FUNDAMENTAL BANKING SYSTEM

**DELIVERABLE 5**

**Prepared by Tech Coders**

**Version 2.0**

**November 16th, 2020.**

**Table of Contents**

1. Requirements………………………………………………………………………………………………………………………………………4

1.1 Functional Requirements………………………………………………………………………………………………………4

1.1.1 Cashier………………………………………………………………………………………………………………………..4

1.2 Non-Functional Requirements……………………………………………………………………………………………… 5

2. UML Design………………………………………………………………………………………………………………………………………….5

2.1 Class Diagram………………………………………………………………………………………………………………………..5

2.2 Sequence Diagram…………………………………………………………………………………………………………………6

2.3 Use Case Diagram………………………………………………………………………………………………………………….7

3. Test Cases…………………………………………………………………………………………………………………………………………….7

4. User Manual………………………………………………………………………………………………………………………………………17

5. Instructions to Compile……………………………………………………………………………………………………………………..29

6. Enhancements…………………………………………………………………………………………………………………………………..30

**Revision History:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 11/10/2020 | Initial version | Vishnu | Initial version of the document is prepared |
| 11/16/2020 | Final version | Sagar | Final version of the document is prepared with all required components. |
| 11/16/2020 | Revised version | Rushi | Reviewed the document and made changes where necessary. |

**Member Contribution Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Member name** | **Contribution description** | **Overall contribution** | **Note** |
| Gunavardhan reddy | Worked on Transfer functionality. Also worked on 1, 2a parts of deliverable 5 report. | 20% |  |
| Akhil Totakura | Worked on View account functionality and bug fixes around Deposit and Withdraw functionalities, DB scripts and unit test cases creation. Also worked on 2c part of deliverable 5 report. | 20% |  |
| Sagar Tulluru | Worked on Mini statement functionality. Also worked on 2b, 3 parts of deliverable 5 report. | 20% |  |
| Rushi Savulge | Worked on Deposit functionality. Also, worked on 2e 2f 2g parts of deliverable 5 report. | 20% |  |
| Vishnu theja reddy | Worked on Withdraw functionality. Also worked on 2b, 2d part of deliverable 5 report | 20% |  |

**1.Requirements for phase 3**

**1.1 Functional Requirements**

**1.1.1 Cashier**

* Manage Account will be the table containing list of all accounts having Deposit, Withdraw and Mini Statement option to its right.
* In this, Search function will come handy to search for a specific account.

Search criteria will be the Customer Id and/or Account Number. If any relevant account is not found, “No record found” message will be displayed.

* Deposit will be a form to be filled to credit amount into an account.

The form contains the below fields

1. Account Number
2. Account Balance
3. Amount to be deposited

The first two fields will be populated by default and amount to be deposited will be the only field filled out by the cashier.

Entered Amount must be greater than 0. Unless an error message (“Invalid amount”) will be displayed.

* Withdraw will be a form to be filled to debit amount from the account.

The form contains the below fields

1. Account Number
2. Account Balance
3. Withdrawal Amount

The first two fields will be populated by default and Withdrawal amount will be the only field filled out by the cashier.

Withdrawal Amount must be greater than 0 and less than the available balance. Unless an error message (“Invalid Withdrawal Amount” or “Insufficient funds”) will be displayed.

* Mini Statement:

It shall have two options,

1. Selected Date range
2. Number of Transactions

Mini Statement can be generated by selecting either of the above options. If there are no transactions recorded in the selected date range or no transactions are made on account, Mini Statement will display “No records”.

* Transfer will be a form to be filled to transfer from source account to a destination account.

The form contains the below fields

1. Source Account Number
2. Destination Account Number
3. Transfer Amount

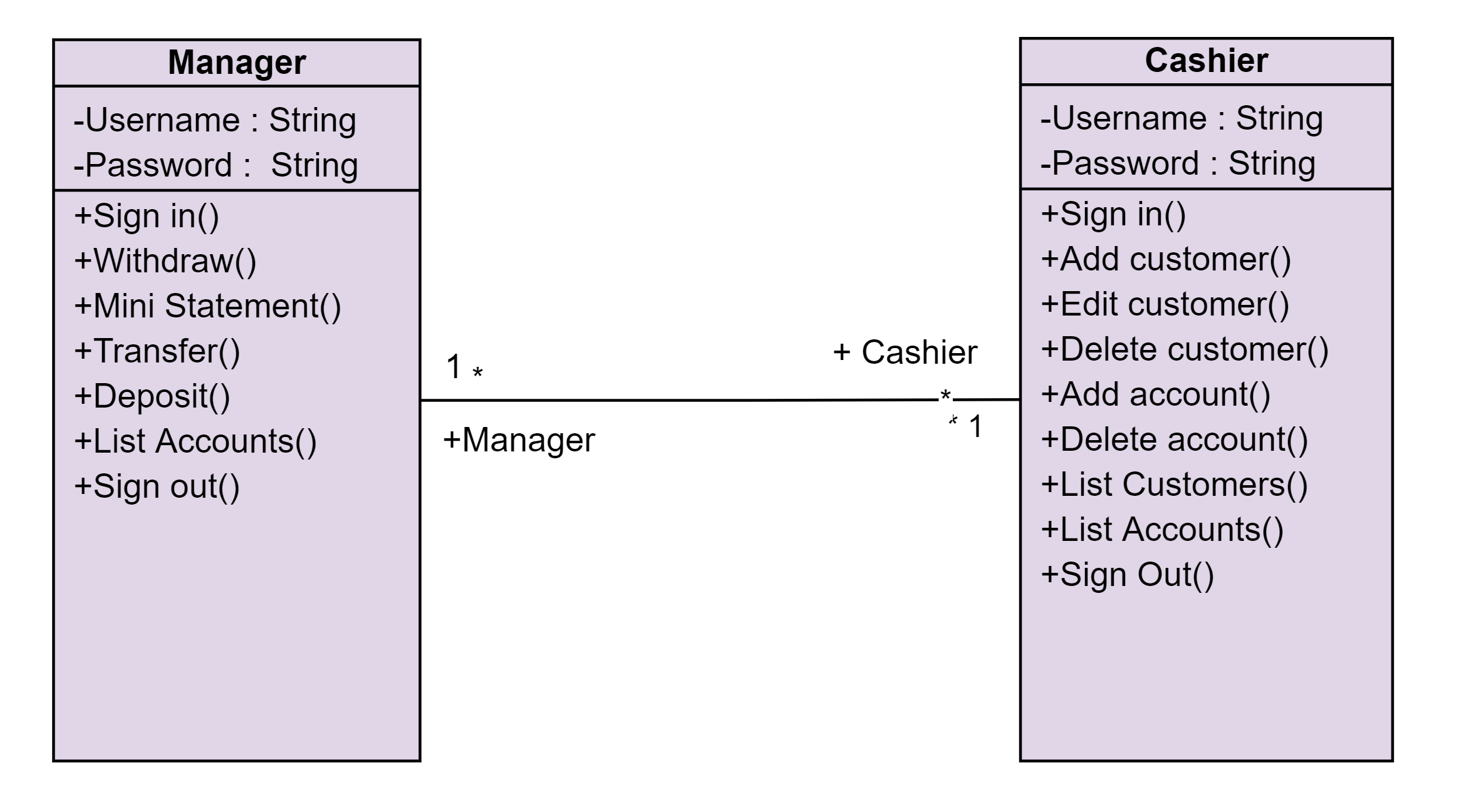
The first two fields will be manually entered by cashier and amount must be greater than 0 and less than the source account balance. If not an error message “insufficient funds” or “Invalid amount” will be displayed on the screen.

**1.2 Non-Functional Requirements**

* Encrypting Database connection strings to make sure no sensitive is revealed.

**2. UML Design**

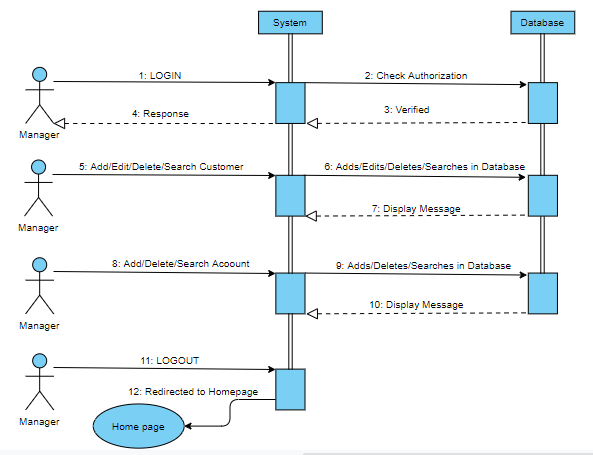
**2.1 Class Diagram**



**Fig: Class Diagram**

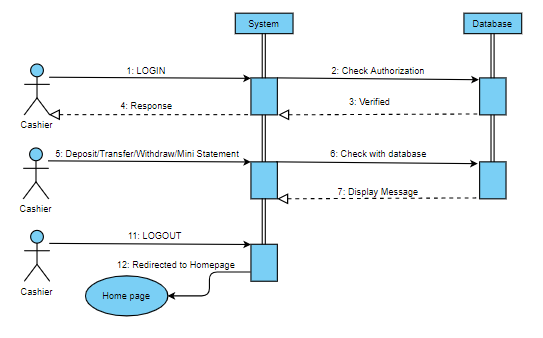
**2.2 Sequence Diagrams**

**Flow Process for Manager:**



**Fig: Sequence Diagram for Manager Role**

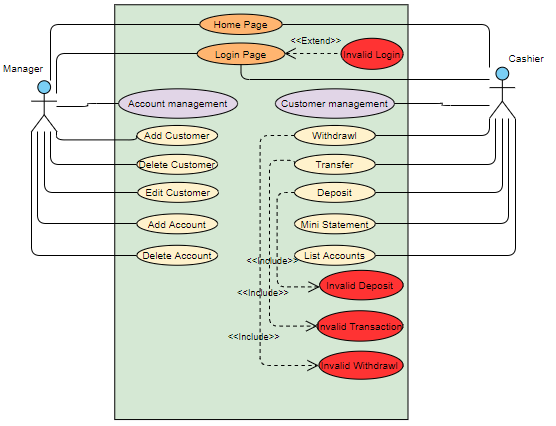
**Flow Process for Cashier:**



**Fig: Sequence Diagram for Cashier Role**

**2.3 Use case Diagram**

**Normal Use Case and Exception Use case:**

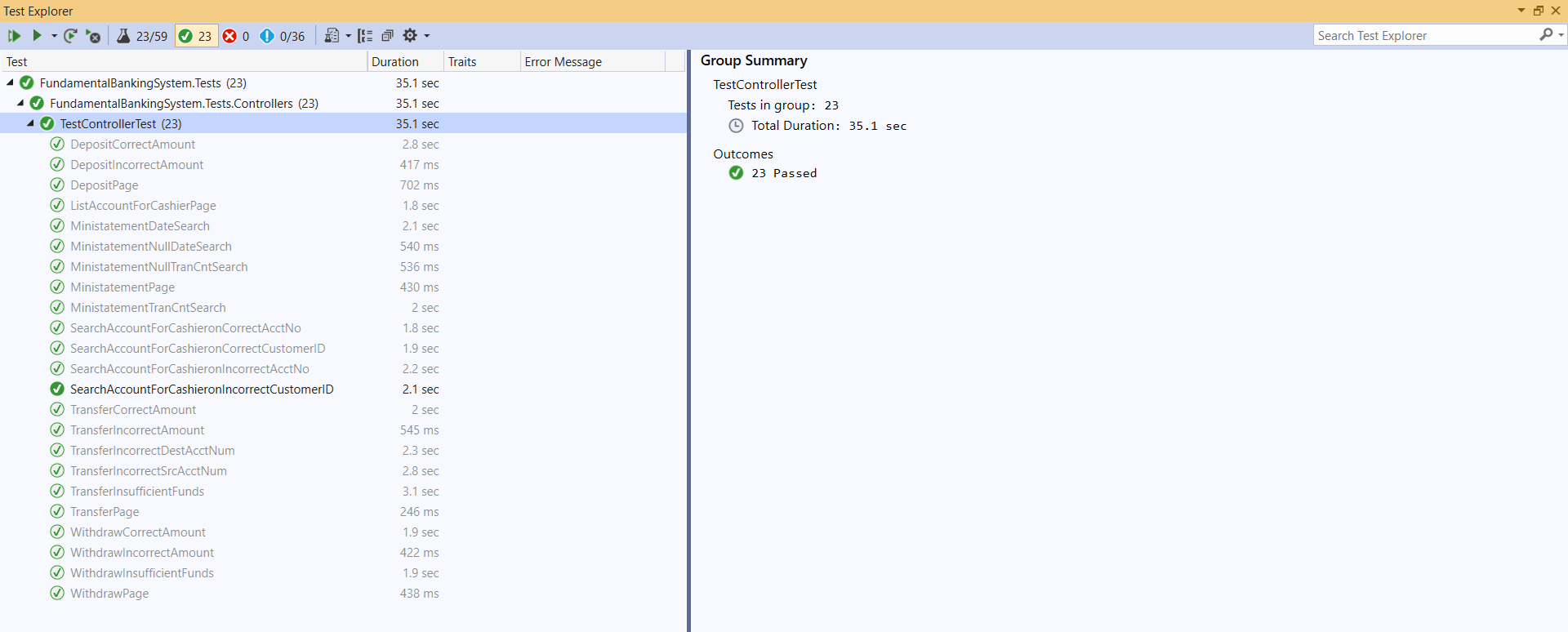
****

**Fig: Use Case Diagram for Functional Behaviors (Normal & Exception case)**

**3. Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Test Case** | **Description** | **Unit** | **Input and Output** | **Status** |
| 1 | ListAccountForCashierPage | Verify whether the view result is ListAccountCashier or not. | ListAccountCashier | Input: Call ListAccountCashier method in Test Controller.  Output: AddNewCustomer method must return it’s view as result. | Passed |
| 2 | SearchAccountForCashieronCorrectAcctNo | Verify whether the search is successful with correct AccountNumber. | ListAccountCashier | Input: AccountNumber of account  Output: Successful search of account. | Passed |
| 3 | SearchAccountForCashieronIncorrectAcctNo | Verify whether the search is unsuccessful with incorrect AccountNumber. | ListAccountCashier | Input: AccountNumber of account  Output: Unsuccessful search of account. | Passed |
| 4 | SearchAccountForCashieronCorrectCustomerID | Verify whether the search is successful with correct ID of customer. | ListAccountCashier | Input: ID of customer  Output: Successful search of account with customerid. | Passed |
| 5 | SearchAccountForCashieronIncorrectCustomerID | Verify whether the search is unsuccessful with incorrect ID of customer. | ListAccountCashier | Input: ID of customer  Output: Unsuccessful search of account with customerid. | Passed |
| 6 | DepositPage | Verify whether the view result is Deposit or not. | Deposit | Input: Call Deposit method in Test Controller.  Output: Deposit method must return it’s view as result. | Passed |
| 7 | DepositIncorrectAmount | Verify whether the Deposit is not successful when amount is incorrect such as negative amount | Deposit | Input: Call Deposit method in Test Controller with the amount to be deposited into account  Output: Unsuccessful deposit with an error message of “Sorry! Please check the Amount entered”. | Passed |
| 8 | DepositCorrectAmount | Verify whether the Deposit is successful when amount is correct | Deposit | Input: Call Deposit method in Test Controller with the amount to be deposited into account  Output: Successful deposit and redirect to ListAccountCashier page with a success message of “Deposit done Successfully”. | Passed |
| 9 | WithdrawPage | Verify whether the view result is Withdraw or not. | Withdraw | Input: Call Withdraw method in Test Controller.  Output: Withdraw method must return it’s view as result. | Passed |
| 10 | WithdrawIncorrectAmount | Verify whether the Withdraw is not successful when amount is incorrect such as negative amount | Withdraw | Input: Call Withdraw method in Test Controller with the amount to be withdrawn from the account  Output: Unsuccessful withdrawal with an error message of “Sorry! Please check the Amount entered”. | Passed |
| 11 | WithdrawInsufficientFunds | Verify whether the Withdraw is not successful when amount is greater than the existing balance | Withdraw | Input: Call Withdraw method in Test Controller with the amount to be withdrawn from the account  Output: Unsuccessful withdrawal with an error message of “Sorry! Insufficient funds”. | Passed |
| 12 | WithdrawCorrectAmount | Verify whether the Withdraw is successful when amount is correct | Withdraw | Input: Call Withdraw method in Test Controller with the amount to be withdrawn from the account  Output: Successful withdrawal and redirect to ListAccountCashier page with a success message of “Withdraw done Successfully”. | Passed |
| 13 | TransferPage | Verify whether the view result is Transfer or not. | Transfer | Input: Call Transfer method in Test Controller.  Output: Transfer method must return it’s view as result. | Passed |
| 14 | TransferIncorrectSrcAcctNum | Verify whether the Transfer is successful with incorrect source AccountNumber. | Transfer | Input: Call Transfer method in Test Controller with the amount to be transferred from the account.  Output: Unsuccessful Transfer with an error message of “Source account doesn't exist”. | Passed |
| 15 | TransferIncorrectDestAcctNum | Verify whether the Transfer is successful with incorrect destination AccountNumber. | Transfer | Input: Call Transfer method in Test Controller with the amount to be transferred from the account.  Output: Unsuccessful Transfer with an error message of “Destination account doesn't exist”. | Passed |
| 16 | Transfer IncorrectAmount | Verify whether the Transfer is not successful when amount is incorrect such as negative amount | Transfer | Input: Call Transfer method in Test Controller with the amount to be transferred from the account  Output: Unsuccessful Transfer with an error message of “Sorry! Please check the Amount entered”. | Passed |
| 17 | Transfer InsufficientFunds | Verify whether the Transfer is not successful when amount is greater than the existing source account balance | Transfer | Input: Call Transfer method in Test Controller with the amount to be transferred from the account  Output: Unsuccessful Transfer with an error message of “Sorry! Insufficient funds”. | Passed |
| 18 | Transfer CorrectAmount | Verify whether the Transfer is successful when amount is correct | Transfer | Input: Call Transfer method in Test Controller with the amount to be transferred from the account  Output: Successful Transfer and redirect to ListAccountCashier page with a success message of “Transfer done Successfully”. | Passed |
| 19 | MinistatementPage | Verify whether the view result is Ministatement or not. | Ministatement | Input: Call Ministatement method in Test Controller.  Output: Ministatement method must return it’s view as result. | Passed |
| 20 | MinistatementNullDateSearch | Verify whether the Ministatement results are null if don’t select any dates. | Ministatement | Input: Call Ministatement method in Test Controller to display a Ministatement of an account based on either date or no of transactions.  Output: No records displayed on Ministatement but with an error message “Please select the dates!”. | Passed |
| 21 | MinistatementNullTranCntSearch | Verify whether the Ministatement results are null if don’t select no of transactions. | Ministatement | Input: Call Ministatement method in Test Controller to display a Ministatement of an account based on either date or no of transactions.  Output: No records displayed on Ministatement but with an error message “Select number of transactions!” | Passed |
| 22 | MinistatementDateSearch | Verify whether the Ministatement results are not null if we select the dates properly. | Ministatement | Input: Call Ministatement method in Test Controller to display a Ministatement of an account based on date.  Output: Transaction records displayed on Ministatement made between the selected dates. | Passed |
| 23 | MinistatementTranCntSearch | Verify whether the Ministatement results are null if we select no of transactions value from the dropdown. | Ministatement | Input: Call Ministatement method in Test Controller to display a Ministatement of an account based on no of transactions.  Output: Transaction records displayed on Ministatement based on the selected transaction count. | Passed |

**New Test Case Results:**

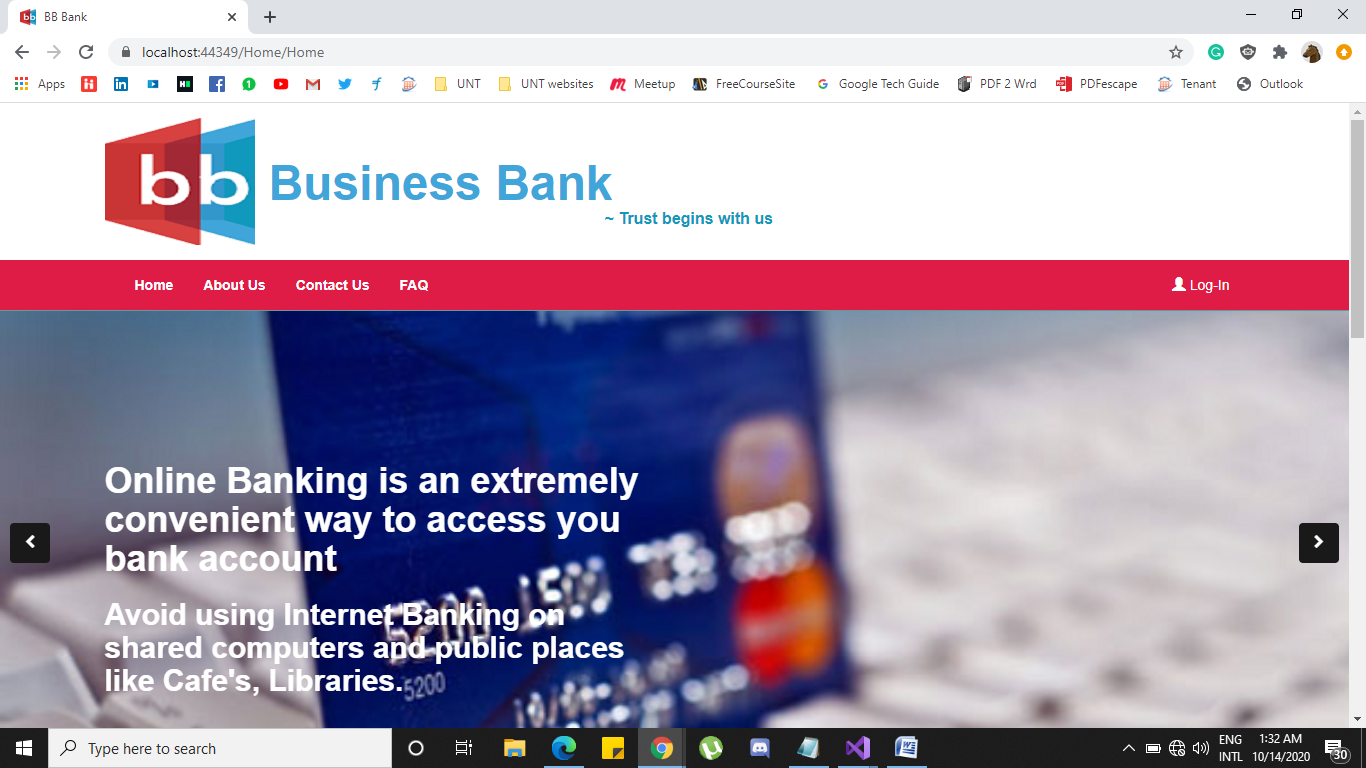


**System test cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Test Case** | **Expected Output** | **Actual Output** | **Requirements validated** | **Status** |
| **Home Page** | | | | | |
| 1 | Home page should contain logo, Name of the bank, navigation bar with Home, About us, Contact, FAQ and Login buttons, slider, Features section, contact info and copyright information | Able to see all the details mentioned in the Home page. | Everything is displayed as per requirements. | 3.1.1, 3.1.2,3.1.3,3.1.4,  3.1.5,3.1.6 | Passed |
| **About Us** | | | | | |
| 2 | About Page should contain information about the bank and the facilities provided. | Able to see information about bank and its facilities. | The page has got all the needed information mentioned in requirements. | 3.2.1 | Passed |
| **FAQ** | | | | | |
| 3 | FAQ page should contain the most frequently asked questions about the bank. | Able to see all the FAQ’s. | The page has the FAQ’s as per the requirements. | 3.2.2 | Passed |
| **Login** | | | | | |
| 4 | DB Admin should create login roles for the application. Login Page should have username and password fields, login and reset buttons. | Able to see all the details mentioned in the requirement. | The page is designed as per the requirement. | 3.3.1,3.3.2 | Passed |
| 5 | Wrong/No username and password should give respective validation messages. | Able to see the error validation messages | Validation messages are displayed as per system requirement. | 3.3.3.3.3.5 | Passed |
| 6 | Correct Login credentials | Redirect to respective role pages. | Control is redirected to respective role pages as per the requirement. | 3.3.4 | Passed |
| **Logout** | | | | | |
| 7 | Logout from the session | Able to come back to login | Takes back to login upon clicking on logout as per the requirement specification. | 3.5.1 | Passed |
| **Manager** | | | | | |
| 8 | Manager home page should contain logo, name of the bank, navigation bar with Home, Customer Management, Account Management, Logout buttons, Manager image, contact and copyright info. | Able to see all the details mentioned in the Manager home page. | Everything is displayed as per requirements. | 3.6.1,3.6.2,3.6.3 | Passed |
| 9 | Customer Management should be a dropdown and have the below values Add Customer, Manage Customer. | Able to see the values in the dropdown | Options displayed as per the system requirement. | 3.6.4 | Passed |
| 10 | Account Management should be a dropdown and have the below values Add Account, Manage Account. | Able to see the values in the dropdown | Options displayed as per the system requirement. | 3.6.7 | Passed |
| 11 | Able to add customer when correct details are entered, and respective validation messages are displayed upon incorrect details. | Able to add customers and display validation messages. | Everything is done as per requirement. | 3.6.5 | Passed |
| 12 | Able to view, edit, delete customer when correct details are entered, and respective validation messages are displayed upon incorrect details. | Able to view, edit, delete customer and display validation messages. | Everything is done as per requirement. | 3.6.6 | Passed |
| 13 | Able to add account when correct details are entered, and respective validation messages are displayed upon incorrect details. | Able to add account and display validation messages. | Everything is done as per requirement. | 3.6.8 | Passed |
| 14 | Able to view, edit, delete account when correct details are entered, and respective validation messages are displayed upon incorrect details. | Able to view, edit, delete account and display validation messages. | Everything is done as per requirement. | 3.6.9 | Passed |
| Cashier | | | | | |
| 15 | Cashier home page should contain logo, name of the bank, navigation bar with Home, Account Management, Logout buttons, Cashier image, contact and copyright info. | Able to see all the details mentioned in the Cashier home page. | Everything is displayed as per requirements. | 3.7.1,3.7.2,3.7.3 | Passed |
| 16 | Account Management should be a dropdown and have the below values Manage Account and Transfer. | Able to see the values in the dropdown | Options displayed as per the system requirement. | 3.7.4 | Passed |
| 17 | Manage account has search, deposit, withdraw and Ministatement options | Able to see all the details mentioned. | Everything is displayed as per requirements. | 3.7.5 | Passed |
| 18. | Deposit contains the below fields   1. Account Number 2. Account Balance 3. Amount to be deposited   When entered correct amount deposit should be success otherwise a respective error message will be displayed. | Able to see all the details mentioned. | Everything is displayed as per requirements. | 3.7.6 | Passed |
| 19 | Withdraw contains the below fields   1. Account Number 2. Account Balance 3. Withdrawal Amount   When entered correct amount withdraw should be success otherwise a respective error message will be displayed. | Able to see all the details mentioned. | Everything is displayed as per requirements. | 3.7.7 | Passed |
| 20. | Ministatement contains the below options   1. Selected Date range 2. Number of Transactions   When entered correct account number Ministatement should display the transaction on that account otherwise an error message will be displayed. | Able to see all the details mentioned. | Everything is displayed as per requirements. | 3.7.8 | Passed |
| 21. | Transfer contains the below fields   1. Source Account Number 2. Destination Account Number 3. Transfer Amount   When entered correct details transfer should be success otherwise a respective error message will be displayed. | Able to see all the details mentioned. | Everything is displayed as per requirements. | 3.7.9 | Passed |

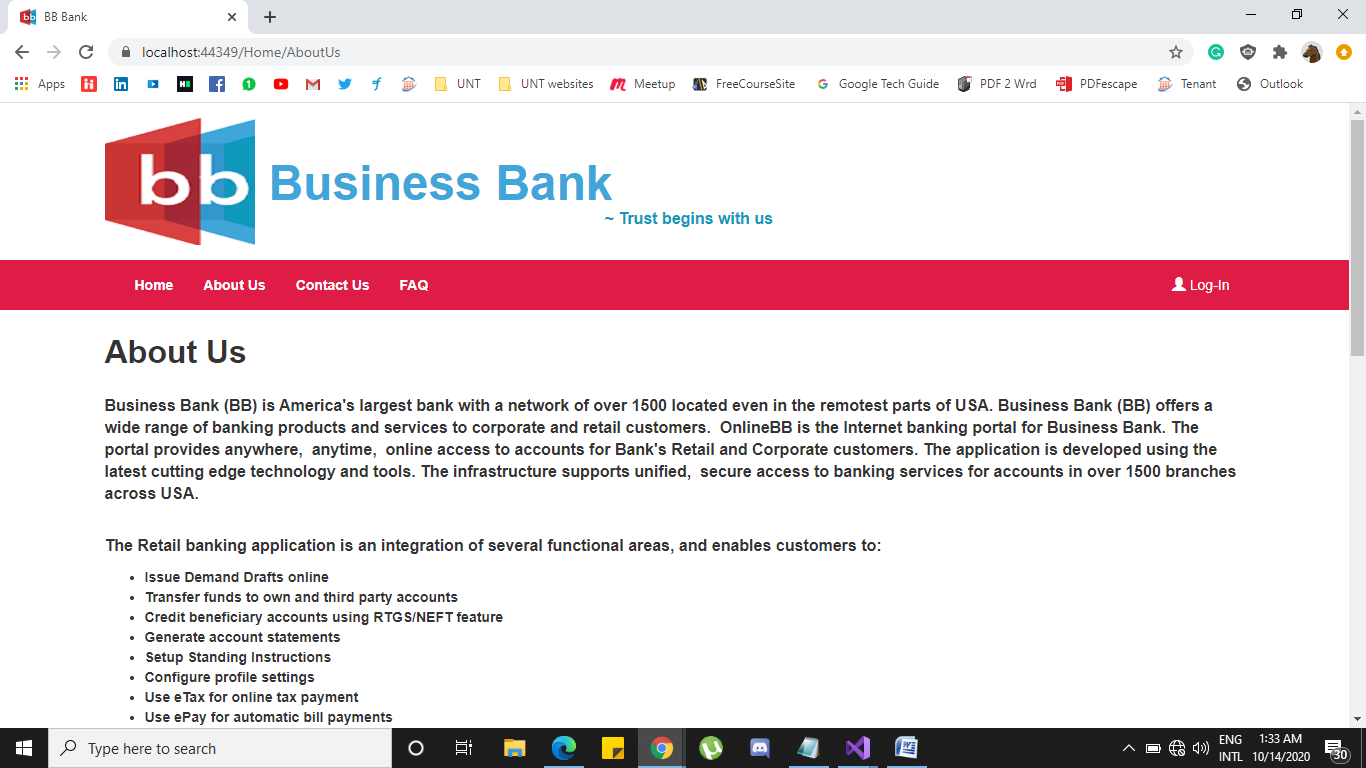
**4. User Manual**

1. This is the Home Page user can see in the beginning.

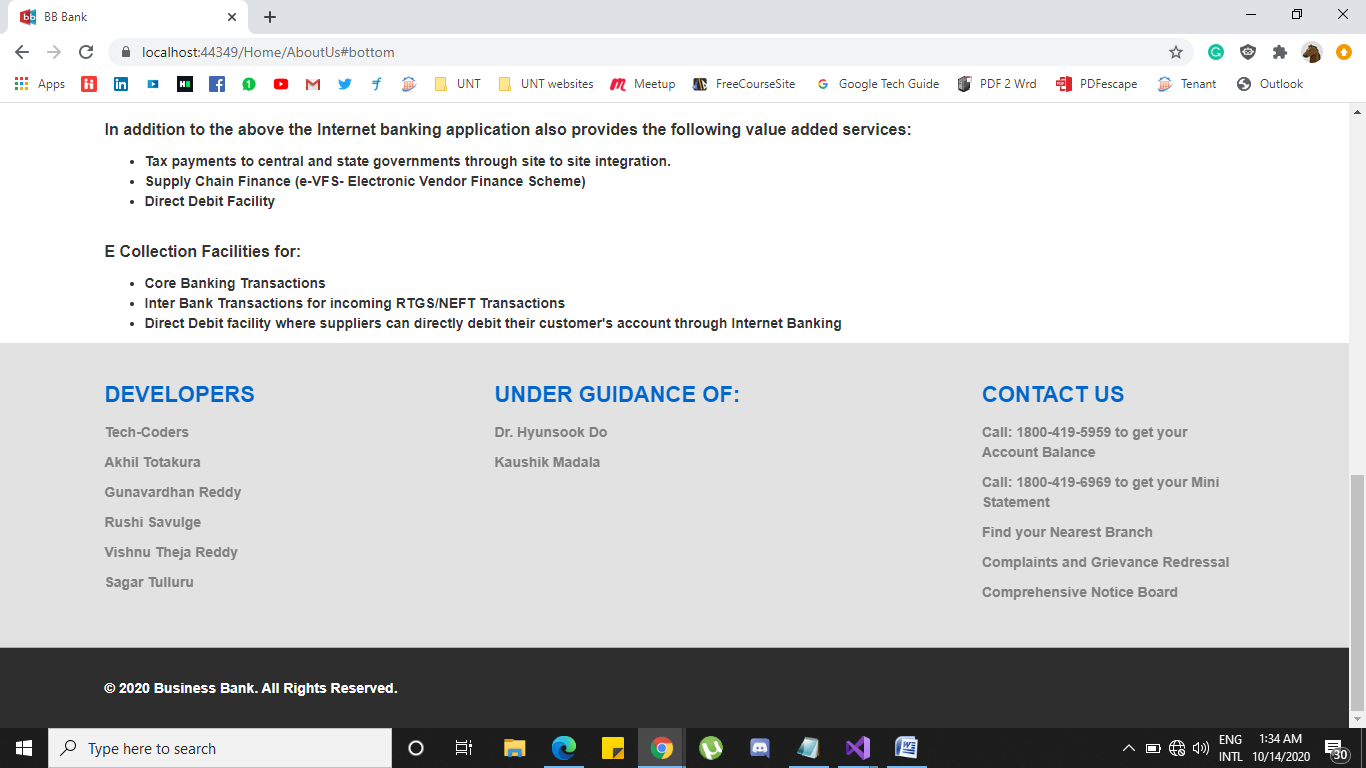


1. In Home page navigation bar there are various functionalities such as Home, About Us, Contact Us, FAQ and Login.

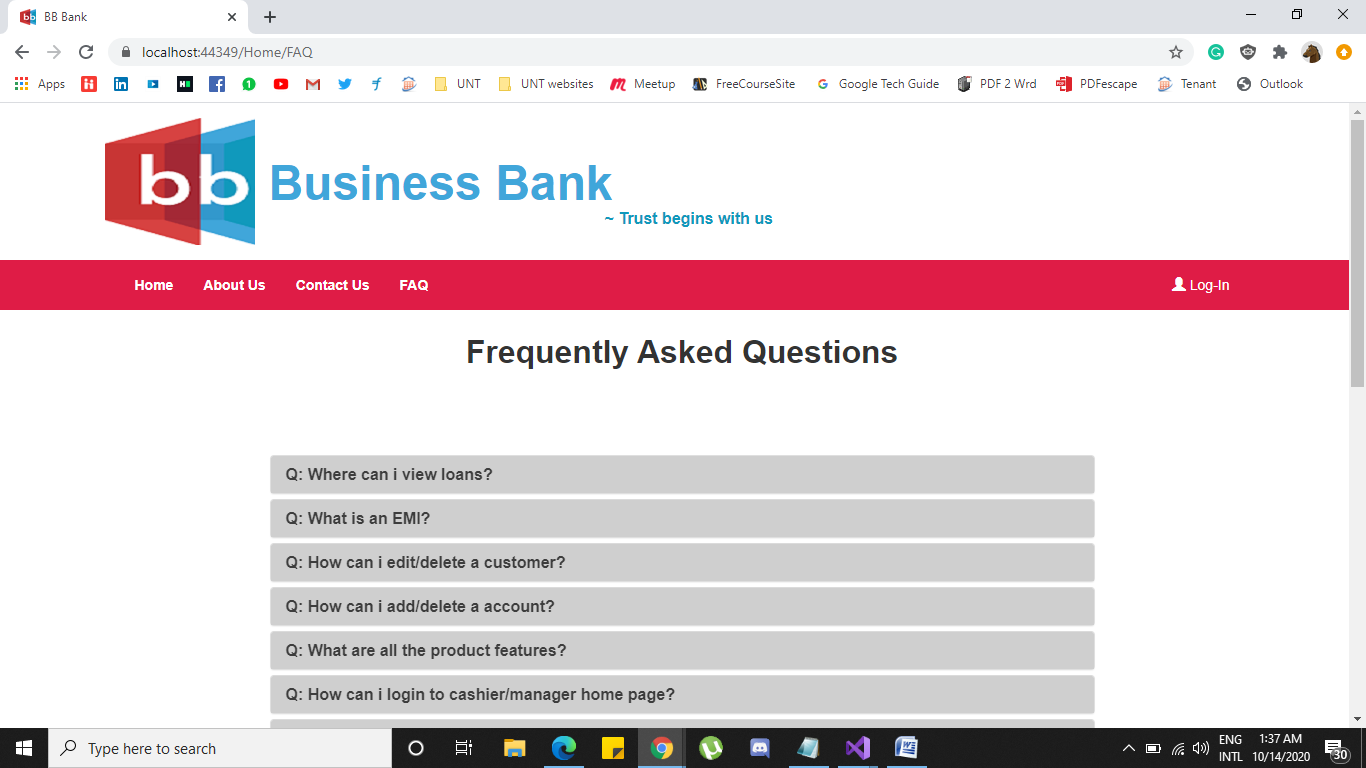
* If User clicks on About Us button, then user will be redirected to about us page where user can see various details about the bank, Features and Facilities.



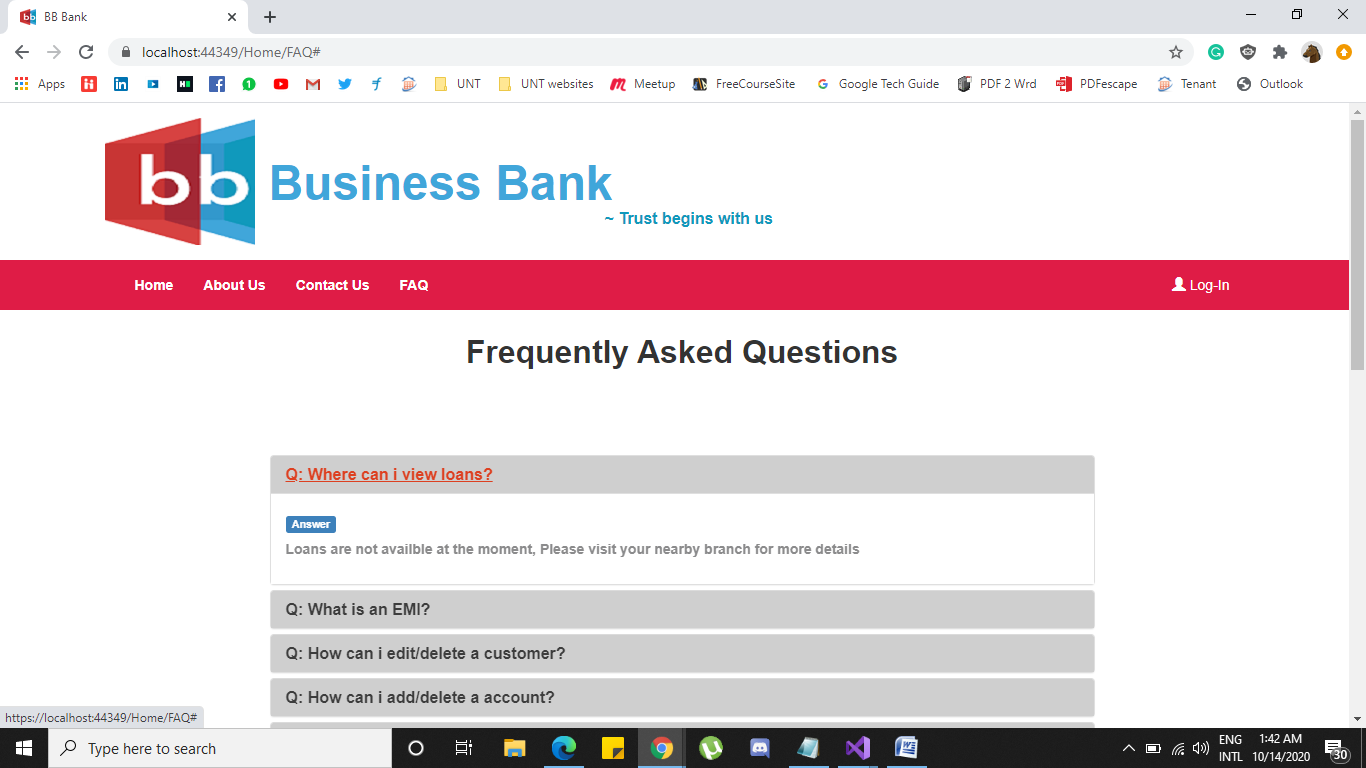
* If User clicks on Contact Us button, then user will be redirected to Contact us page where user can see helpline number, nearest branch and many more.



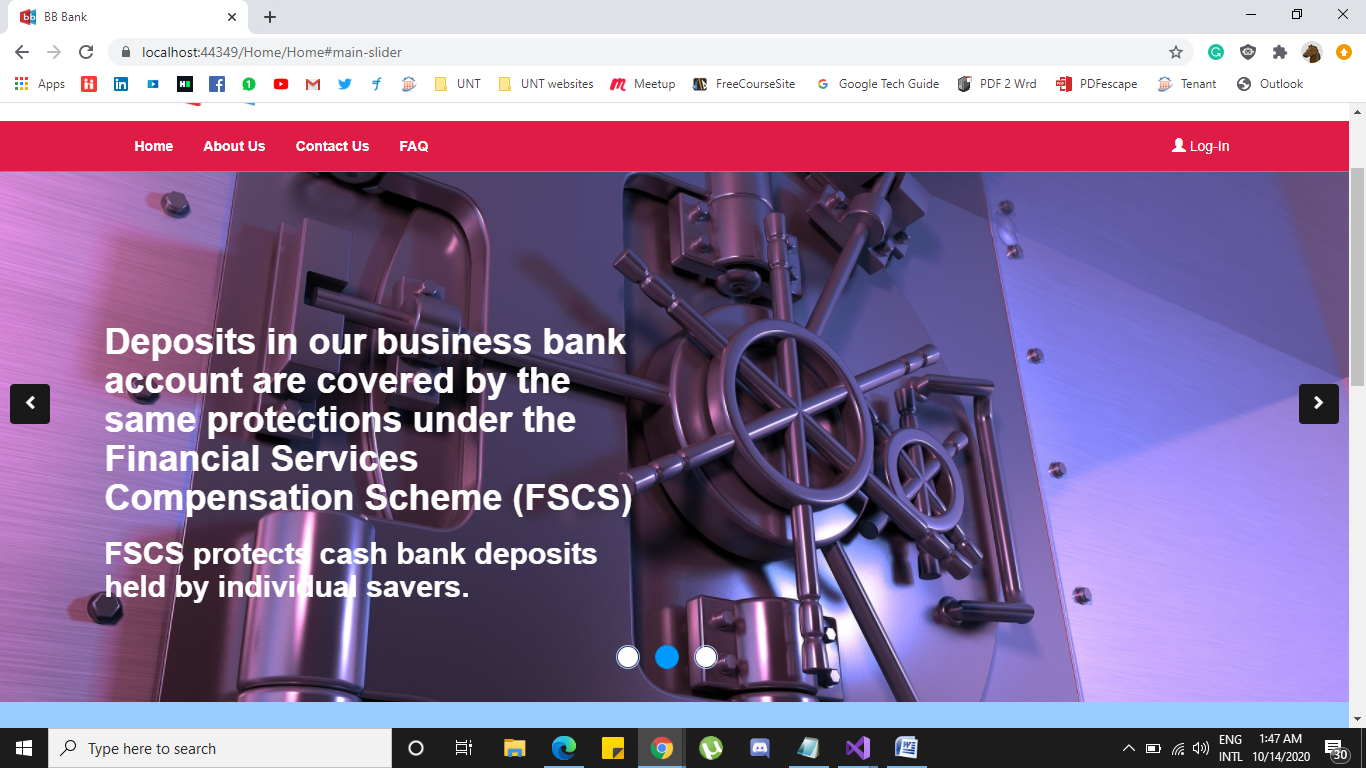
* If User clicks on FAQ button, then user will be redirected to FAQ page where user can see various questions that are frequently asked by other users.



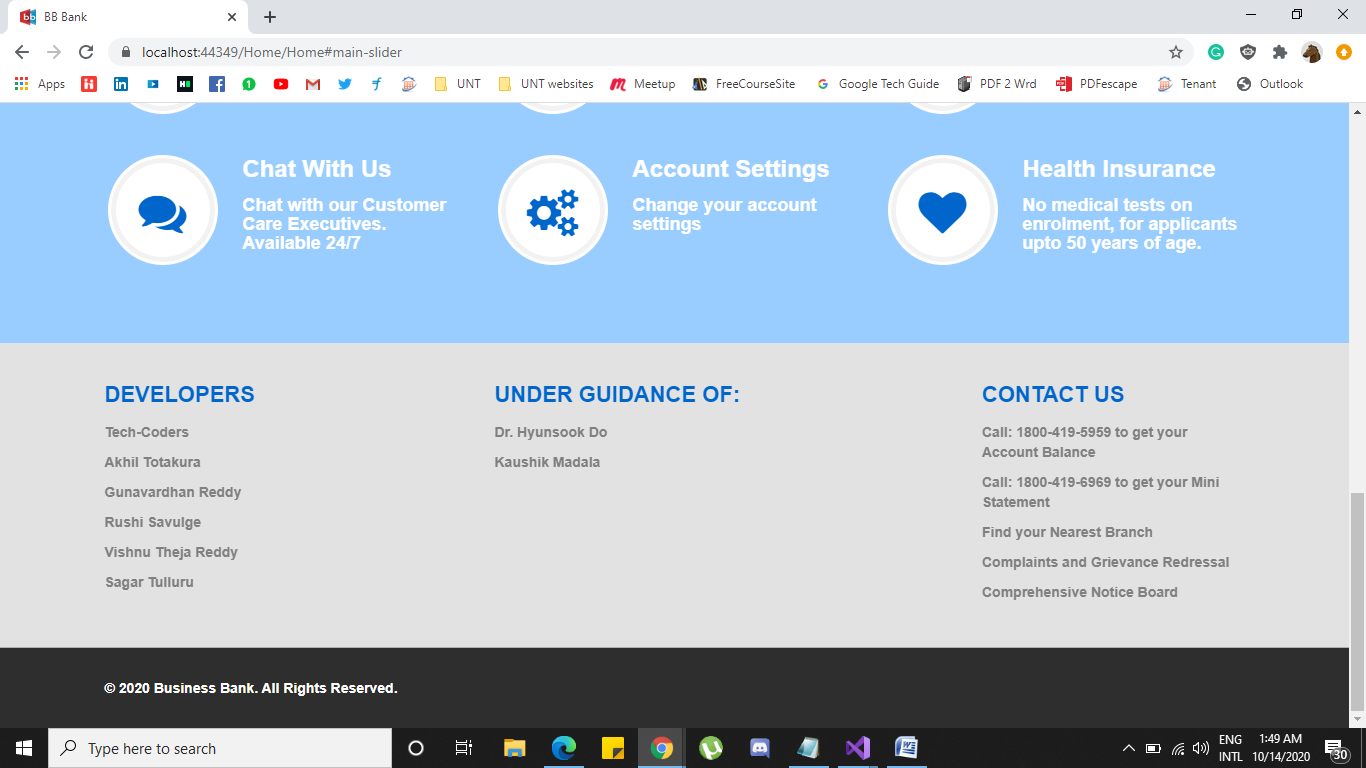
* When user clicks on any individual question, user will be shown its corresponding answer.



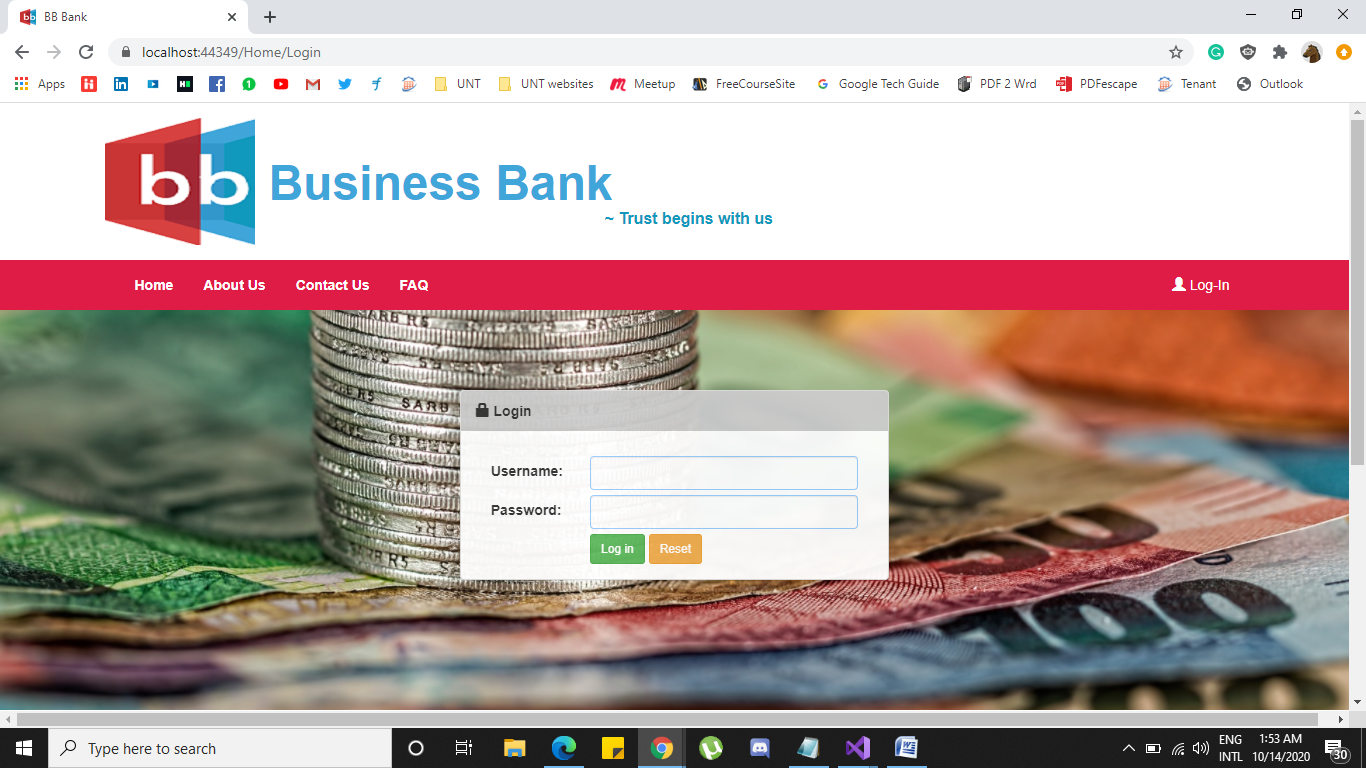
1. There is a slider provided in the homepage which displays various product features of the bank. User will also have option to slide them using the arrow marks at the edges.



1. At the bottom of the page, User will also have information regarding developers of this project, under guidance of info, Contact details and copy rights information



1. If user clicks on Login button in home page then user will be redirected to login page where user can enter details of Username and Password.

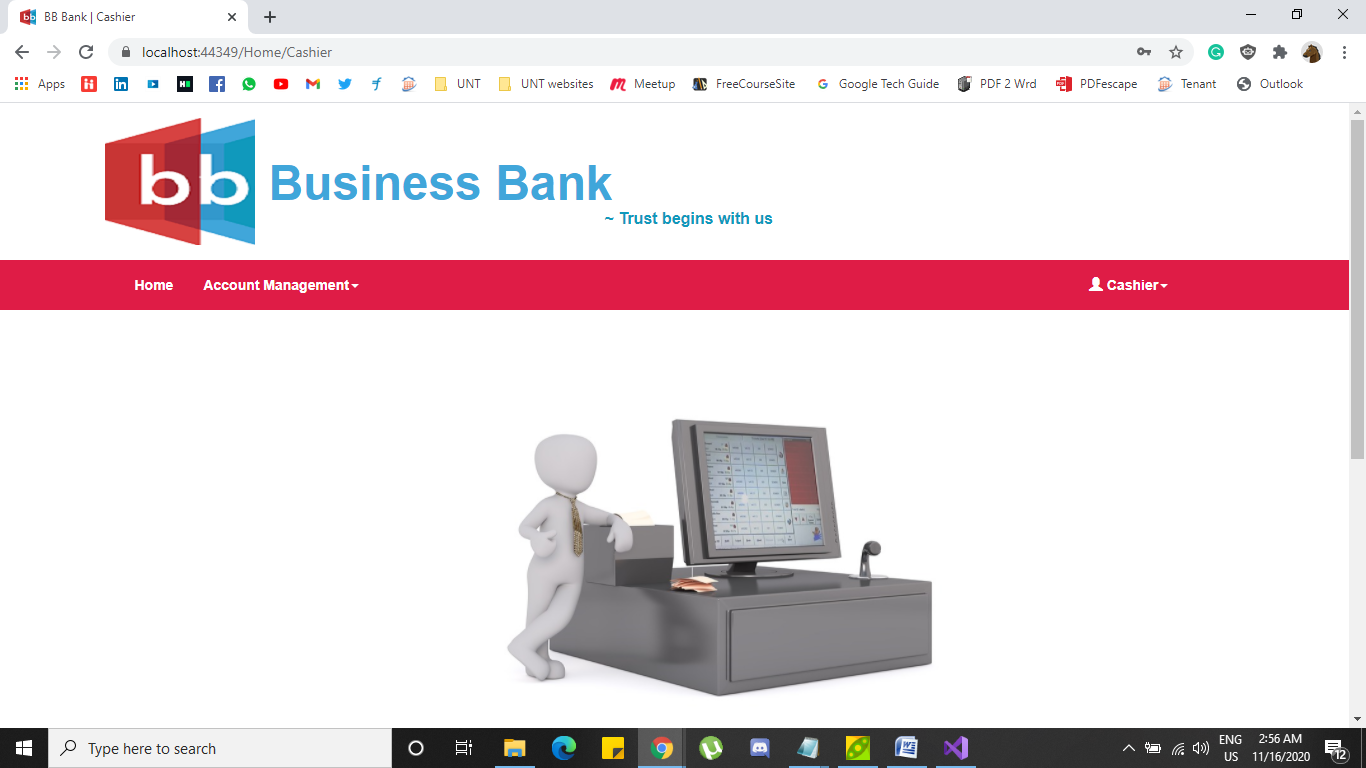


**Cashier Log in:**

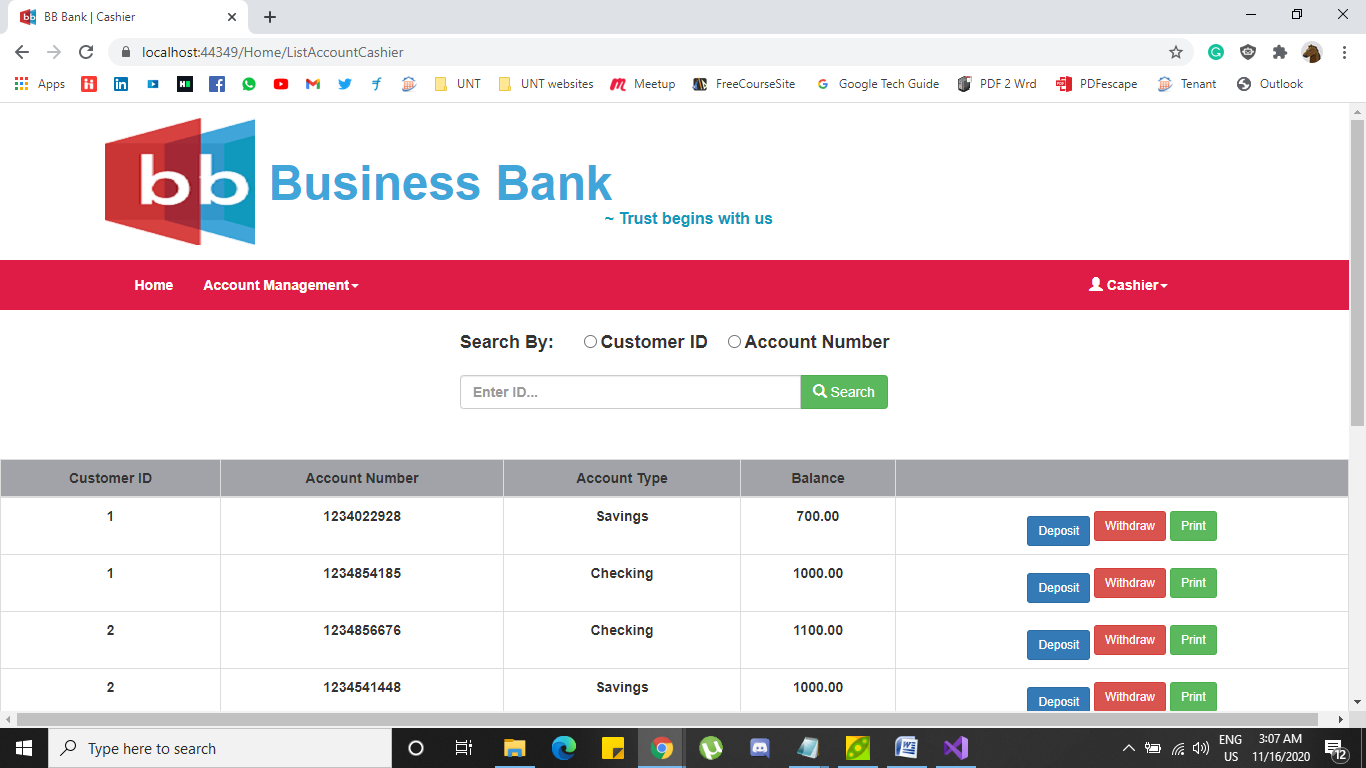
Username: **Cashier**

Password: **bb@ch**

If user using the above credentials tries to login then Cashier log in page.

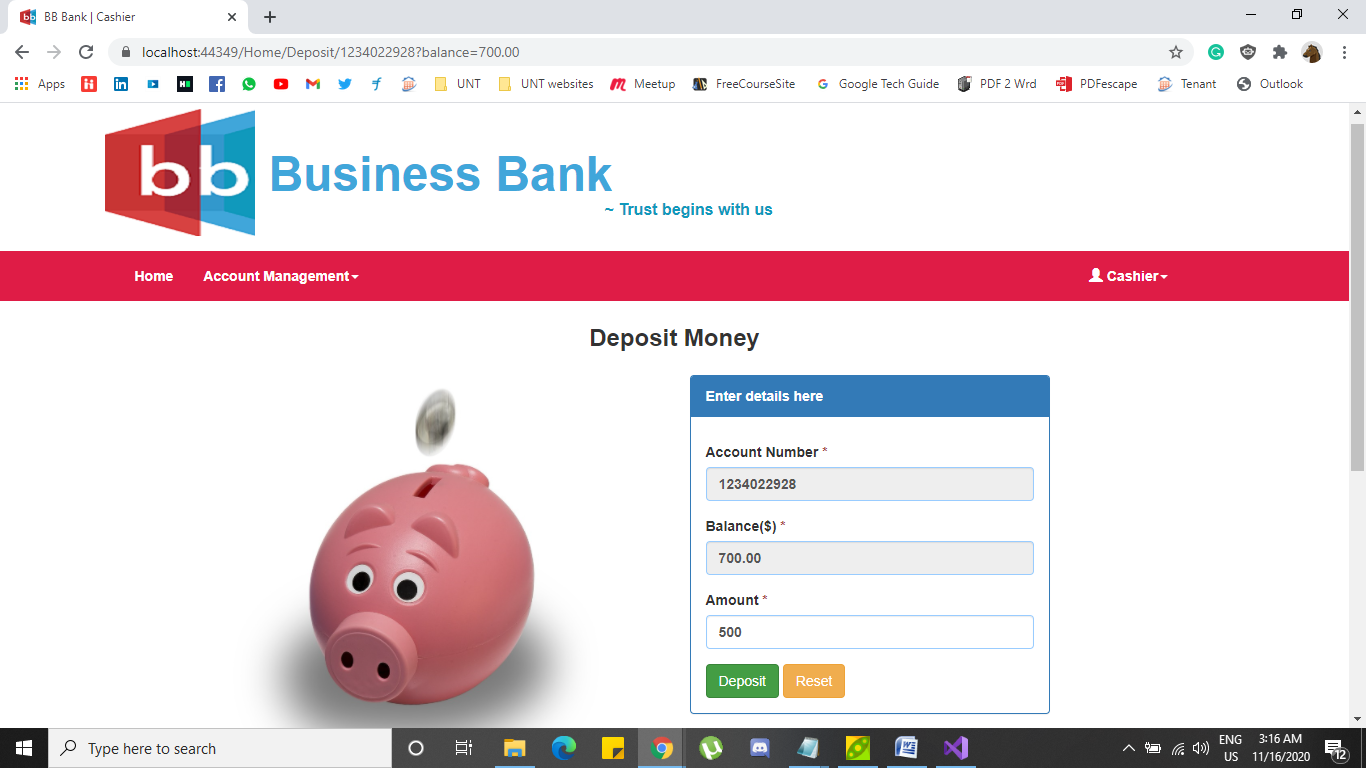
****

Under Account management dropdown, **Manage Account** functionality is selected and manage account page pops out and the page looks like

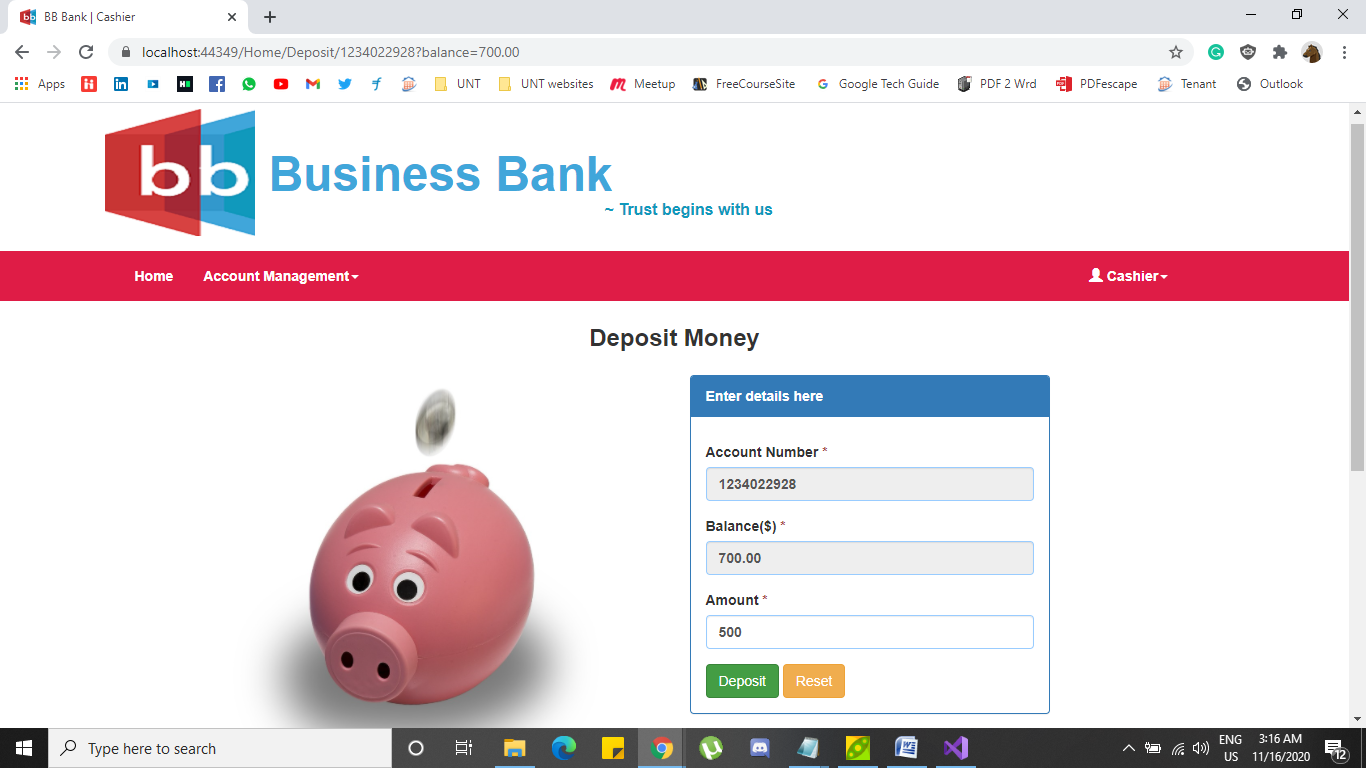


Here you can search any customer by there customer ID and also you can search with account number.

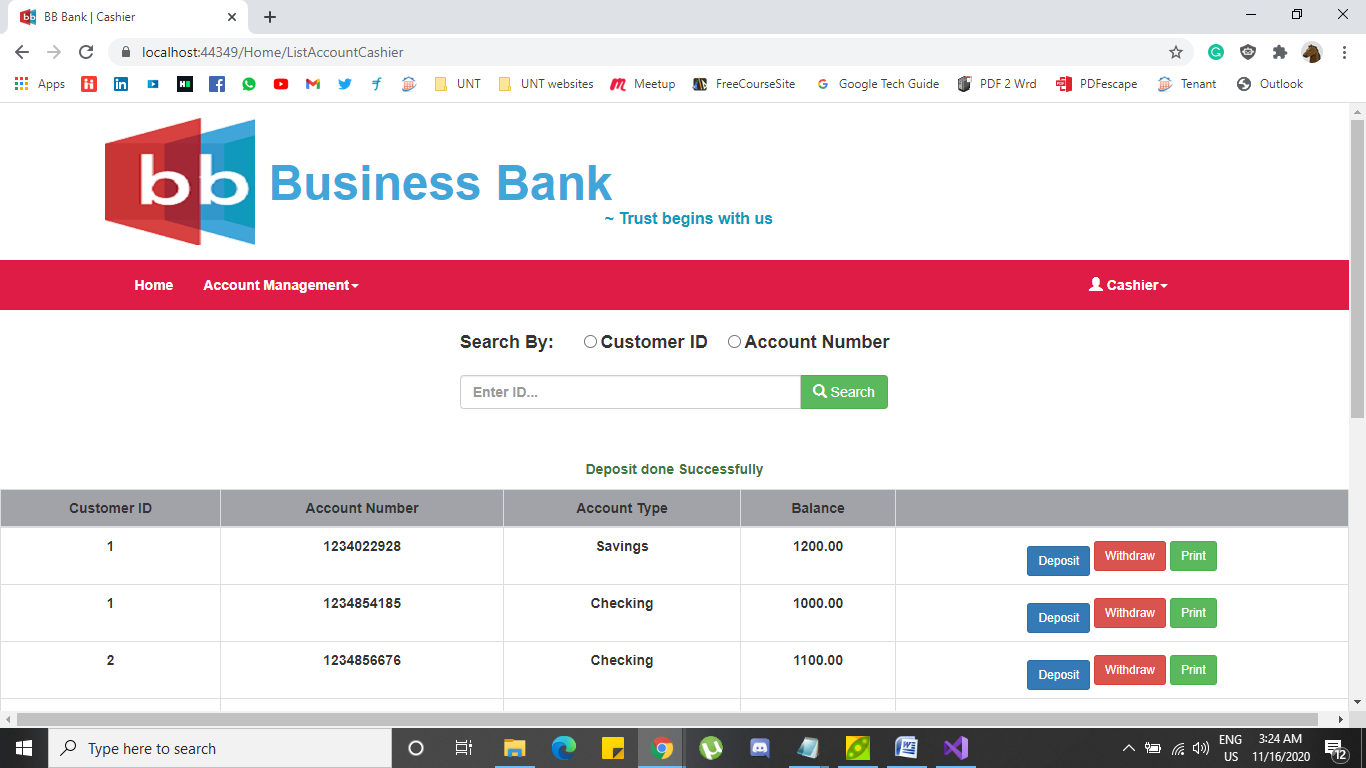
If you want to deposit any certain amount to any account then you can click on Deposit button on the required account that you would like to deposit and then the below page pops out.



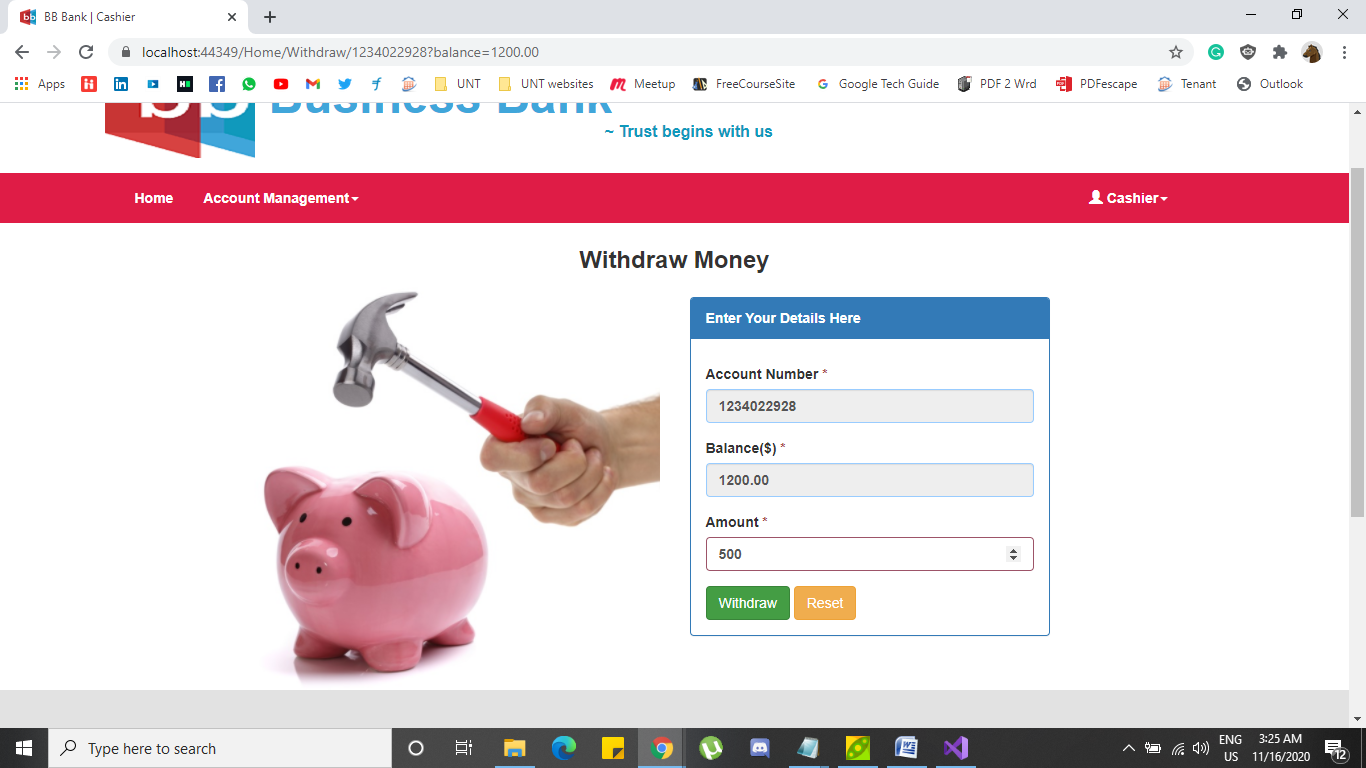
Here you can enter you amount. Amount cannot be null or less than null and then click on Deposit button. If you want to enter new amount you can also click Reset button and enter new amount.



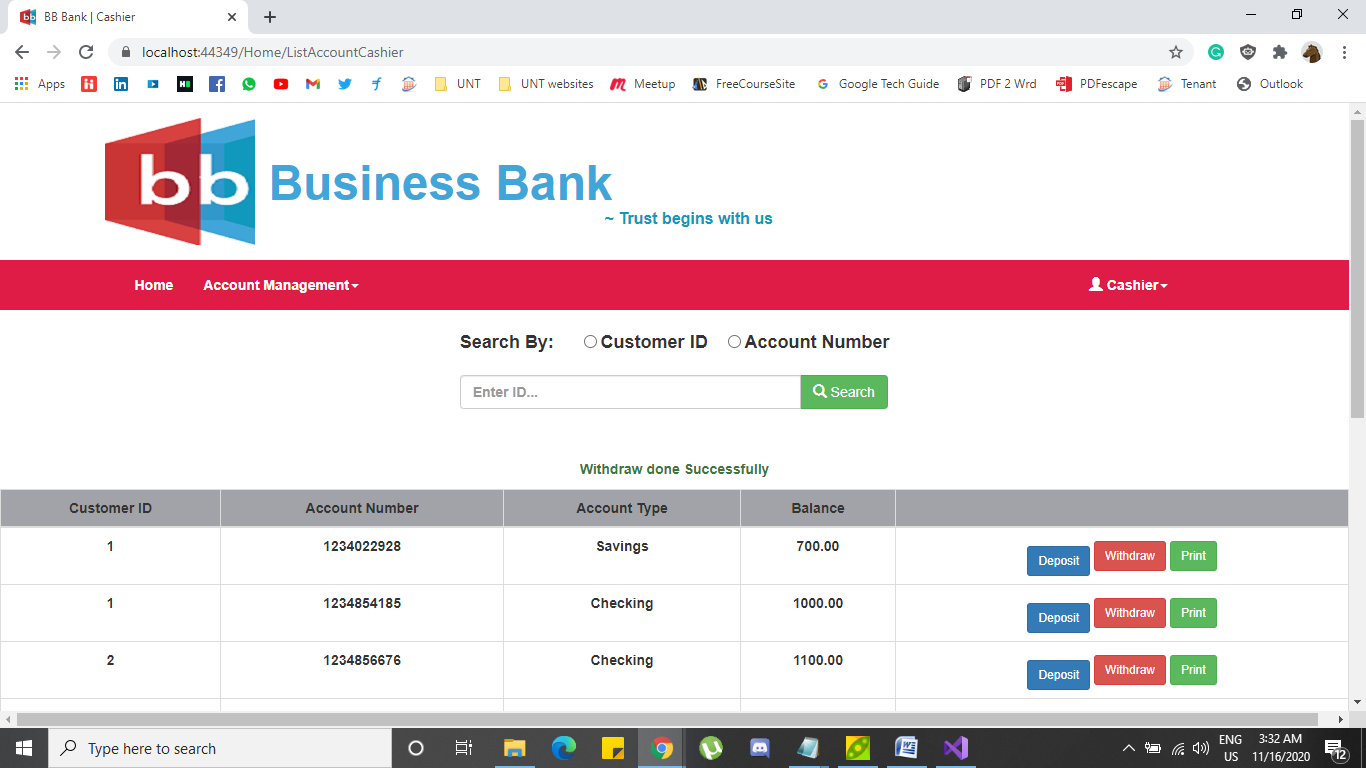
If the amount entered passes all the validations, then the particular amount will be deposited to the required account. The text named “**Deposit done Successfully**” will be shown.



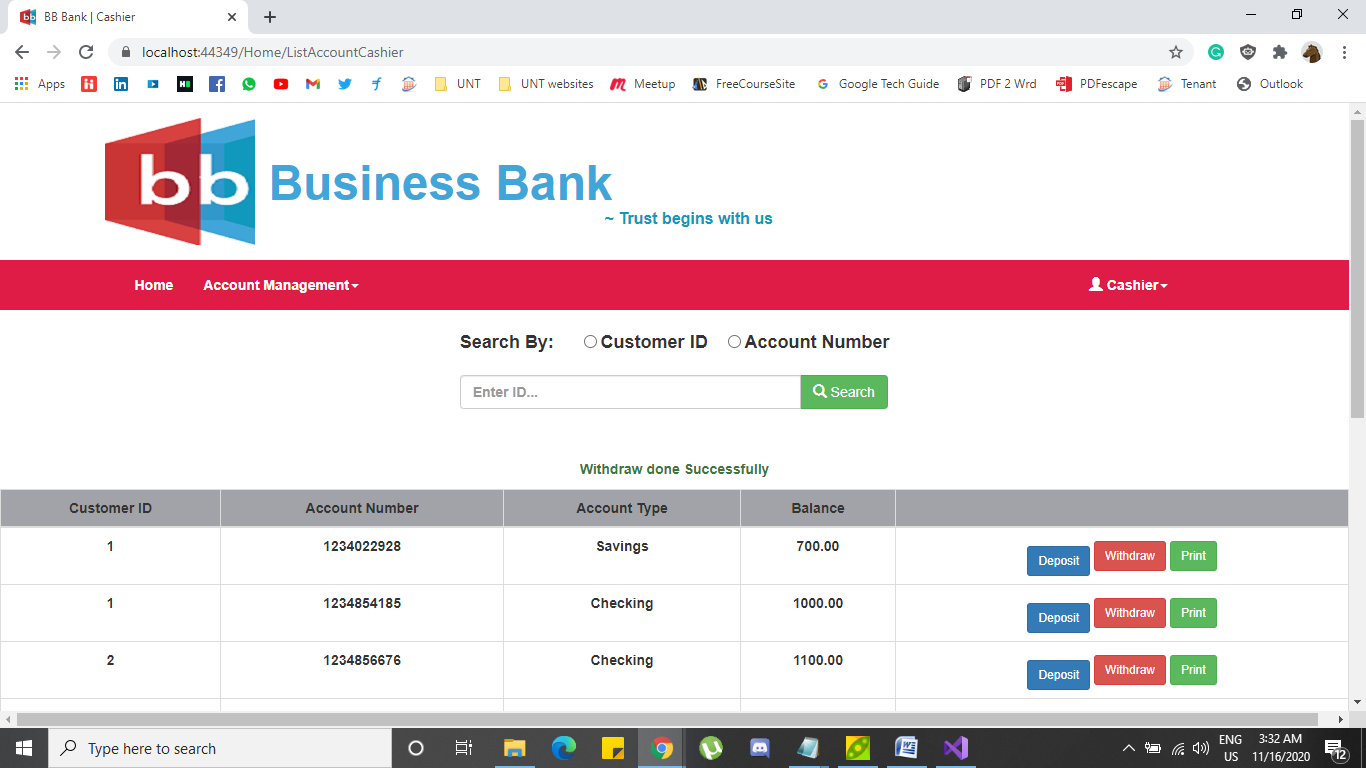
Similarly, if you want to withdraw money from a particular account upon customers’ request. You can click on Withdraw button and then withdraw page pops out.



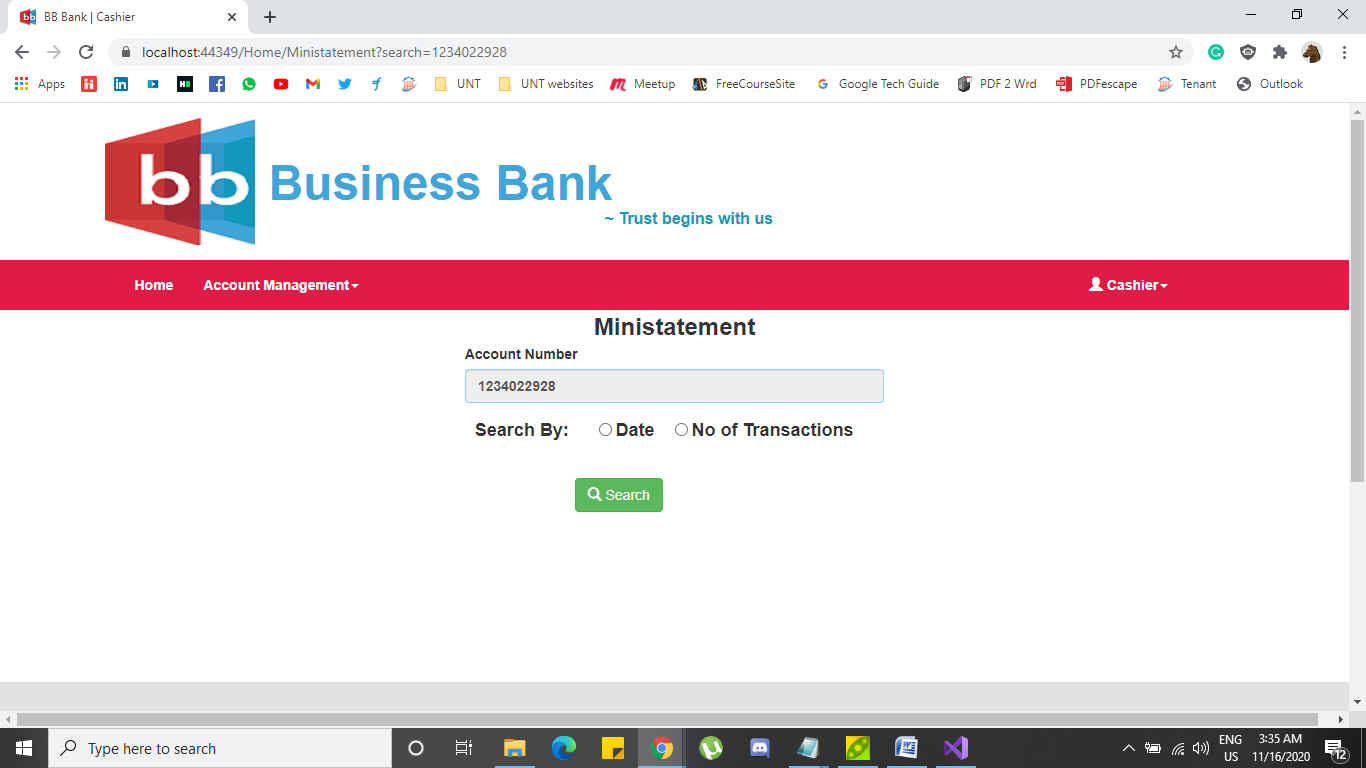
You can enter any amount in the field, the withdraw amount should be no less than the Balance. If the amount satisfies all the validations, then click on Withdraw button and the amount will be withdrawn from the bank.



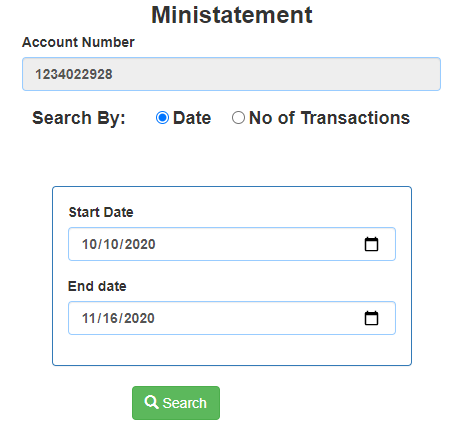
After the amount is withdrawn the message named “**Withdraw done Successfully**” is shown and the page redirects to manage account page.



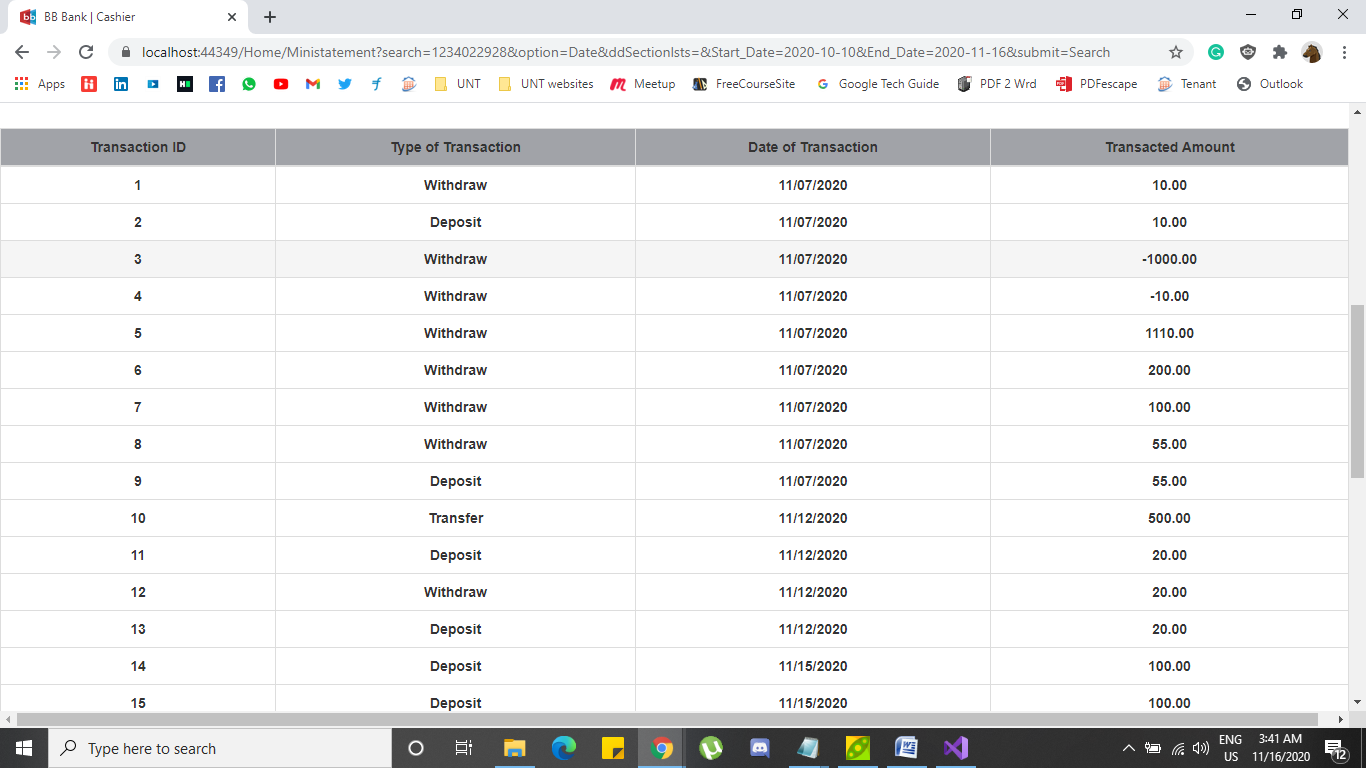
If you click on Print button then the page redirects to “Mini statement” page.



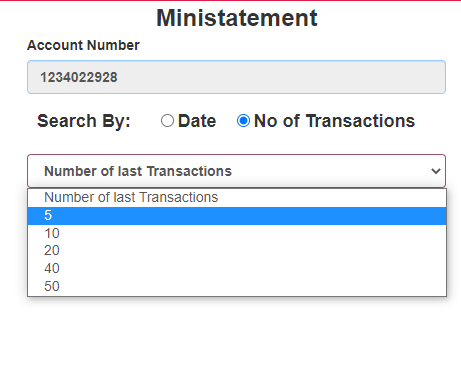
Here you can search the statement by Date, It shows all the transactions done with in the date period.



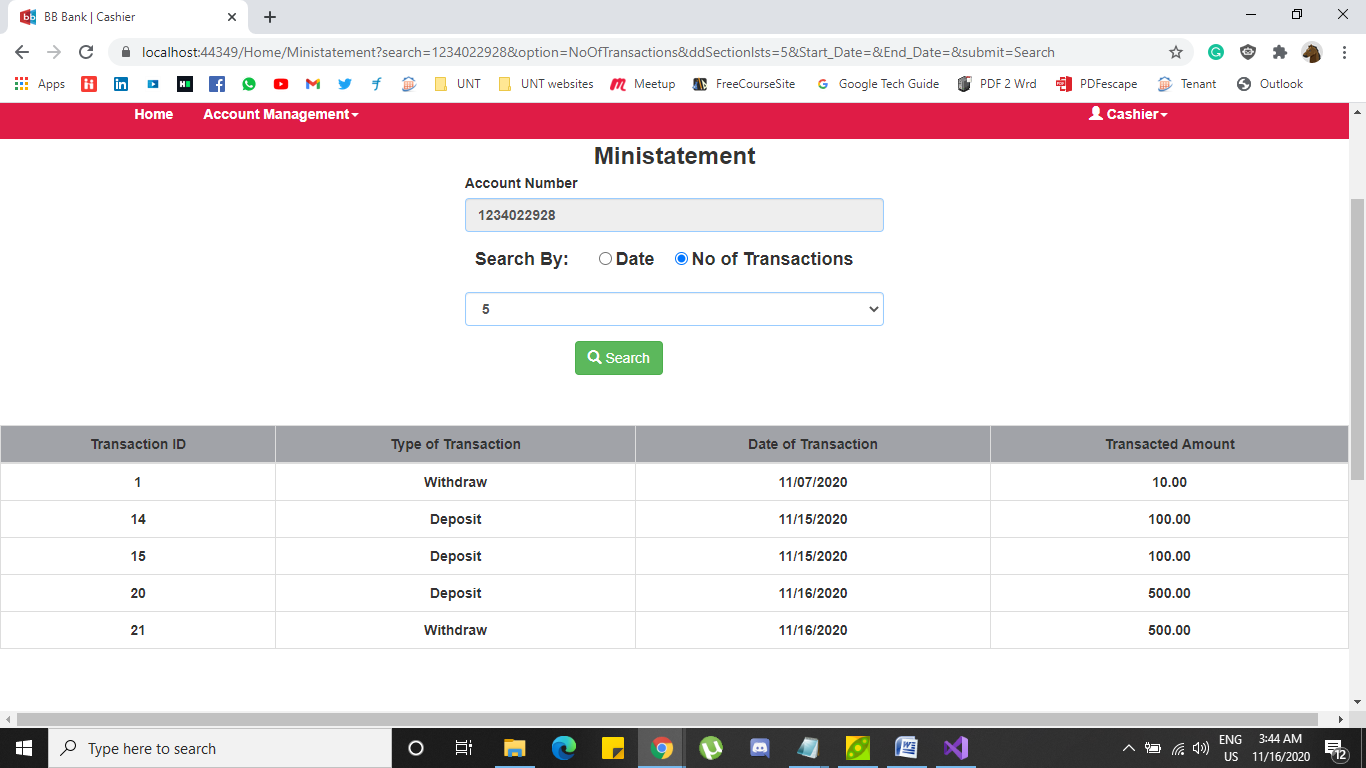
After clicking on search button.



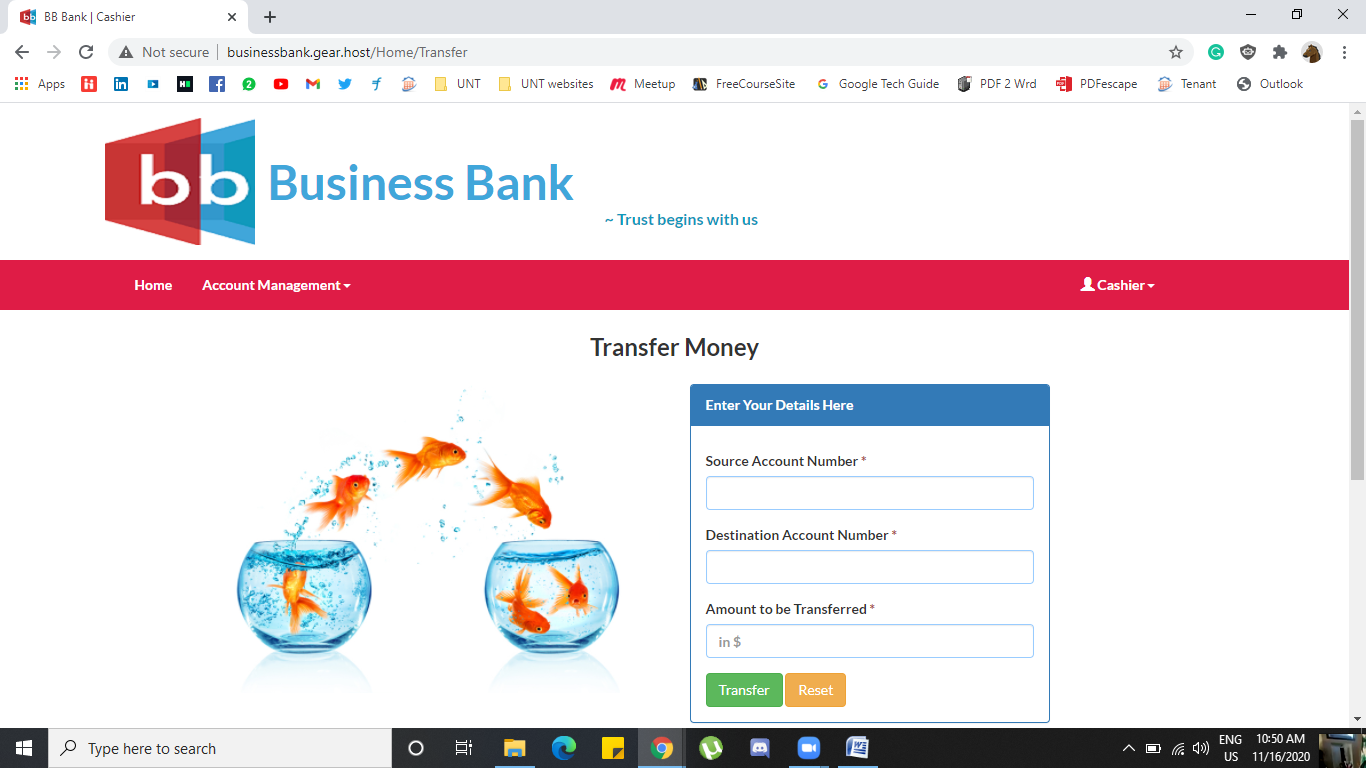
You can also search by No of transactions made. You can select last 5, 10, 20, 40, 50 transactions.



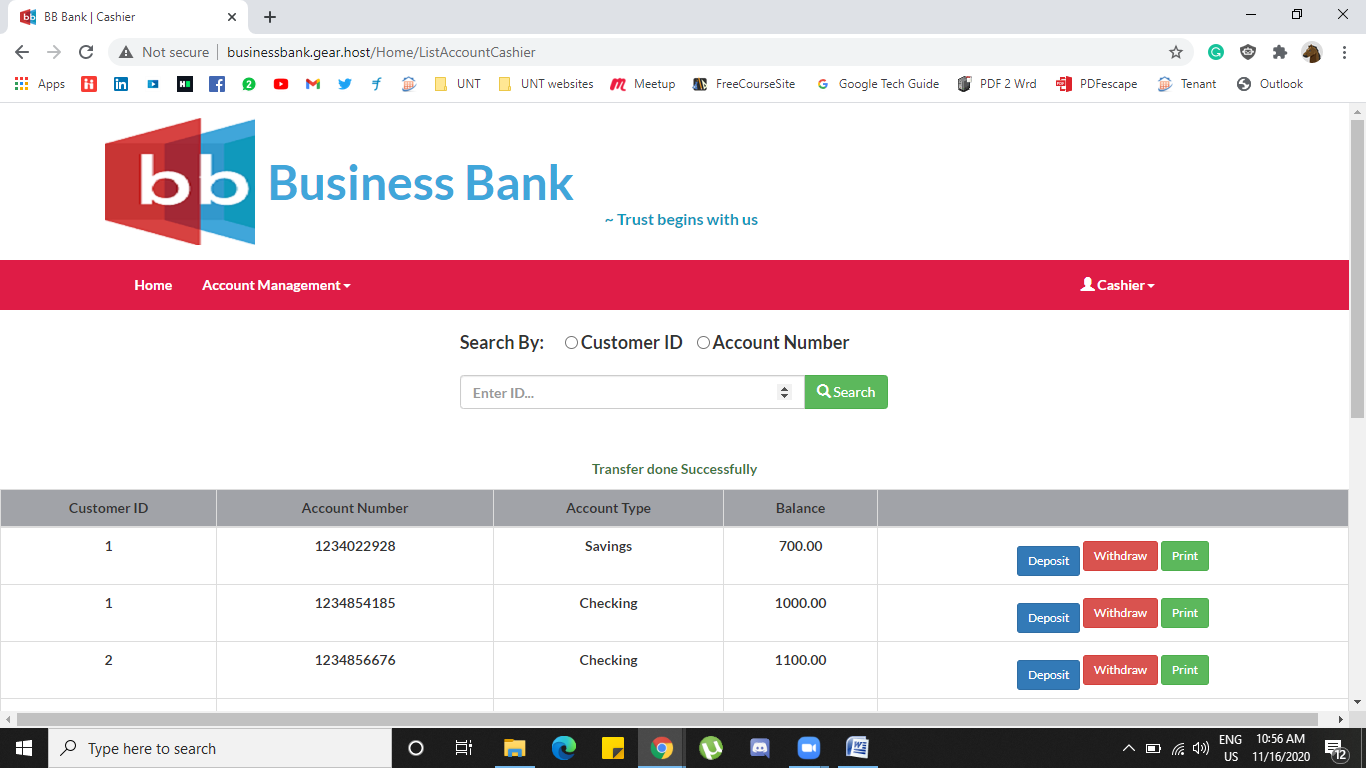
After Clicking Search with last 5 Transactions.



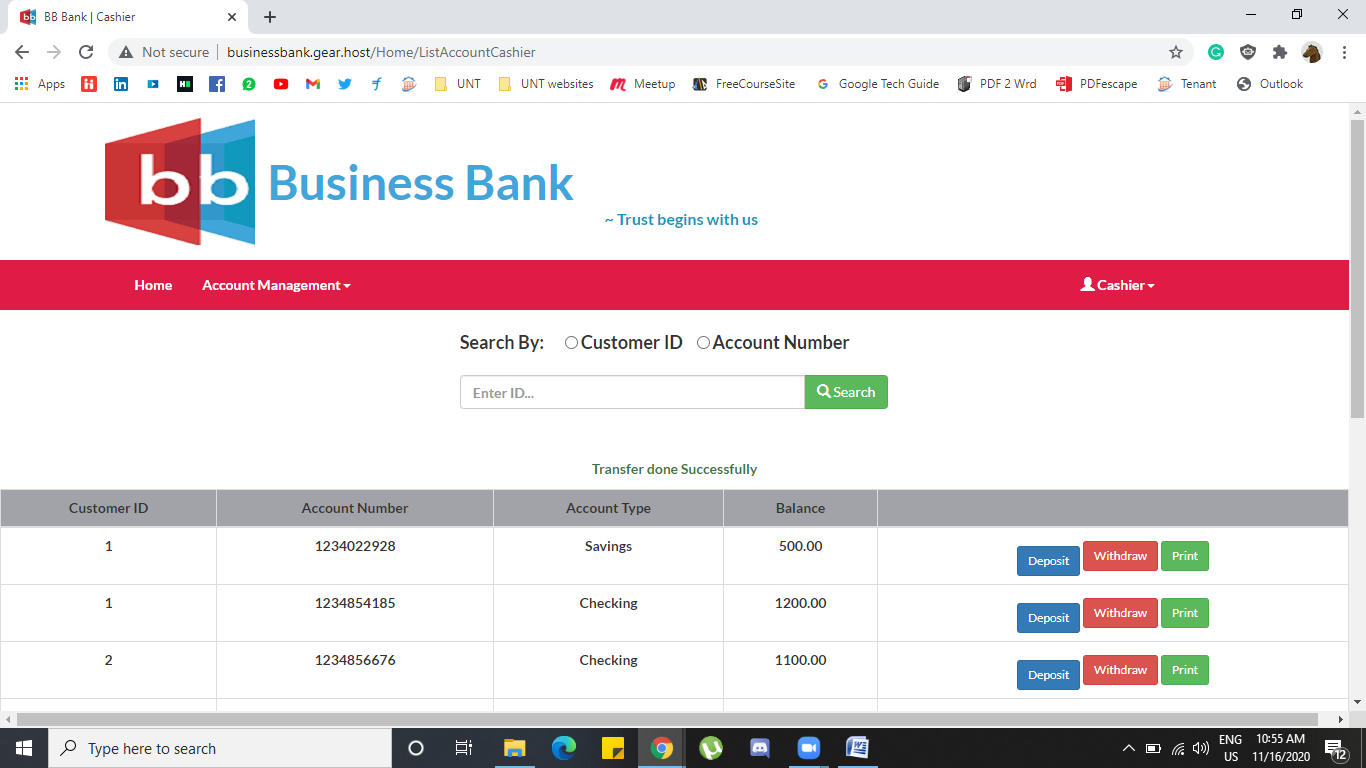
Under Account management dropdown, **Transfer** functionality is selected, and Transfer page pops out and the page looks like



In the transfer pager, in order to make any transaction you will need to have two account numbers typically source and Destination and the amount to the transferred. If it passes all the validations, then the transfer is made.



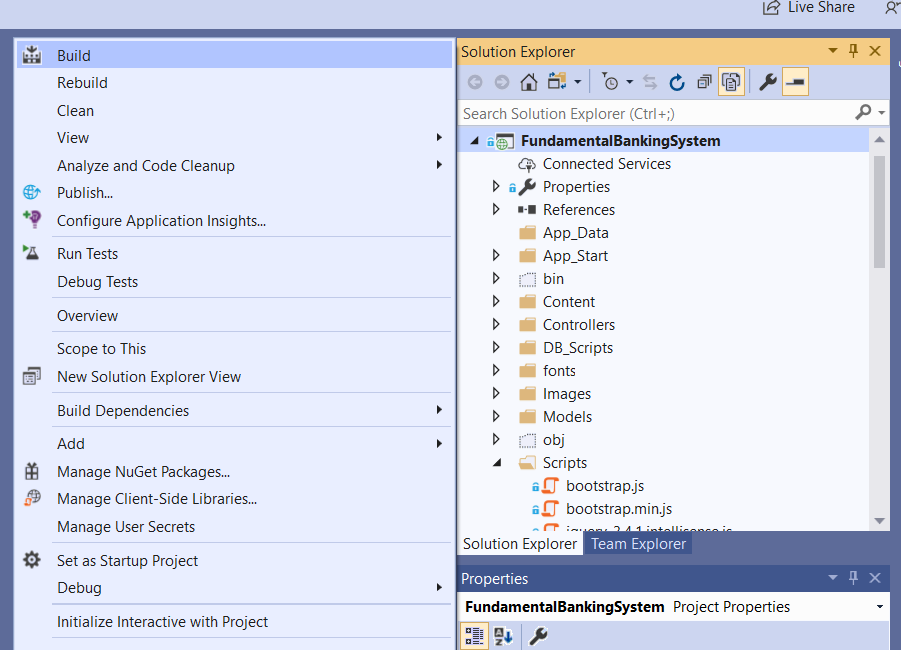
Here the Source and destination numbers are 1234022928 and 1234854185 and the amount is 200 and the transfer is made.



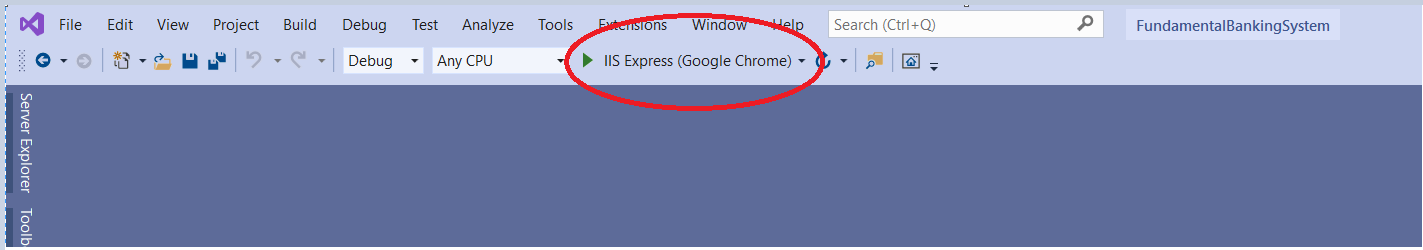
These are all the functionalities designed and this website can perform.

**5. Instructions to compile/run program.**

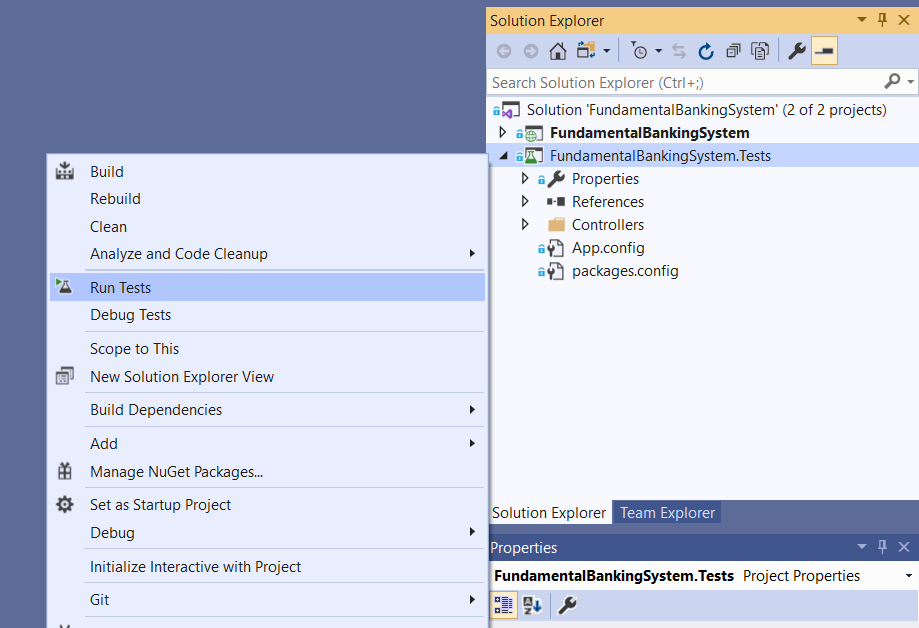
Open the Visual studio 2019 application and clone the project repository. In the solution explorer right click on the “FundamentalBankingSystem” solution and click on clean. After cleaning the solution is successful now right click on the same solution and click on build as shown in figure.



Once the build is successful, click on IIS Express on the top navigation bar in visual studio.



To run the test cases right click on “FundamentalBankingSystem.Tests” and click on build. Once the build is successful right click on same solution and click on Run Tests as shown in figure.



**6. Enhancements**

Below functionalities are successfully implemented in our project:

* Sign In
* Withdraw
* Transfer
* Deposit
* Mini Statement
* List Accounts
* Sign Out
* Add Customer
* Edit Customer
* Delete Customer
* Add Account
* Delete Account

Date of Birth field in Edit Customer functionality is not populating correctly. This is the only limitation that is left incomplete.

Currently our software is used by two users i.e., Cashier and Manager within the bank. In the future enhancements we want to include Loan officer and Auditor users. Where the loan officer can issue the loans, upload the loan papers, and all other related documents and functionalities. Similarly, we want to include Auditor functionalities to perform audit. Thus, increasing the scope of the software to more users in the bank. Additionally, during the peer review sessions with our partner team, they asked whether we included any functionality to take actions on the bank account which has no transactions for two years and more. This functionality can be included in the next phase by including a notification alert system whenever there are such bank accounts that are inactive for more than specific period.