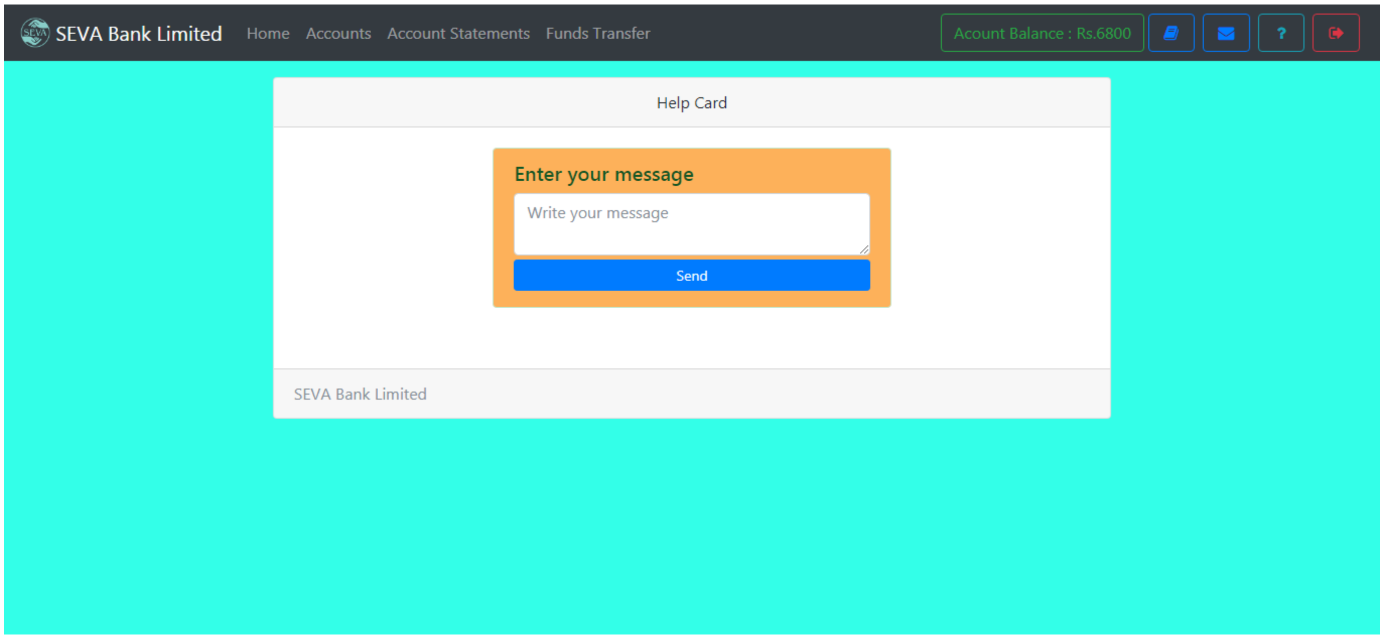
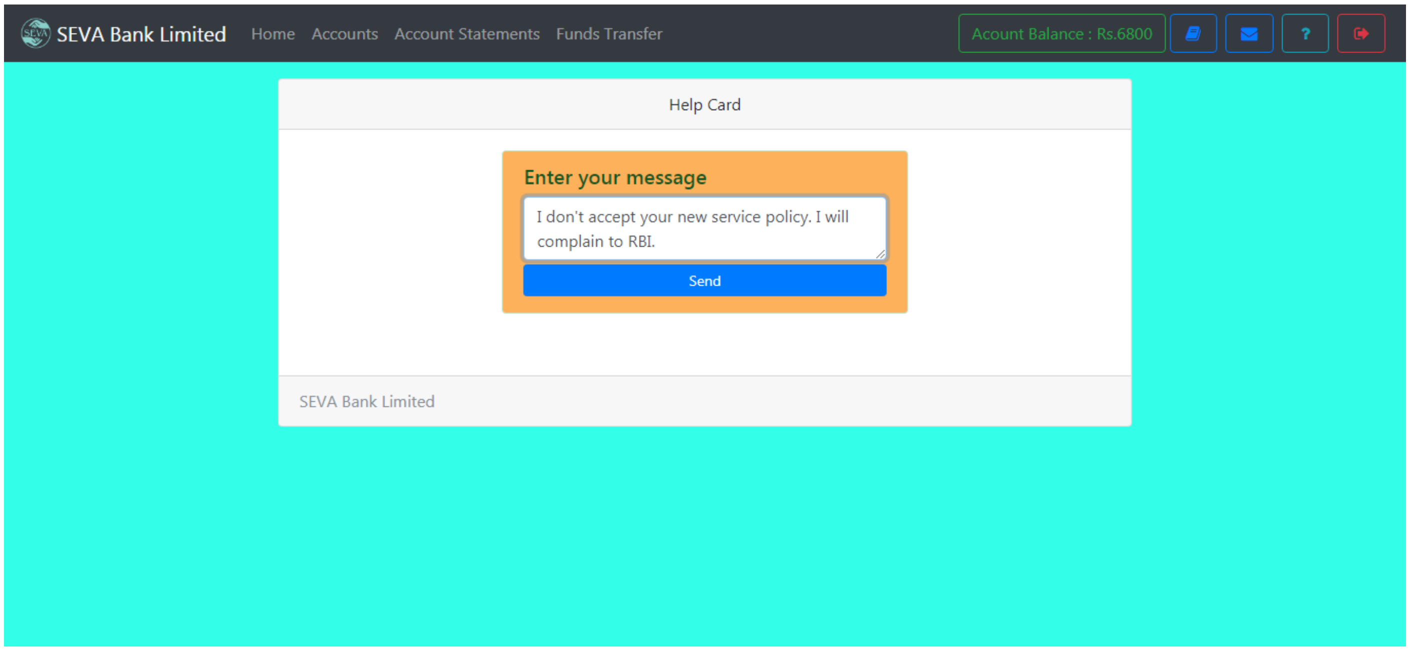
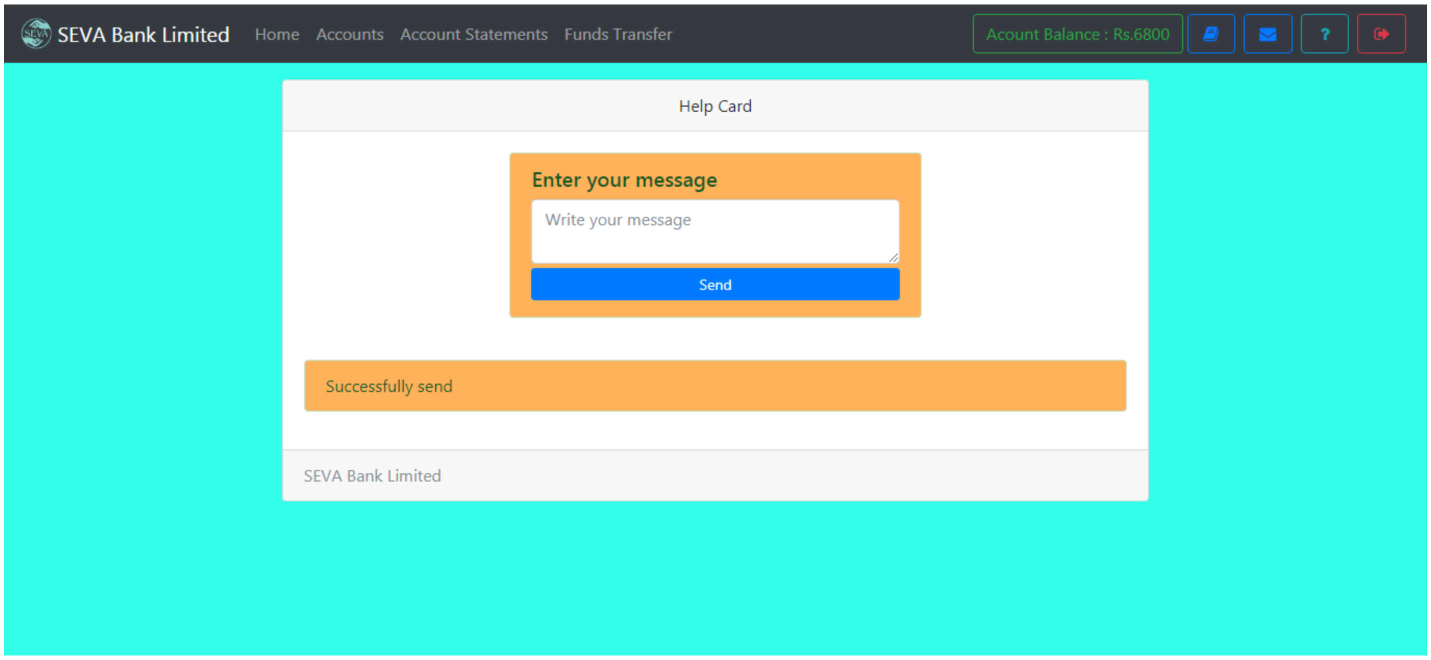


Figure 5.11: Help/Feedback Page



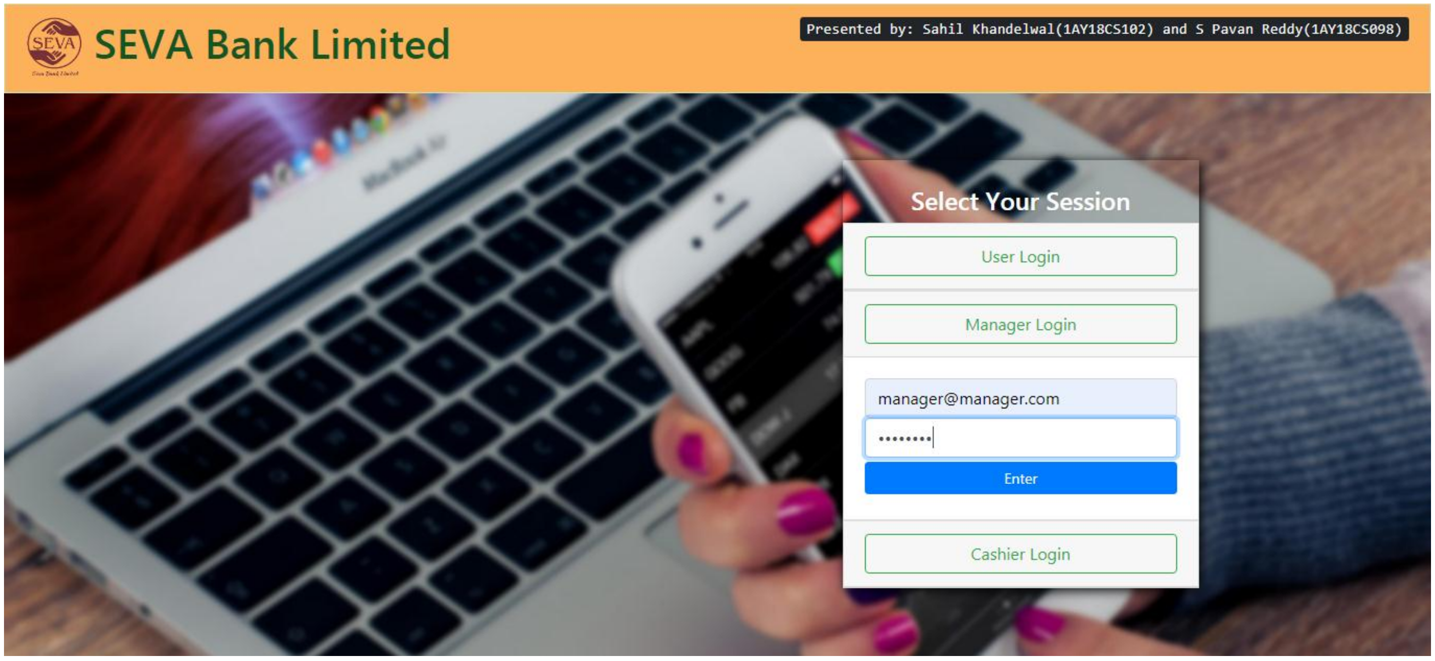
I don't accept your new service policy. I will complain to RBI.

Figure 5.12: User giving feedback



Successfully send

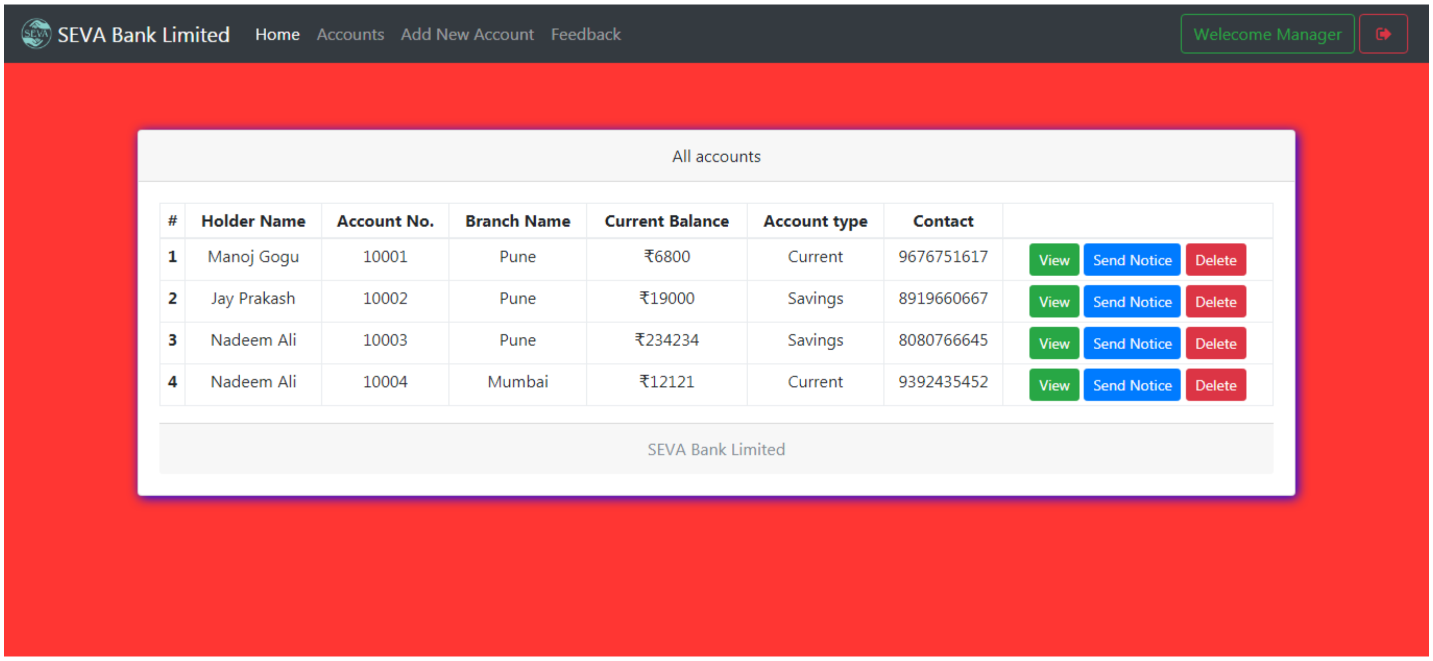
Figure 5.13: Page after user give feedback



Presented by: Sahil Khandelwal(1AY18CS102) and S Pavan Reddy(1AY18CS098)

manager@manager.com

Figure 5.14: Manager Login

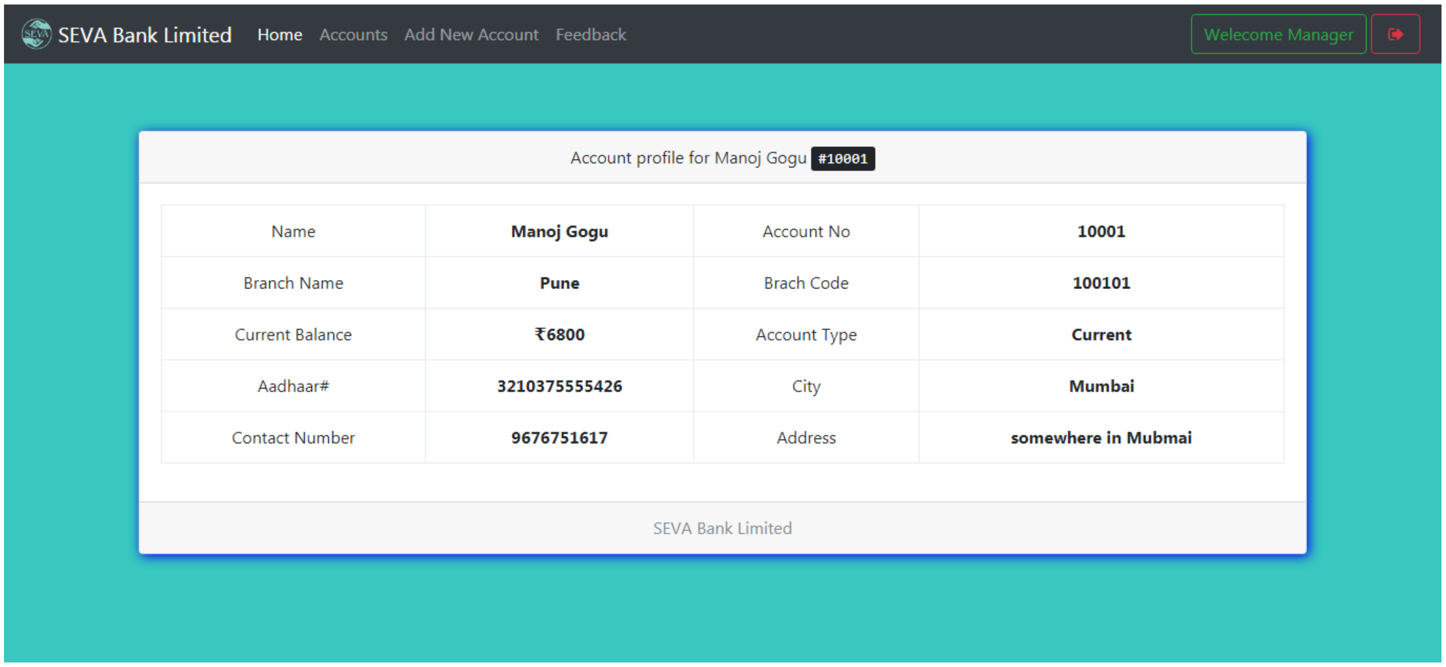


All accounts

#	Holder Name	Account No.	Branch Name	Current Balance	Account type	Contact	
1	Manoj Gogu	10001	Pune	₹6800	Current	9676751617	View Send Notice Delete
2	Jay Prakash	10002	Pune	₹19000	Savings	8919660667	View Send Notice Delete
3	Nadeem Ali	10003	Pune	₹234234	Savings	8080766645	View Send Notice Delete
4	Nadeem Ali	10004	Mumbai	₹12121	Current	9392435452	View Send Notice Delete

SEVA Bank Limited

Figure 5.15: Manager Home Page and User Account Control Page.

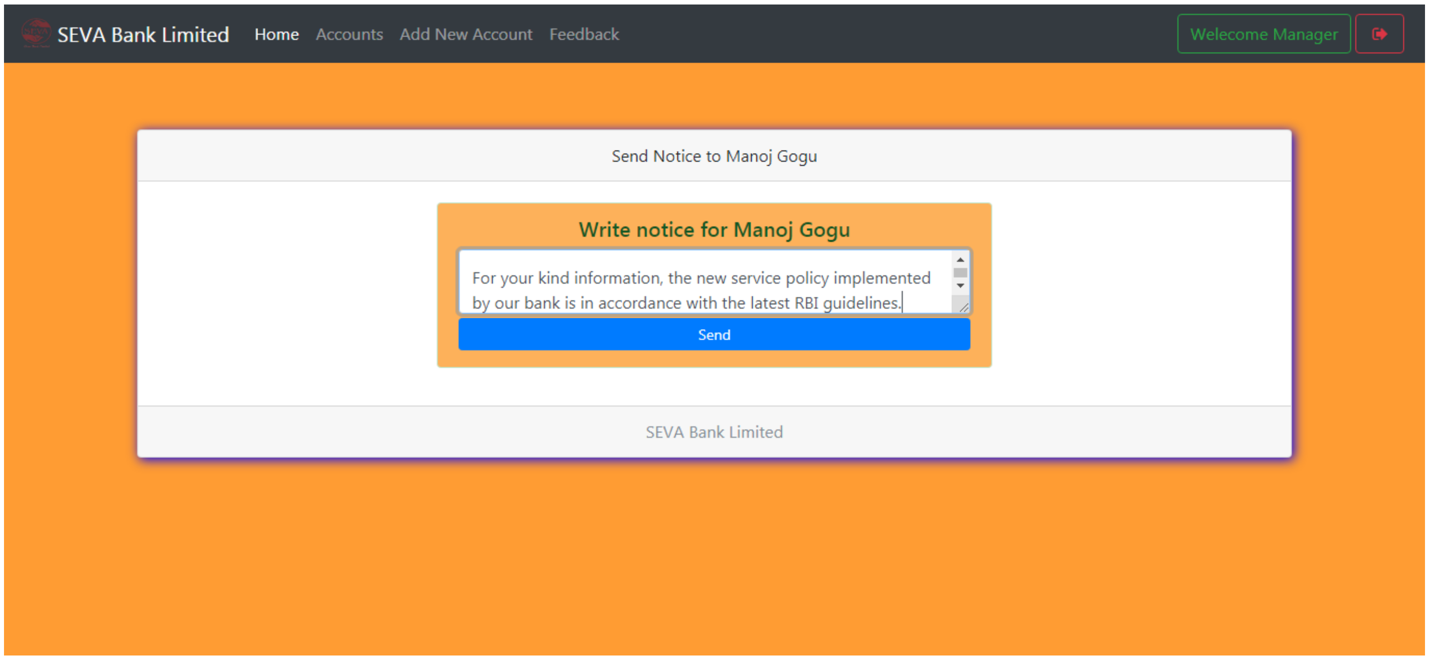


Account profile for Manoj Gogu #10001

Name	**Manoj Gogu**	Account No	**10001**
Branch Name	**Pune**	Brach Code	**100101**
Current Balance	**₹6800**	Account Type	**Current**
Aadhaar#	**3210375555426**	City	**Mumbai**
Contact Number	**9676751617**	Address	**somewhere in Mubmai**

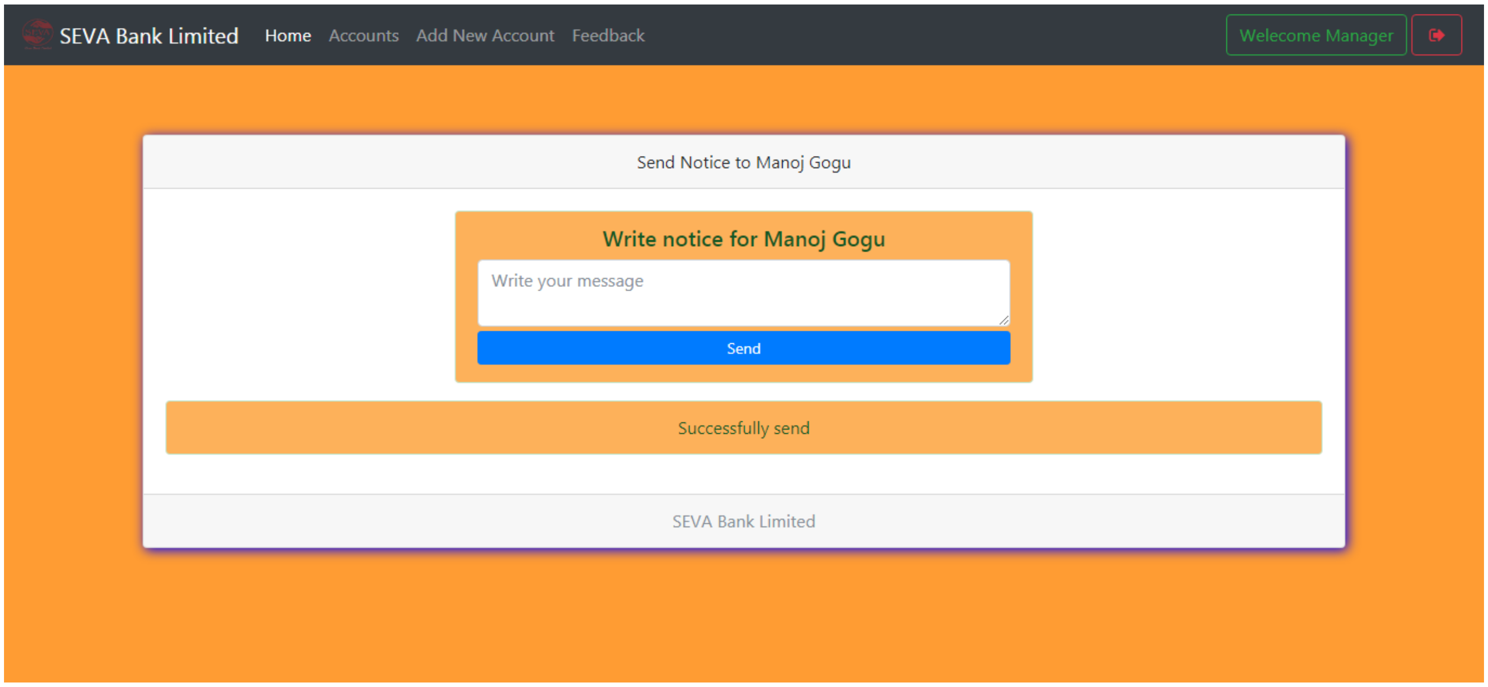
SEVA Bank Limited

Figure 5.16: User account information page for user with account #10001.



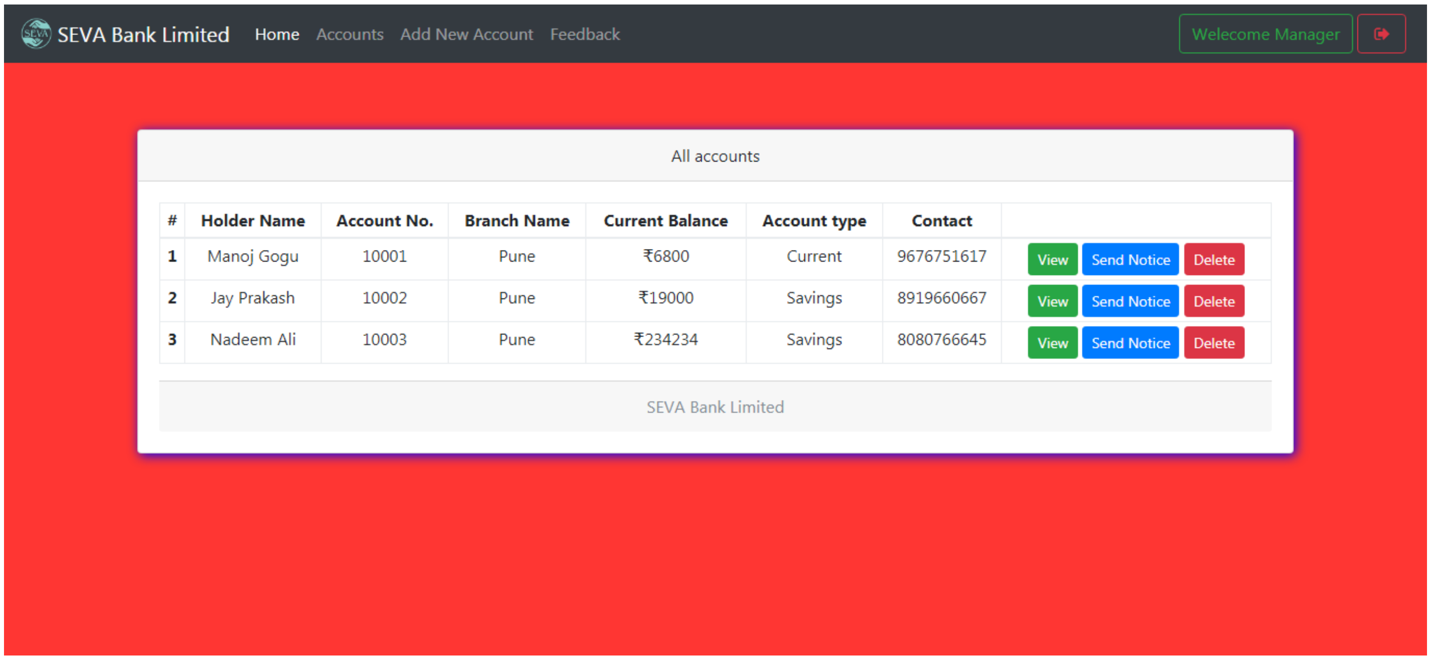
For your kind information, the new service policy implemented by our bank is in accordance with the latest RBI guidelines.

Figure 5.17: Manager sending notice to a user in response to feedback.



Successfully send

Figure 5.18: Page after manager sends notice.

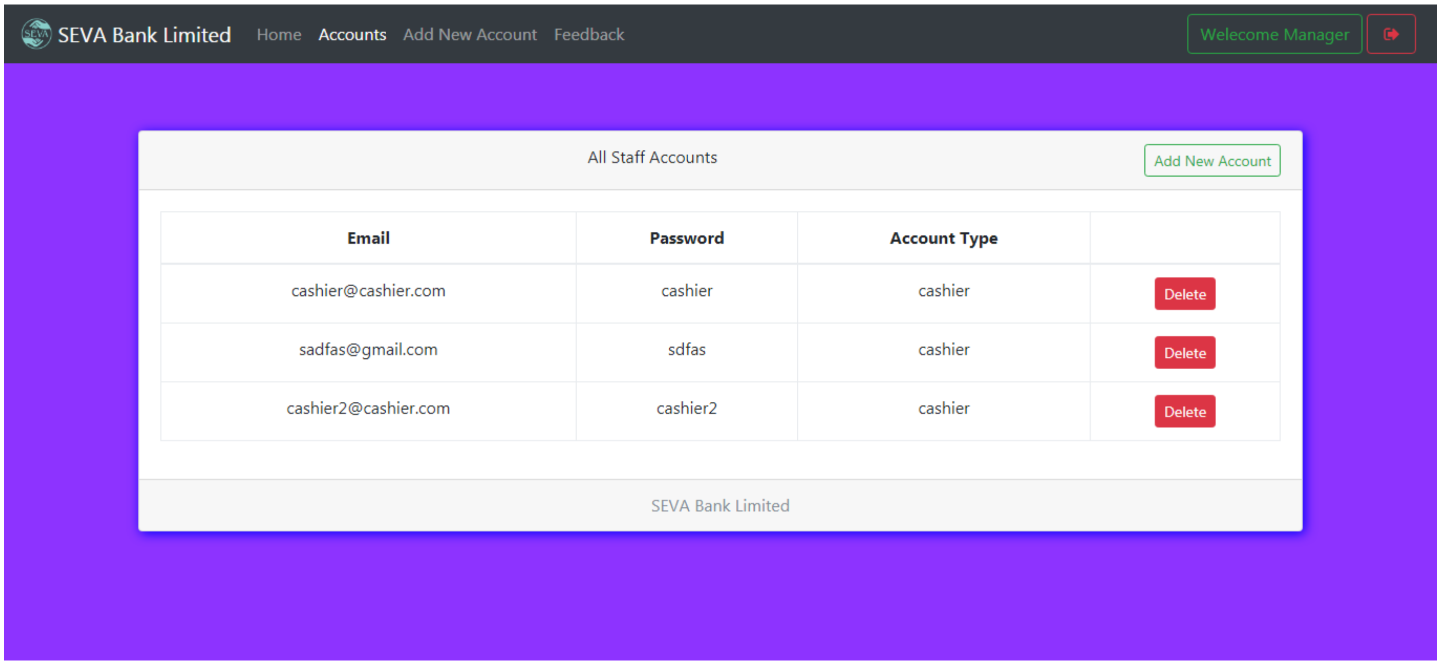


All accounts

#	Holder Name	Account No.	Branch Name	Current Balance	Account type	Contact	
1	Manoj Gogu	10001	Pune	₹6800	Current	9676751617	View Send Notice Delete
2	Jay Prakash	10002	Pune	₹19000	Savings	8919660667	View Send Notice Delete
3	Nadeem Ali	10003	Pune	₹234234	Savings	8080766645	View Send Notice Delete

SEVA Bank Limited

Figure 5.19: Page after Manager deletes an account.

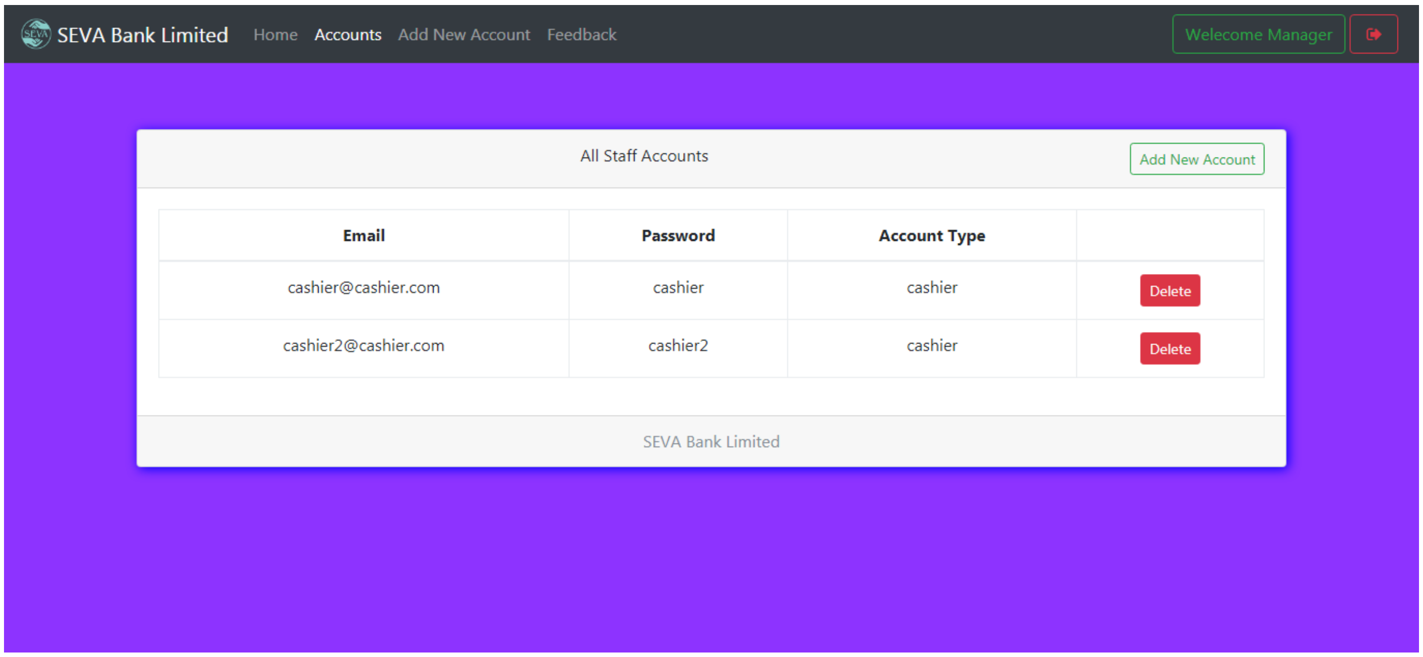


All Staff Accounts

Email	Password	Account Type	
cashier@cashier.com	cashier	cashier	Delete
sadfas@gmail.com	sdfas	cashier	Delete
cashier2@cashier.com	cashier2	cashier	Delete

SEVA Bank Limited

Figure 5.20: Page for Manager to add or remove cashiers.



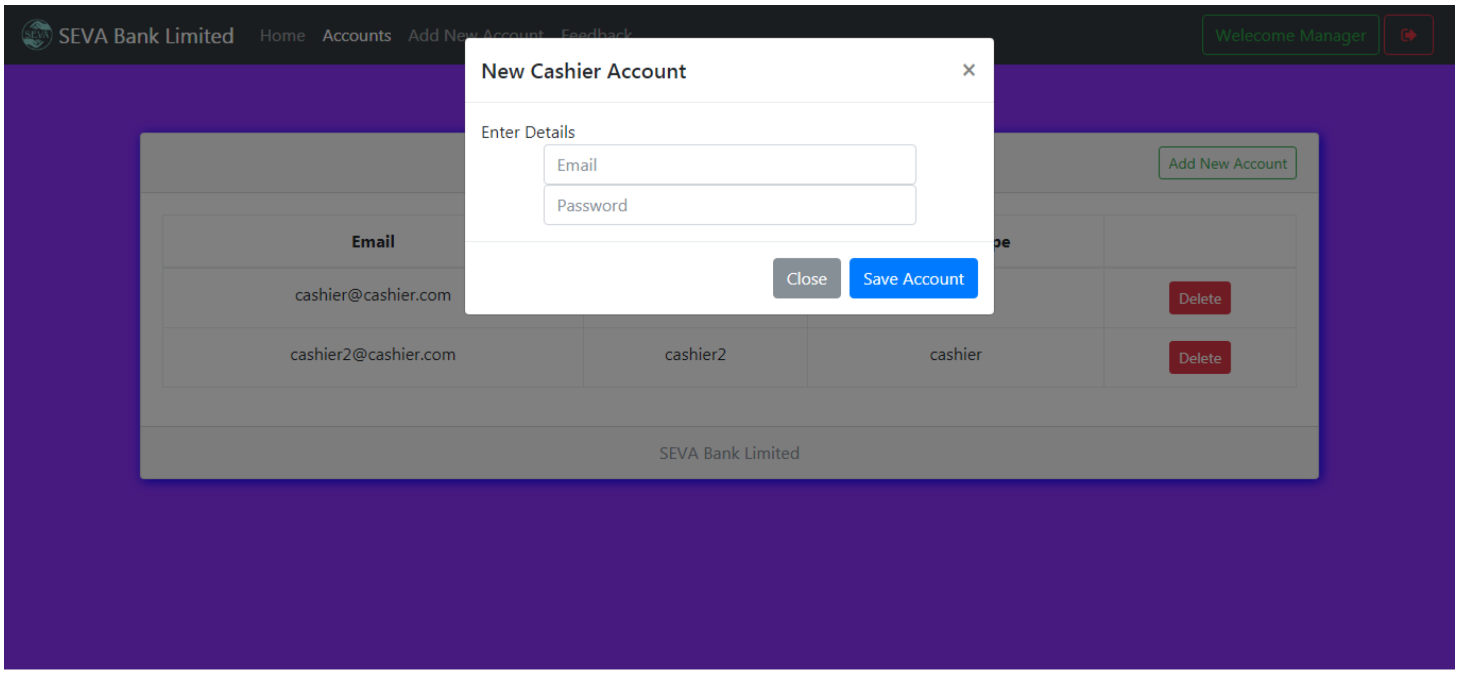
All Staff Accounts

Add New Account

Email	Password	Account Type	
cashier@cashier.com	cashier	cashier	Delete
cashier2@cashier.com	cashier2	cashier	Delete

SEVA Bank Limited

Figure 5.21: Page after manager deletes a cashier.



New Cashier Account ×

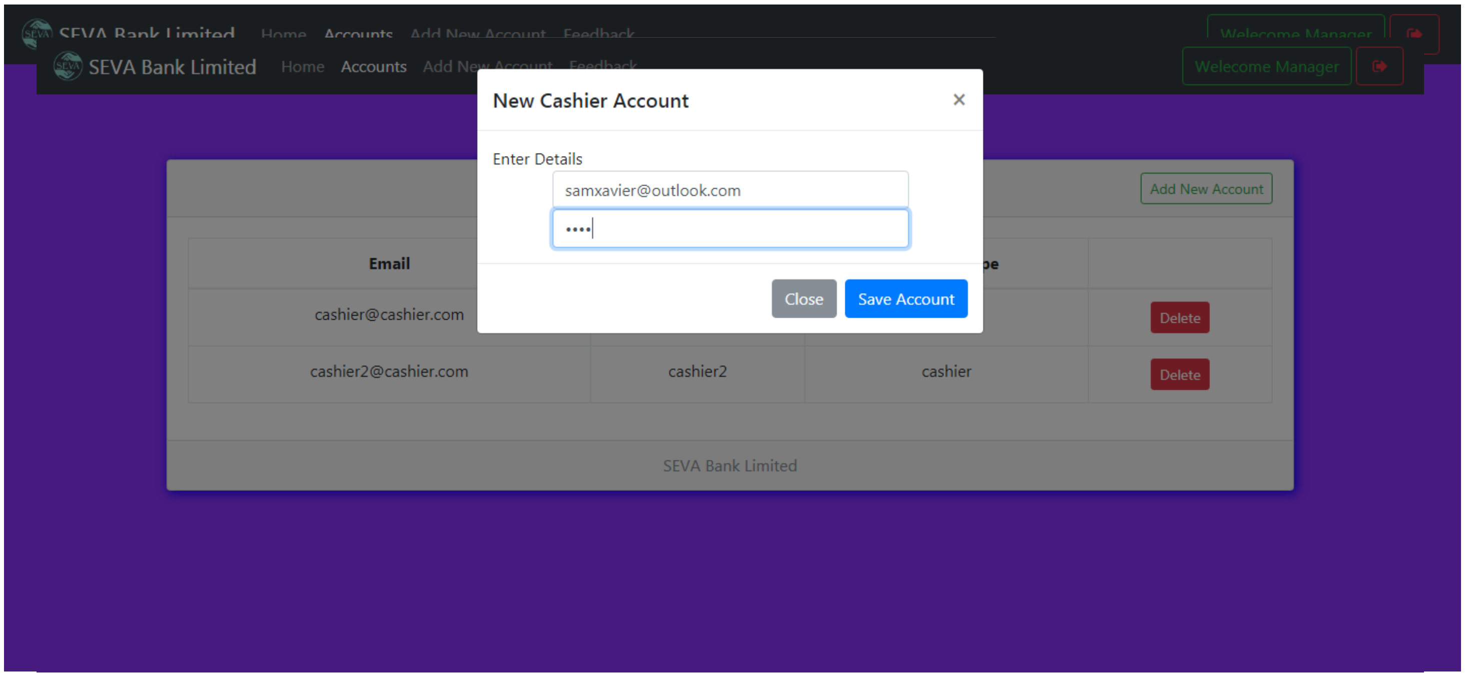
Enter Details

Email

Password

Close Save Account

Figure 5.21: Pop-up box for adding new cashier.



New Cashier Account

Enter Details

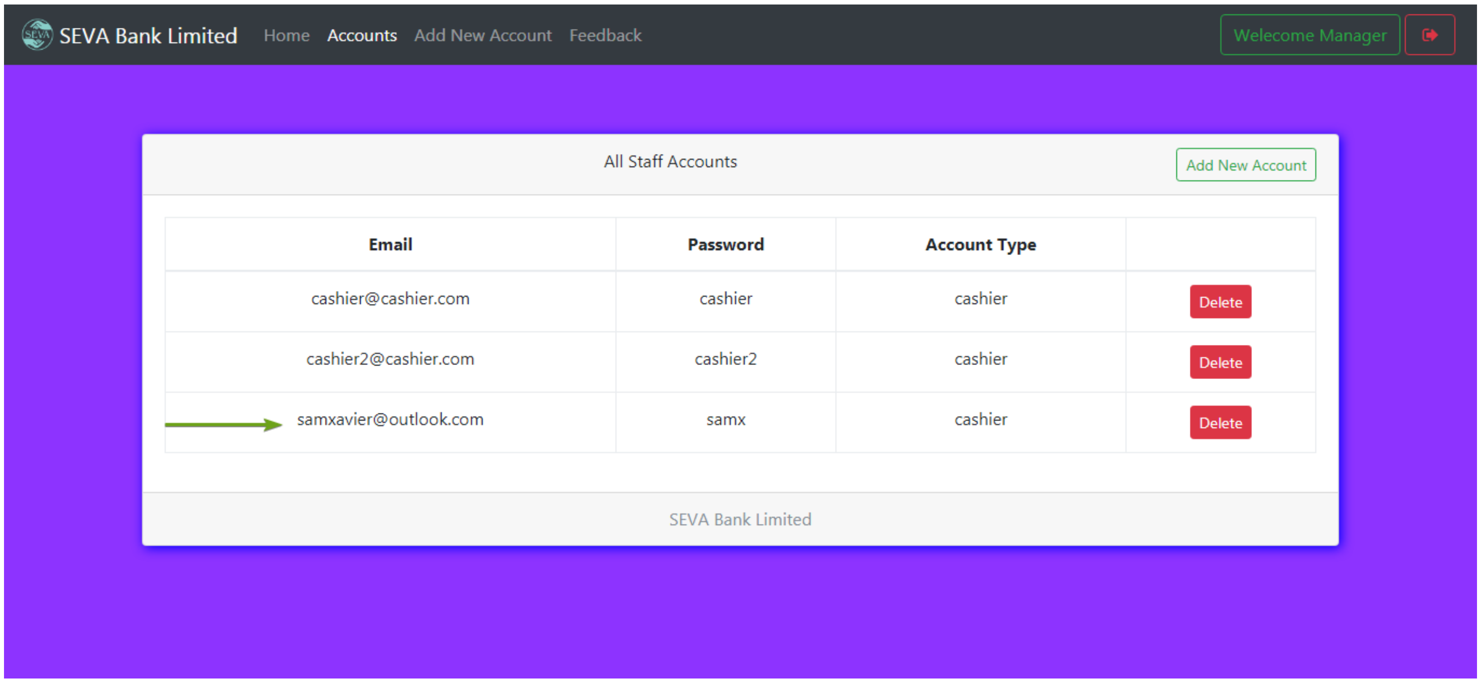
samxavier@outlook.com

••••

Email			
cashier@cashier.com			
cashier2@cashier.com	cashier2	cashier	

SEVA Bank Limited

Figure 5.22: Manager adding new cashier details.

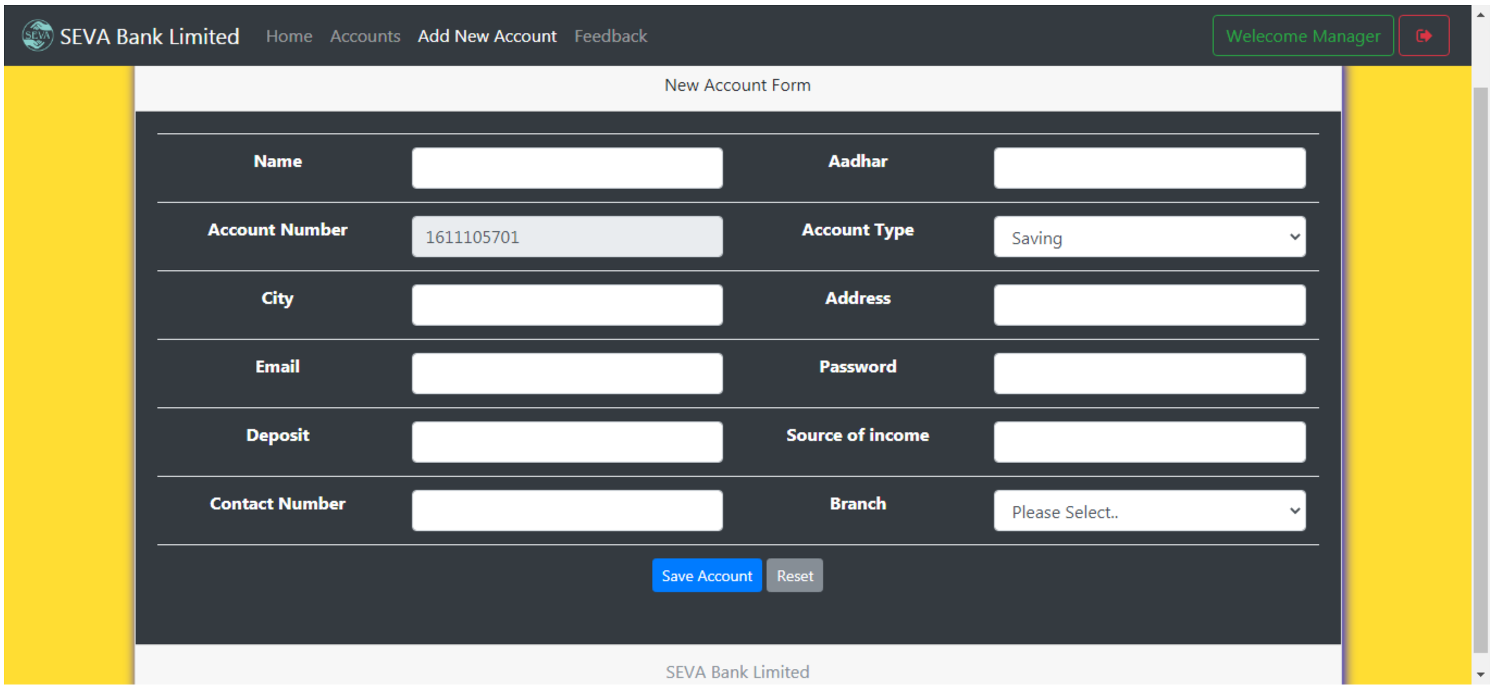


All Staff Accounts

Email	Password	Account Type	
cashier@cashier.com	cashier	cashier	Delete
cashier2@cashier.com	cashier2	cashier	Delete
samxavier@outlook.com	samx	cashier	Delete

SEVA Bank Limited

Figure 5.23 Page showing new cashier added.



SEVA Bank Limited | Home | Accounts | Add New Account | Feedback | Welecome Manager

New Account Form

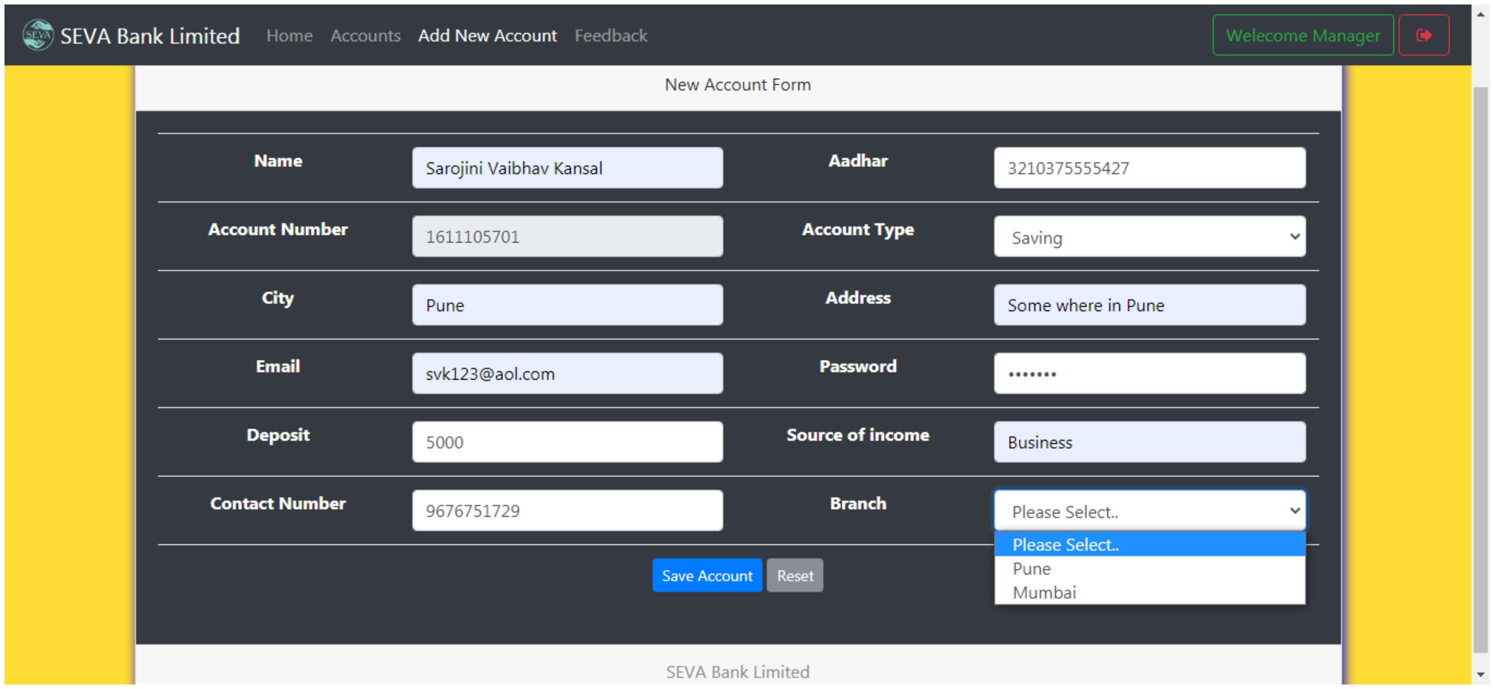
Name		Aadhar	
Account Number	1611105701	Account Type	Saving
City		Address	
Email		Password	
Deposit		Source of income	
Contact Number		Branch	Please Select..

Save Account | Reset

SEVA Bank Limited

Figure 5.24: Add new User account page

Note: *The account number is php time() function which shows time in seconds.*



SEVA Bank Limited | Home | Accounts | Add New Account | Feedback | Welecome Manager

New Account Form

Name	Sarojini Vaibhav Kansal	Aadhar	3210375555427
Account Number	1611105701	Account Type	Saving
City	Pune	Address	Some where in Pune
Email	svk123@aol.com	Password	•••••••
Deposit	5000	Source of income	Business
Contact Number	9676751729	Branch	Please Select..

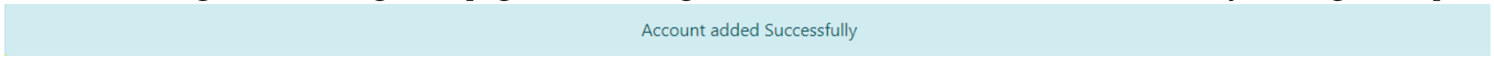
Please Select.. / Pune / Mumbai

Save Account | Reset

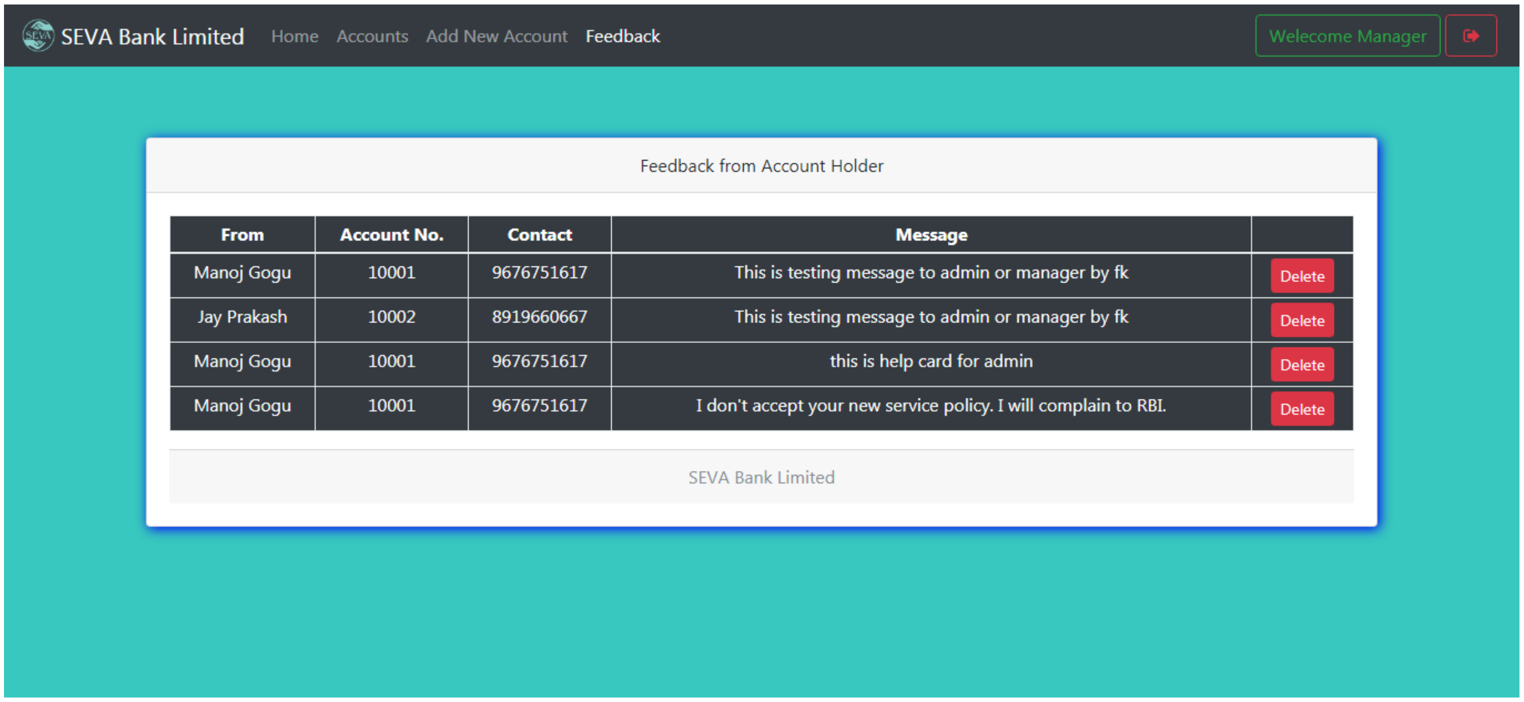
SEVA Bank Limited

Figure 5.25: Manager adding new User account.

After adding account we get the page shown in figure 5.24 with Account Added Successfully message on top



Account added Successfully

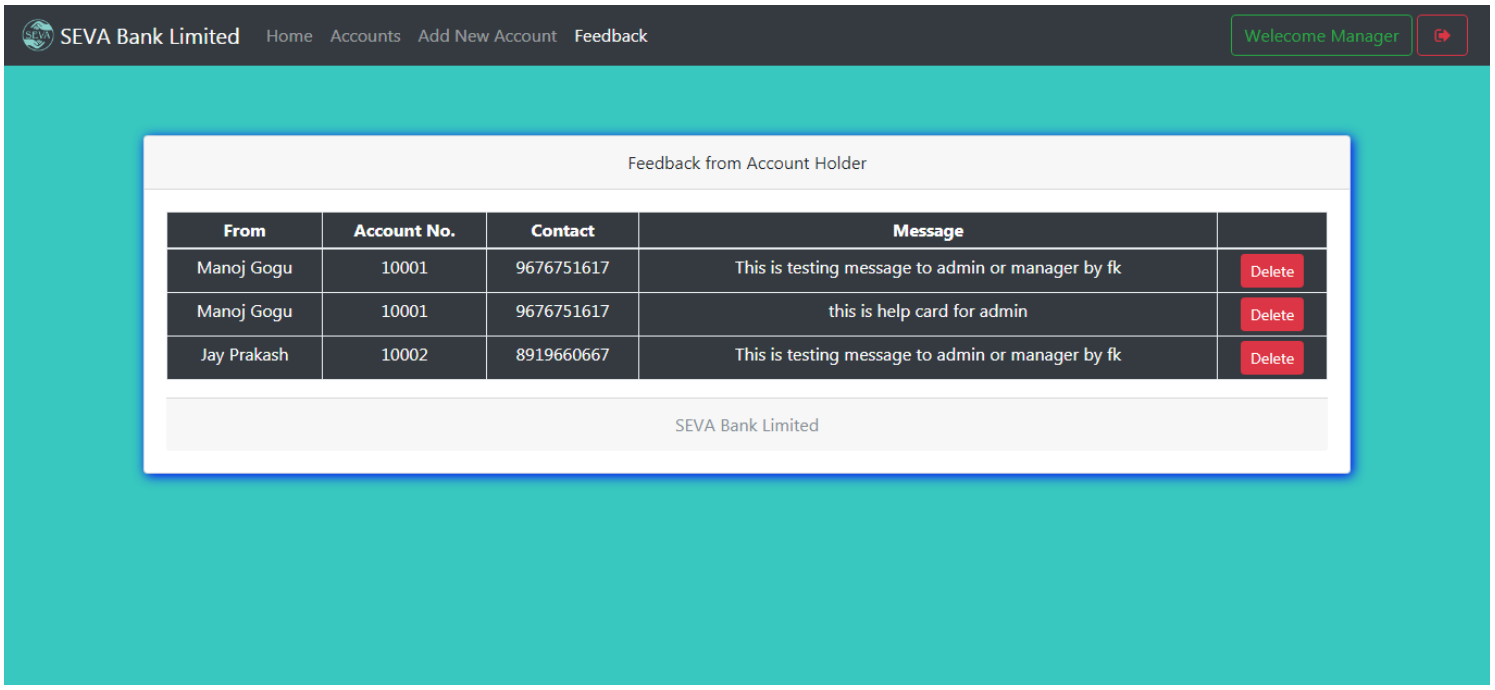


Feedback from Account Holder

From	Account No.	Contact	Message	
Manoj Gogu	10001	9676751617	This is testing message to admin or manager by fk	Delete
Jay Prakash	10002	8919660667	This is testing message to admin or manager by fk	Delete
Manoj Gogu	10001	9676751617	this is help card for admin	Delete
Manoj Gogu	10001	9676751617	I don't accept your new service policy. I will complain to RBI.	Delete

SEVA Bank Limited

Figure 5.26: Feedback page where manager can view user feedbacks.



Feedback from Account Holder

From	Account No.	Contact	Message	
Manoj Gogu	10001	9676751617	This is testing message to admin or manager by fk	Delete
Manoj Gogu	10001	9676751617	this is help card for admin	Delete
Jay Prakash	10002	8919660667	This is testing message to admin or manager by fk	Delete

SEVA Bank Limited

Figure 5.27: Feedback page after deleting a feedback.

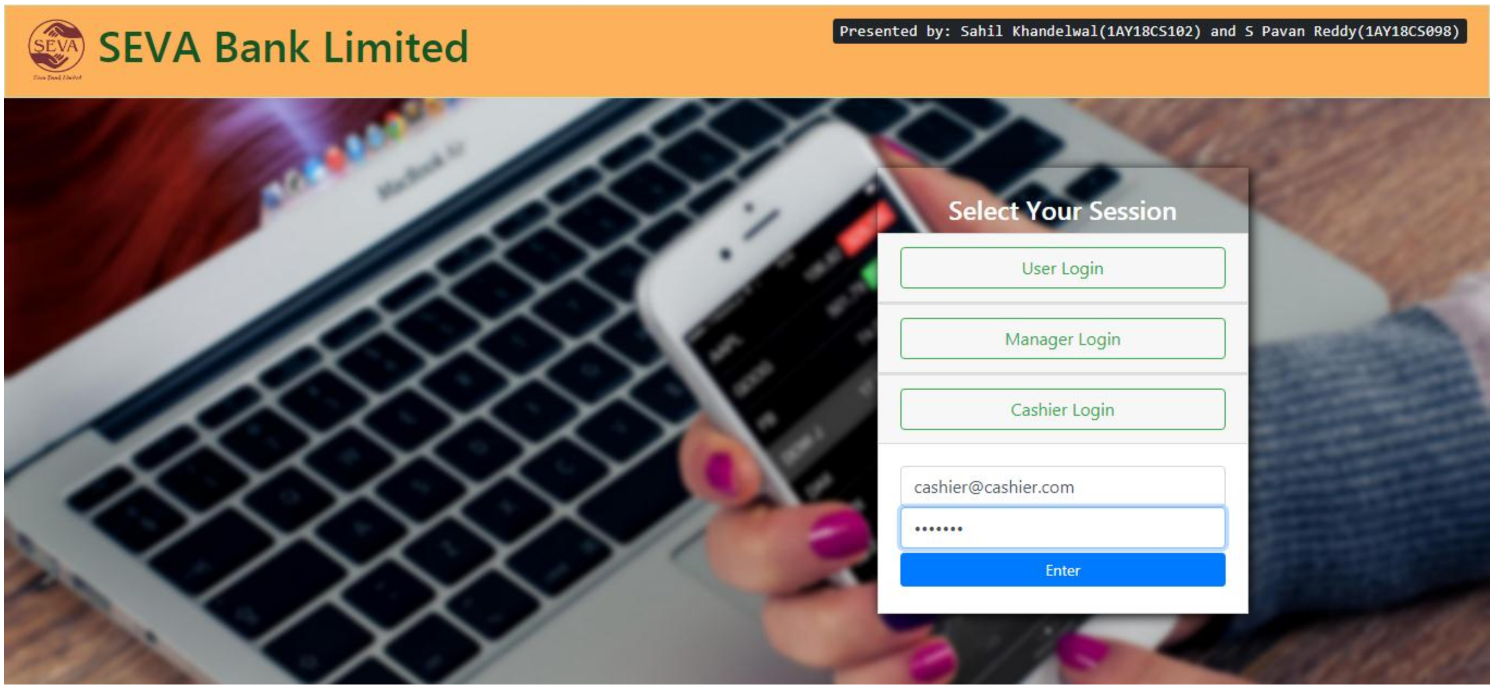


Figure 5.28: Cashier Login Page

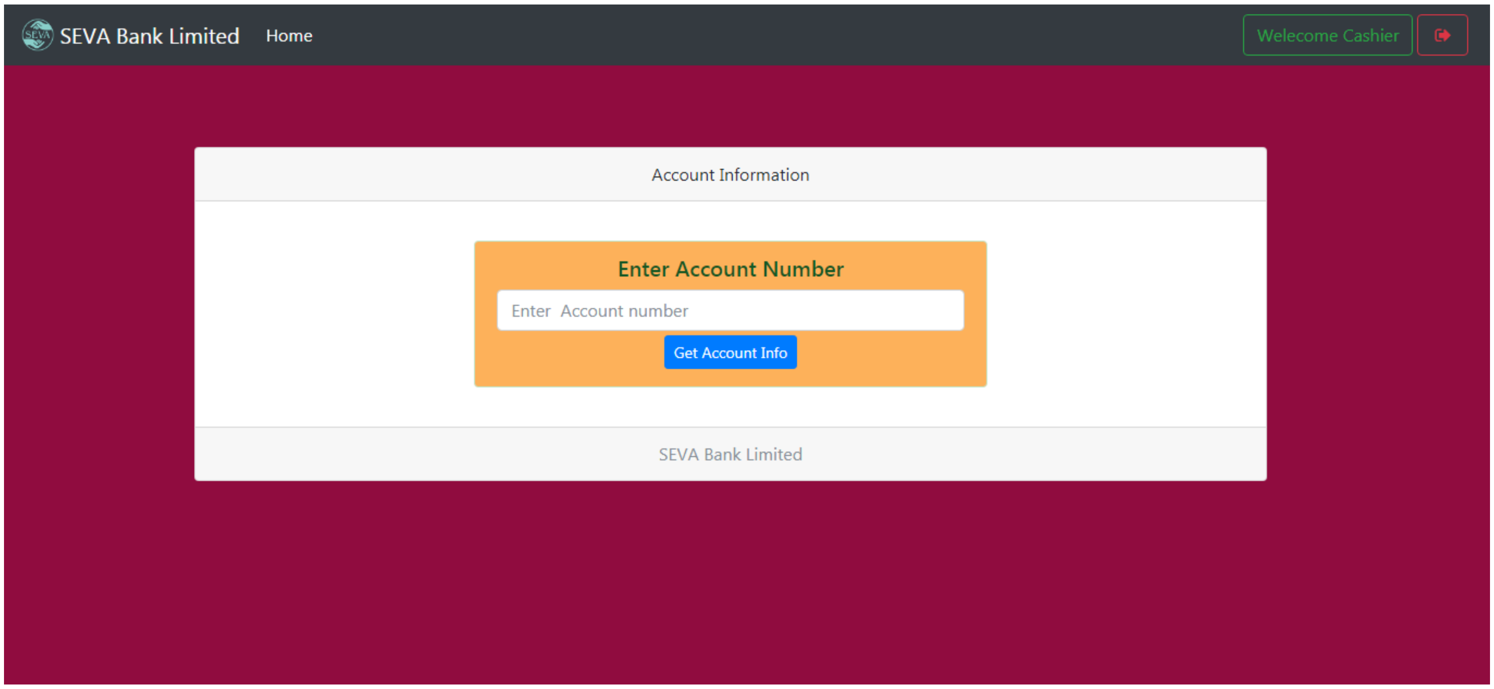
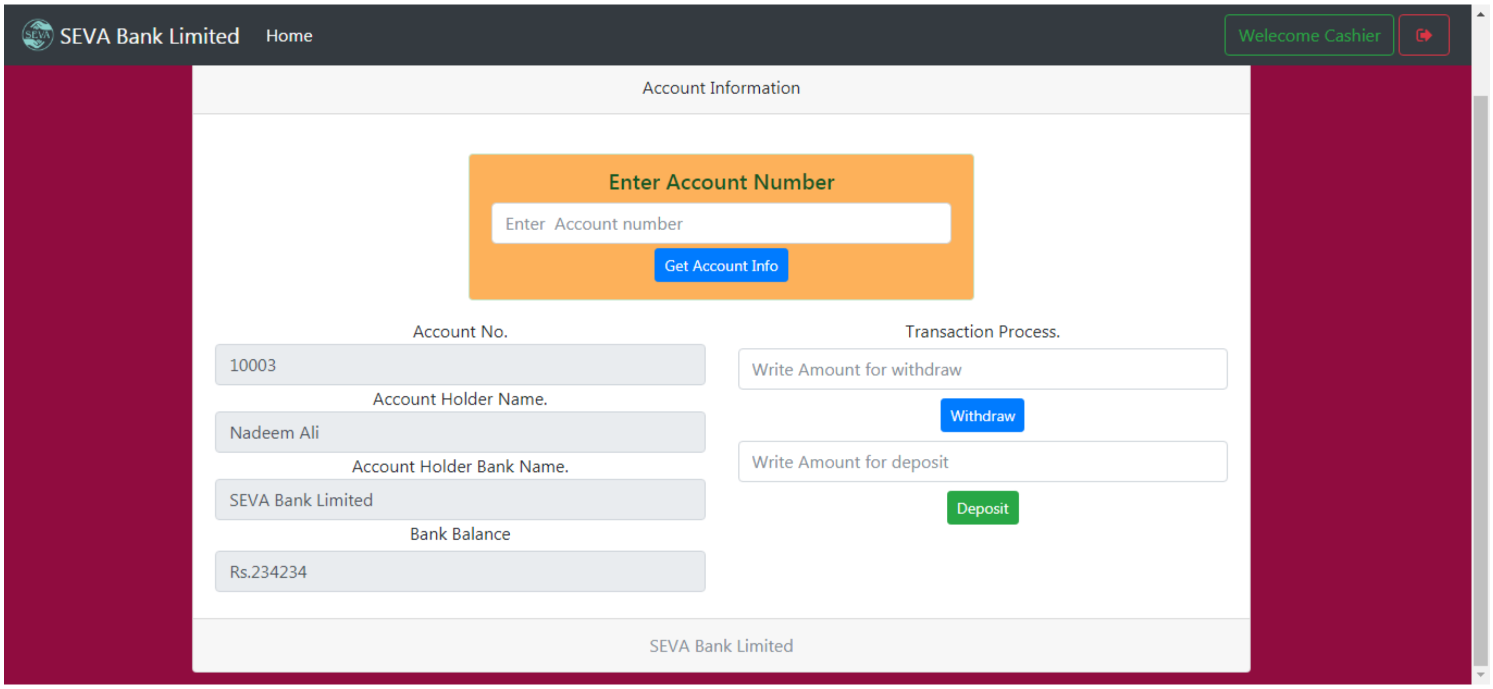


Figure 5.29: Cashier Home page



Account Information

Enter Account Number

Enter Account number

Get Account Info

Account No.

10003

Account Holder Name.

Nadeem Ali

Account Holder Bank Name.

SEVA Bank Limited

Bank Balance

Rs.234234

Transaction Process.

Write Amount for withdraw

Withdraw

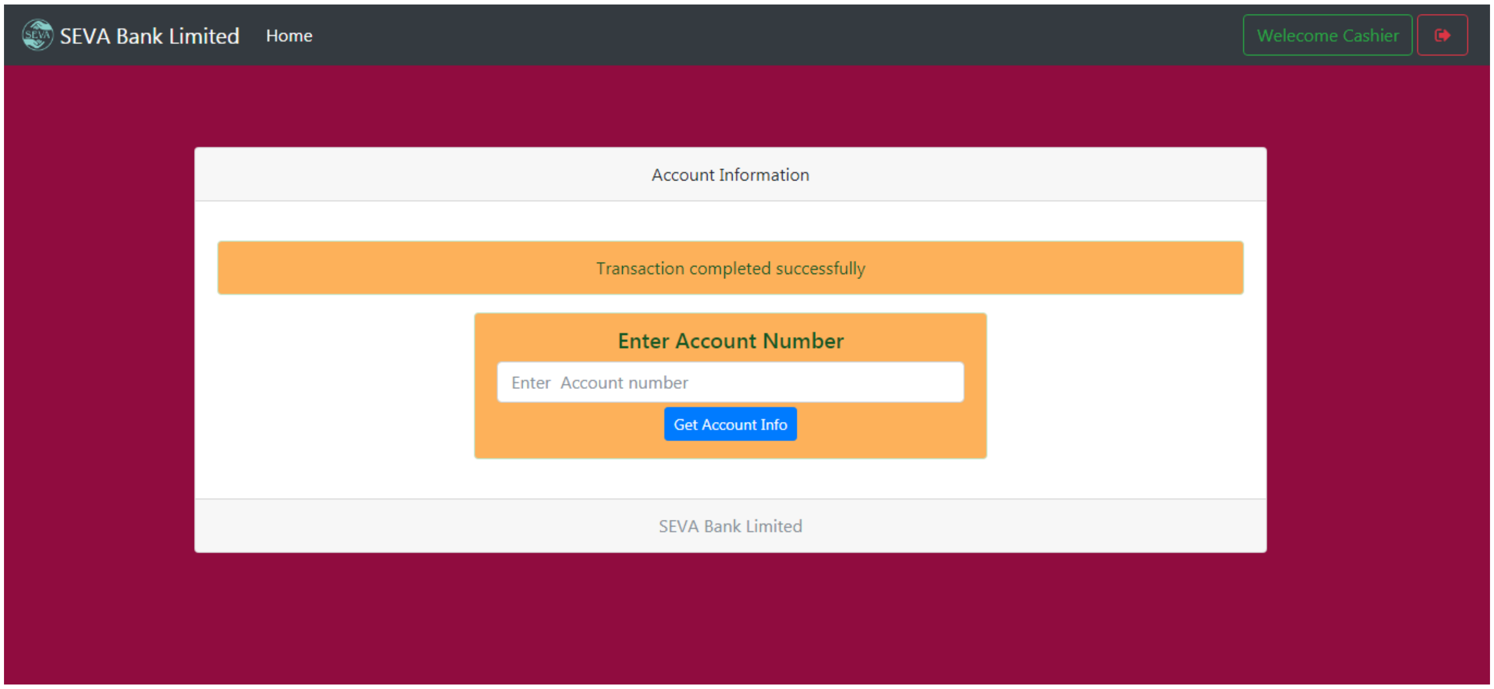
Write Amount for deposit

Deposit

SEVA Bank Limited

Figure 5.30: Cashier Transaction Page (After entering account #).

The cashier can either withdraw or deposit money from/to a bank account. The cashier will receive cash from customer and deposit money into their account or withdraw money and give cash to user.



Account Information

Transaction completed successfully

Enter Account Number

Enter Account number

Get Account Info

SEVA Bank Limited

Figure 5.31: Page shown after any kind of successful transaction.

Chapter 6

Conclusion and Future Enhancements

6.1 Conclusion

The fundamental usage of database systems in the management of a Bank has been successfully demonstrated. We have a records of useraccounts, notices, branches, feedbacks, logins, transactions. These collectively form a basic component of simple Bank database.

6.2 Future Enhancements

Nothing is perfect in this world. So, we are also no exception. Although, we have tried our best to present the information effectively, yet, there can be further enhancement in the Application. We have taken care of all the critical aspects, which need to take care of during the development of the Project.

Like many other things, this project also has some limitations and can further be enhanced by someone who belongs to banking sector and have complete knowledge of the rules for transactions and how transactions between various banks happen. Features such as loan application and repayment can be add to take this project closer and closer to model real world problems and events. In reality Bank management systems are very complex in nature because of the concurrency and protocols involved in Bank transactions.

Chapter 7

References

1. **https://www.w3schools.com/ (multiple pages)**

2. **https://www.educba.com/mysql-trigger/ (multiple pages)**

3. **https://www.dev.mysql.com (multiple pages)**

4. **https://www.stackoverflow.com (multiple pages)**

5. **https://www.computerhope.com/htmcolor.htm**

6. **https://www.tutorialspoint.com (multiple pages)**

7. **https://www.javatpoint.com/php-mysql-connect**