CAPSTONE PROJECT: BANKING SYSTEMS

1. SYSTEM REQUIREMENT SPECIFICATION:

1.1. Software Requirements:

- Operating System: windows 10
- Coding language: Java
- Data Base: MySQL
- Front end: browser (recent version)

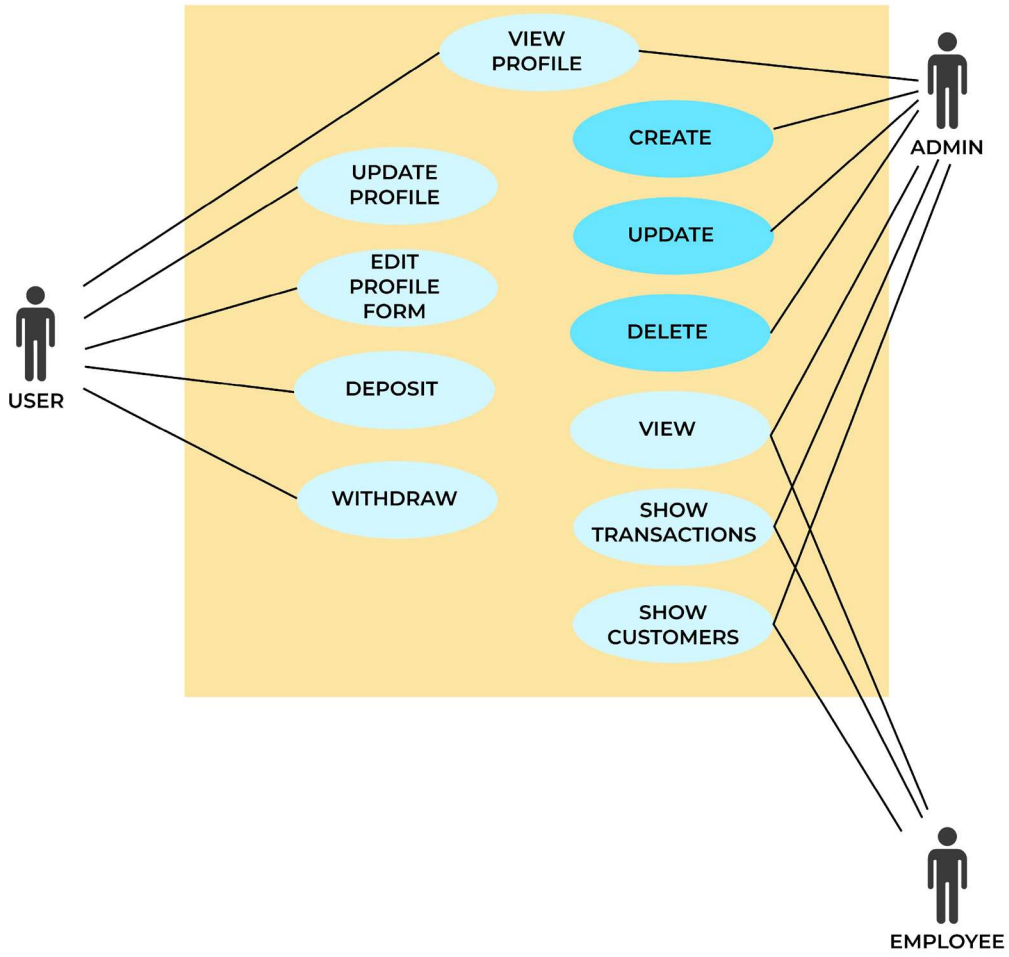
1.2. Hardware Requirements:

- Personal computer with keyboard and mouse maintained with uninterrupted power supply.
- Processor: Intel core™ i5
- Installed Memory (RAM): 8.00 GB

2. PROJECT SCOPE:

- Employees of the bank can use the front-end to view transactional details and perform the necessary actions.
- Front-end for the customers to be able to view their specific transactions and details. They should also be able to interact with the system and perform required transactions.
- The admin can examine all of the details of all customers and employees through the front-end.

3. USE CASE DIAGRAM:



The use case diagram is a behavioral representation of the problem statement, the players involved, and their functionality. There are three different actors in the above diagram and system:

a) User

b) Admin

c) Employee

4. FLOW OF EVENTS:

- **Basic flow**

4.1. Withdraw:

4.1.1. User enters customer Id.

4.1.2. Bank Database validates the user.

4.1.3. On success user can withdraw money.

4.2. Deposit:

4.2.1. User enters customer Id.

4.2.2. Bank Database validates the user.

4.2.3. On success user can deposit money.

4.3. View Profile

4.3.1. User/Admin/Employee enters Id.

4.3.2. Bank Database validates the user.

4.3.3. On success User/Admin/Employee can view profile. (They can also change their own profile detail)

4.4. View Transactions:

4.4.1. Admin/Employee logs in

4.4.2. Bank Database validates the Admin/Employee.

4.4.3. After successful validation, Admin/Employee can view all transaction details.

4.5. Show Customers:

4.5.1. Admin/Employee logs in

4.5.2. Bank Database validates the Admin/Employee.

4.5.3. After successful validation, Admin/Employee can view all customers and their details.

- Alternate Flow

4.6. If the user/admin/employee details are invalid in the basic flow, the user/admin/employee is notified that their login attempt has failed. The user has the option of re-entering correct information or creating a new account.

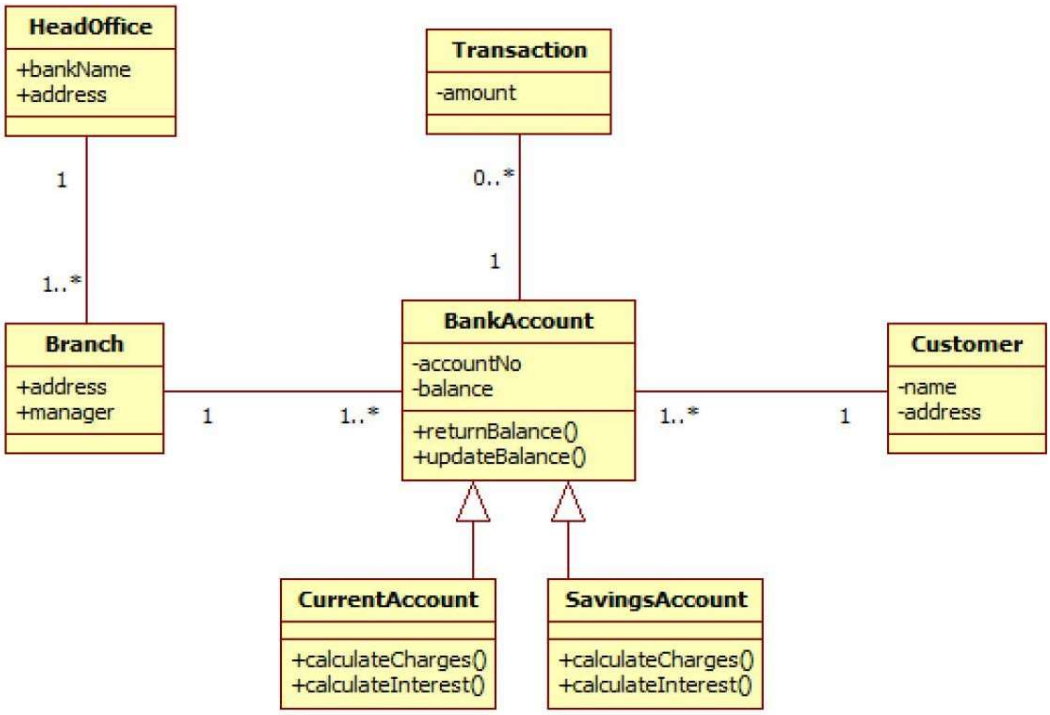
- Pre-conditions:

4.7. The user must have a valid account in the bank.

- Post-conditions:

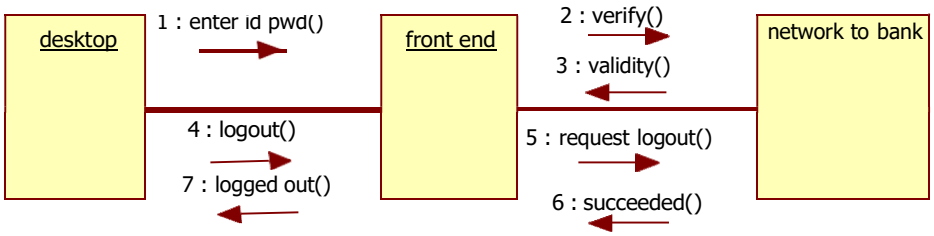
4.8. The database is modified after transaction.

5. CLASS DIAGRAM:

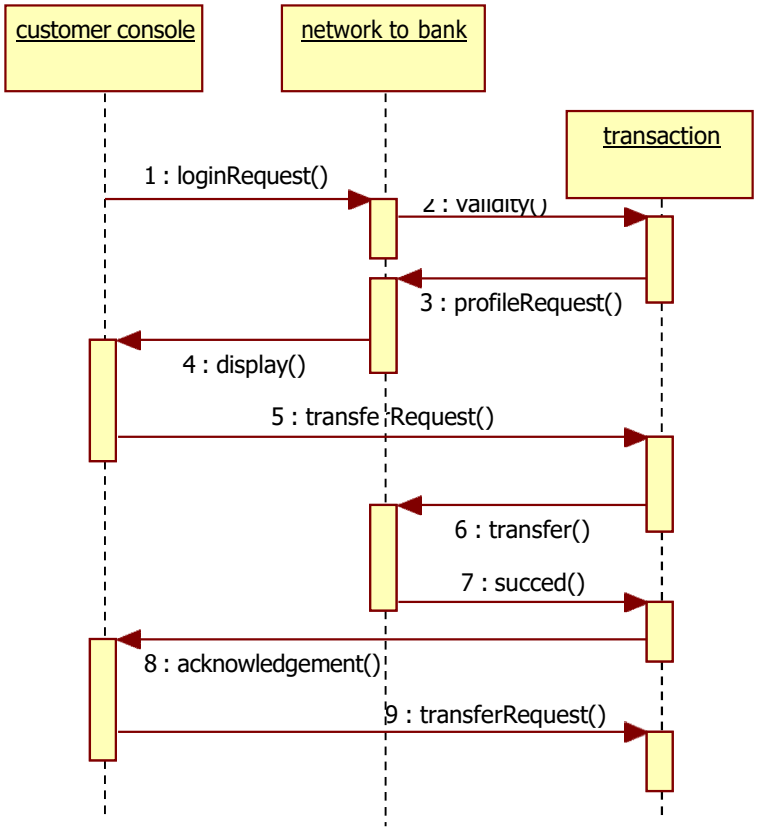


6. SEQUENCE DIAGRAM:

6.1. LOGIN & LOGOUT COLLABORATION DIAGRAM:



6.2. ONLINE TRANSACTION SEQUENCE



7. IDENTIFICATION OF RESPONSIBILITIES OF CLASSES:

7.1. Controllers:

CLASS NAME: AdminController		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
- View admin profile - Create update delete employees and customers	Service layer classes.	Admin, employee, customer models and AdminRepository, EmployeeRepository, CustomerRepository.

CLASS NAME: BankAccountController		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Contains the creation, updation, and deletion code for employees and customers done by the administrator	Service layer classes.	BankAccount, CurretnAccount, SavingsAccount models and BankAccountRepository.

CLASS NAME: BankController		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Create, update, delete, and view details of the bank.	Service layer classes.	Bank model and BankService service and AddressHelper mapper class.

CLASS NAME: BranchController		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES

- C.R.U.D. for Branch - View transactions - View customers	Service layer classes.	AddressHelper mapper class, Branch and Customer models, and BranchService service layer.

CLASS NAME: CustomerController		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Provides functionalities for users to view, update, and edit their profiles.	Service layer classes.	BankAccount, Customer, and Transaction models. BankAccountRepository and CustomerRepository.

CLASS NAME: EmployeeController		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
- CRUD for customer - View employee profile details.	Service layer classes.	Bank, BankAccount, Customer, Employee, Transaction models. BankAccountRepository, CustomerRepository, and EmployeeRepository.

CLASS NAME: HomeController		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
-Shows welcome screen.	Service layer classes.	

CLASS NAME: TransactionController		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Contains the deposit and withdraw functionalities for the user and the show all transactions for administrator and employees.	Service layer classes.	Transaction model and Transaction services.

7.2. DAO:

CLASS NAME: AdminDAO		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Provides abstract interface for implementation on AdminRepository		Admin, Customer and Transaction models.

CLASS NAME: BankAccountDAO		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES

Provides abstract interface for implementation on BankAccountRepository		BankAccount model

CLASS NAME: BankDAO		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Provides abstract interface for implementation on BankRepository		Bank model

CLASS NAME: BranchDAO		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Provides abstract interface for implementation on BranchRepository		Branch, Customer, and Transaction models

CLASS NAME: CustomerDAO		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Provides abstract interface for implementation on CustomerRepository		Customer and Transaction models.

CLASS NAME: EmployeeDAO		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Provides abstract interface for implementation on EmployeeRepository		Bank, Employee, and Transaction models.

CLASS NAME: TransactionDAO		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Provides abstract interface for implementation on TransactionRepository		Transaction model.

7.3. Mappers:

CLASS NAME: AddressHelper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Mapping of each row of Address to a result object.	Database.	Address model.

CLASS NAME: AdminMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Mapping of each row of Admin to a result object.	Database.	Admin model.

CLASS NAME: BankMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Mapping of each row of Bank to a result object.	Database.	Bank model.

CLASS NAME: BranchMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Mapping of each row of Branch to a result object.	Database.	Branch, Customer, and Transaction model.

CLASS NAME: CurrentAccountMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES

Mapping of each row of CurrentAccount to a result object.	Database.	BankAccount and CurrentAccount models.

CLASS NAME: CustomerMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Mapping of each row of Customer to a result object.	Database.	Customer model.

CLASS NAME: EmployeeMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Mapping of each row of Employee to a result object.	Database.	Employee model.

CLASS NAME: ReversalRequestsMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES

Mapping of each row of ReversalRequests to a result object.	Database.	ReversalRequests model.

CLASS NAME: SavingsAccountMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Mapping of each row of SavingsAccount to a result object.	Database.	BankAccount and SavingsAccount models.

CLASS NAME: TransactionMapper		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Mapping of each row of Transaction to a result object.	Database.	Transaction and TransactionType model.

7.4. Models:

CLASS NAME: AccountType

Enumerator for account type.		

CLASS NAME: Address		
For setting the address and displaying it.		

CLASS NAME: Admin		
For setting and displaying Admin details.		

CLASS NAME: Bank		
For setting and displaying bank details such as code, name, address, etc.		

CLASS NAME: BankAccount		
For setting and displaying bank account details of a customer like account number, customer id, branch, available balance, etc.		

CLASS NAME: Branch		
For setting and displaying branch details like branch code, bank code, manager, etc.		

CLASS NAME: CurrentAccount		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Calculates charge and interest and returns account balance of current account.		BankAccount model.

CLASS NAME: Customer

For setting and displaying customer details like name, address, customer id, etc.		

CLASS NAME: Employee		
For setting and displaying employee details like id, branch code, name, etc.		

CLASS NAME: ReversalRequests		
For setting and displaying date, time, request id of reversal requests.		

CLASS NAME: SavingsAccount		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
Calculates charge and interest and returns account balance of savings account.		Extends BankAccount model

CLASS NAME: Transaction		
For setting and displaying transaction details like timestamp, amount, transaction number, etc.		

CLASS NAME: TransactionType		
Enumerator for account type.		

7.5. Repository:

CLASS NAME: AdminRepository		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For getting Admin by ID, showing Transactions and showing customers.		Admin, Customer and Transaction Mapper. Admin, Customer and Transaction models. Admin DAO.

CLASS NAME: BankAccountRepository		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES

For creating and showing accounts, getting all accounts by Branch code, Account number and ID and deleting a bank account.		SavingAccount Mapper. BankAccount models. BankAccount DAO.

CLASS NAME: BankRepository		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For creating Bank and viewing details by IFSC.		Bank Mapper. Bank models. Bank DAO.

CLASS NAME: BranchRepository		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For creating branch, viewing details by IFSC, showing Transactions, customers and branches, deleting by IFSC and updating Branch manager.		Branch, Customer and Transaction Mappers. Branch, Customer and Transaction Models. Branch DAO.

CLASS NAME: CustomerRepository		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES

For creating Customer, deleting customer by ID, getting customer by Account Number, by Branch code or by ID, updating Customer by ID, showing Transactions TypeWise and getting Customer by branches.		Customer and Transaction Mappers. Customer and Transaction models. Customer DAO

CLASS NAME: EmployeeRepository		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For creating and deleting employees, getting employee bank, getting employee by ID, showing all employees, updating Employee and getting employee transactions.		Bank, Employee and Transaction Mappers. Bank, Employee and Transaction models. Employee DAO.

CLASS NAME: TransactionRepository		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For depositing, showing transactions, withdrawing, depositing in transactions and crediting in transactions		Transaction Mapper. Transaction Model. Transaction DAO.

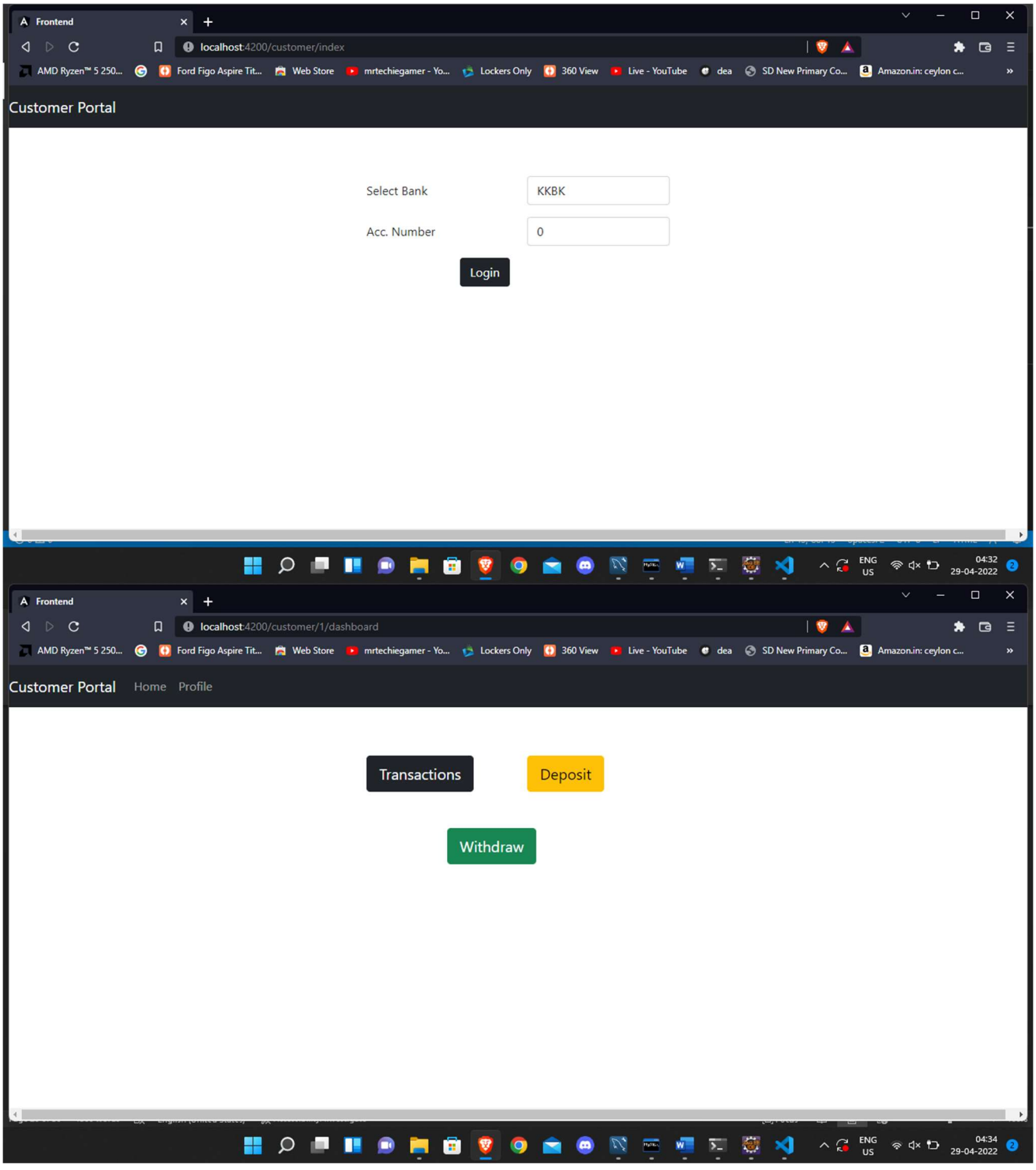
7.6. Services:

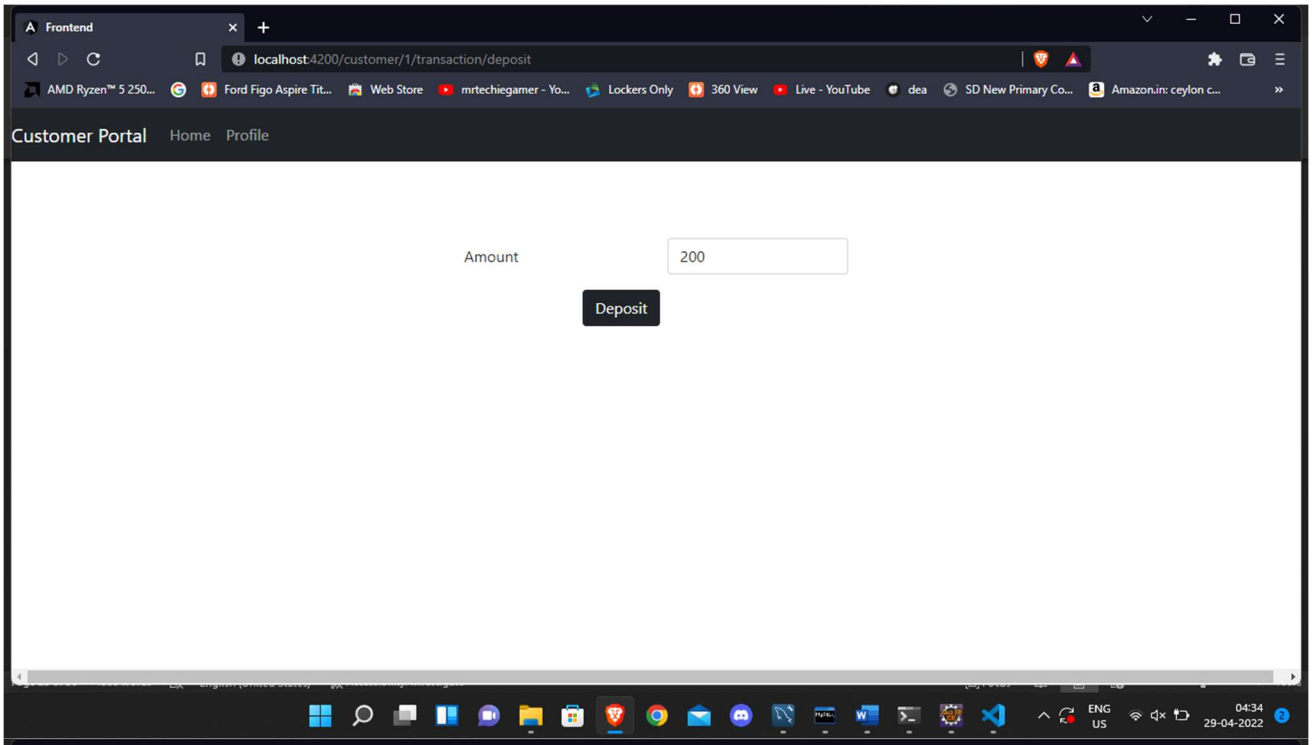
CLASS NAME: BankService		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For Creating Bank and Viewing The details.		Bank model. Bank Repository.

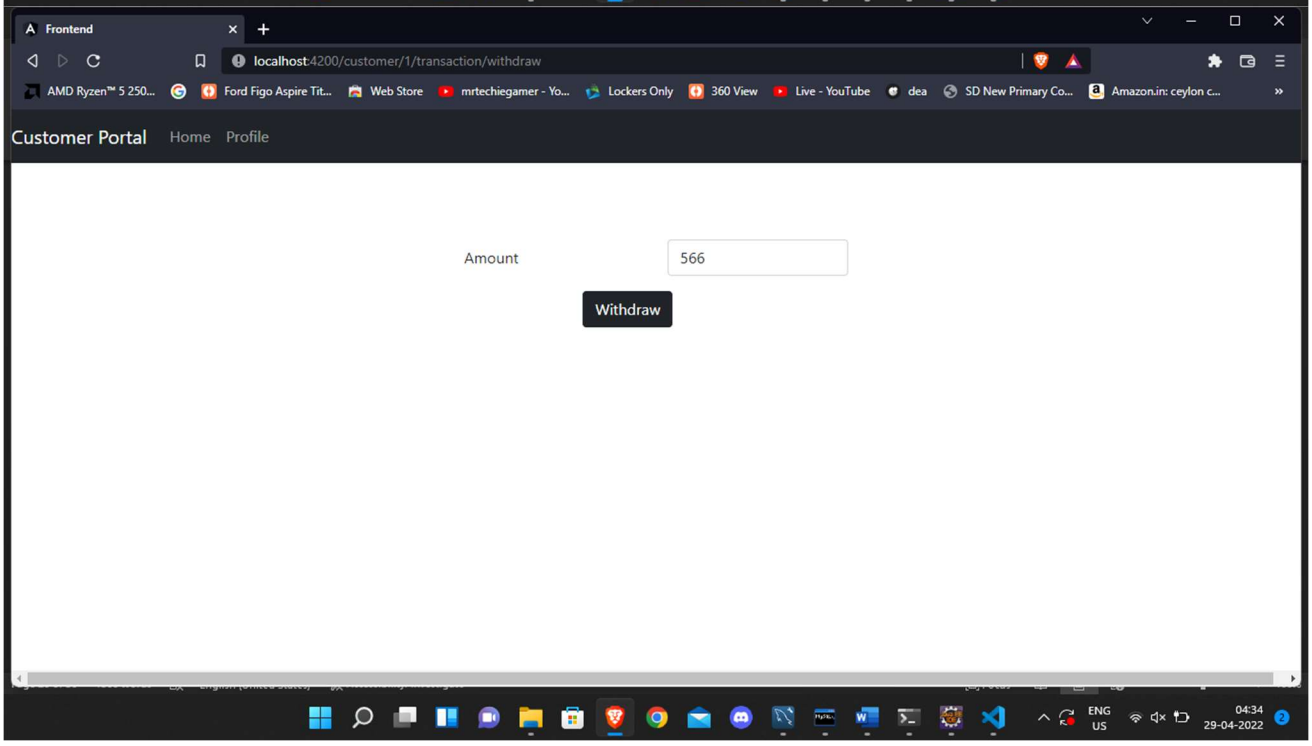
CLASS NAME: BranchService		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For creating Branch, get the branches, get the branches by IFSC, deleting Branch by IFSC, updating Branch manager name and getting Customer by Branches.		Branch and Customer models. Branch and Customer Repository.

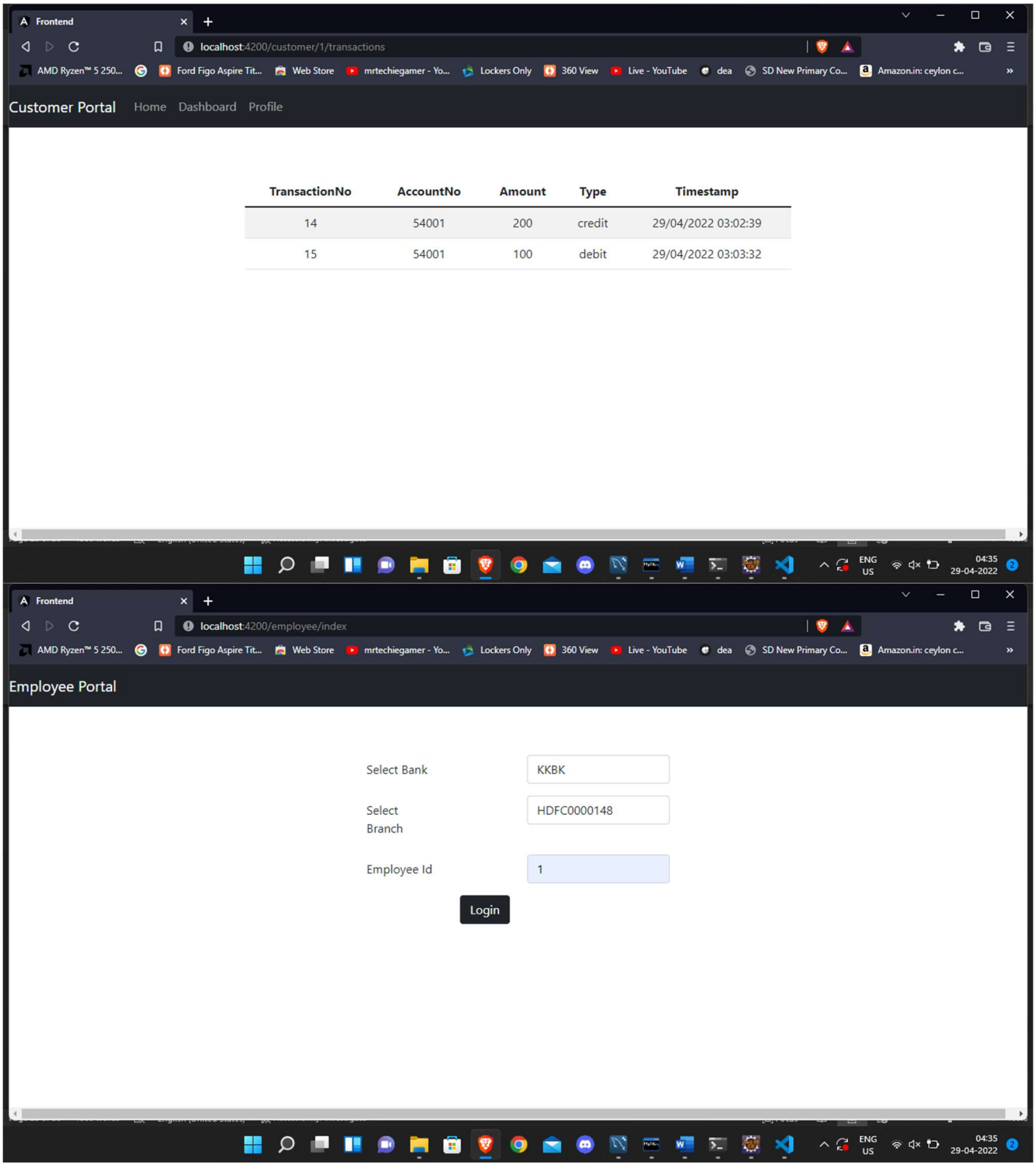
CLASS NAME: TransactionService		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For depositing, withdrawing and getting all the transactions.		Transaction Model. Transaction Repository.

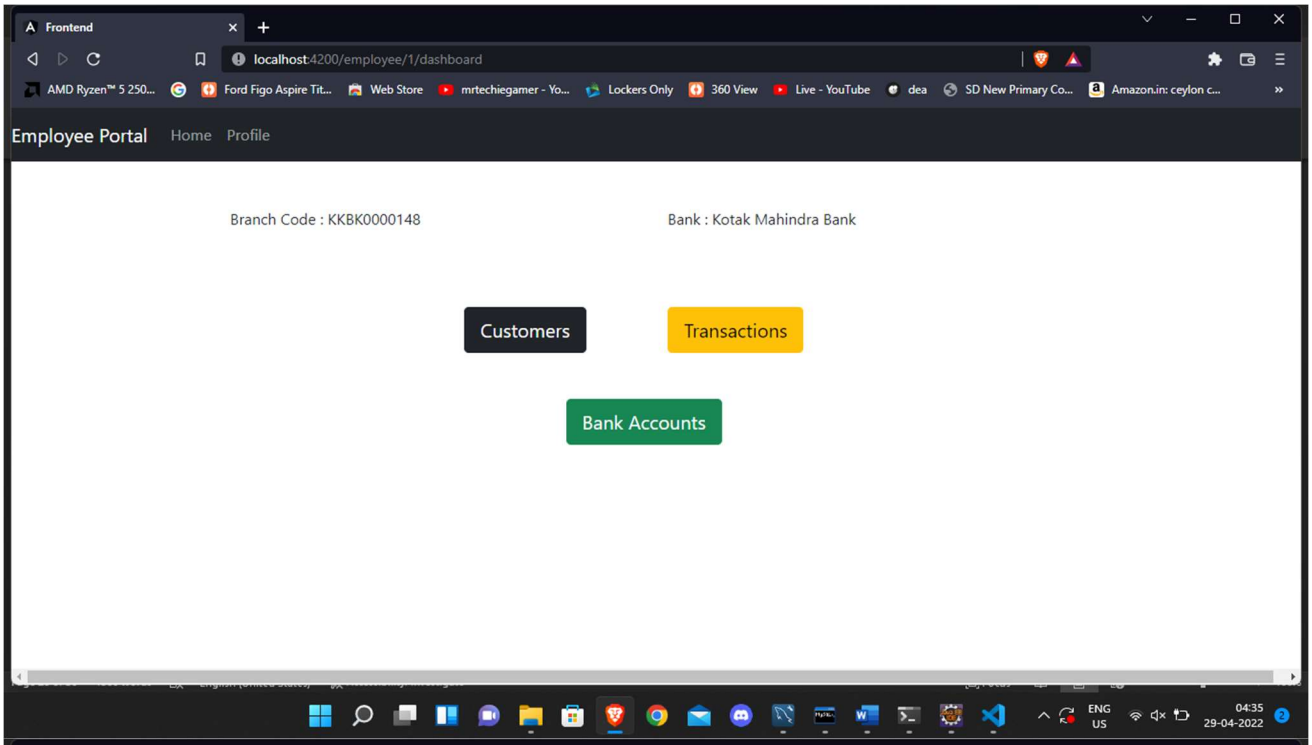
8. IMPLEMENTATION:

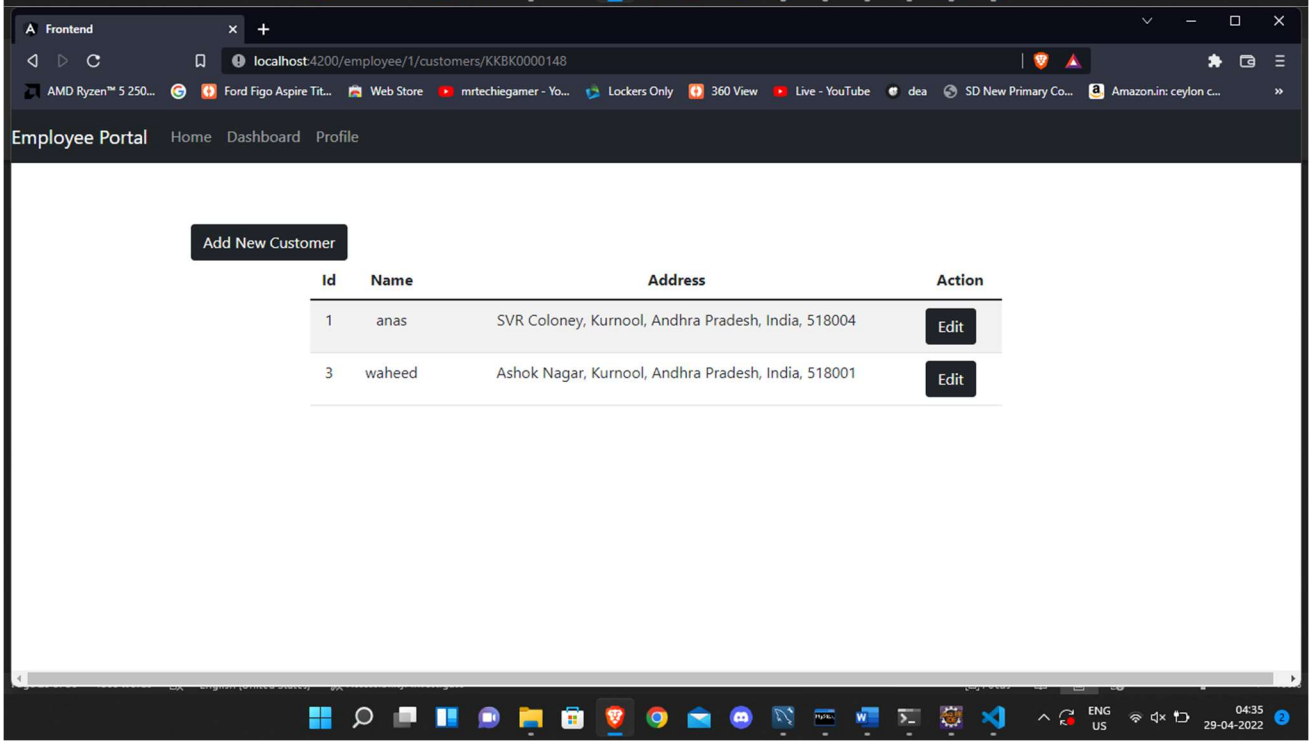


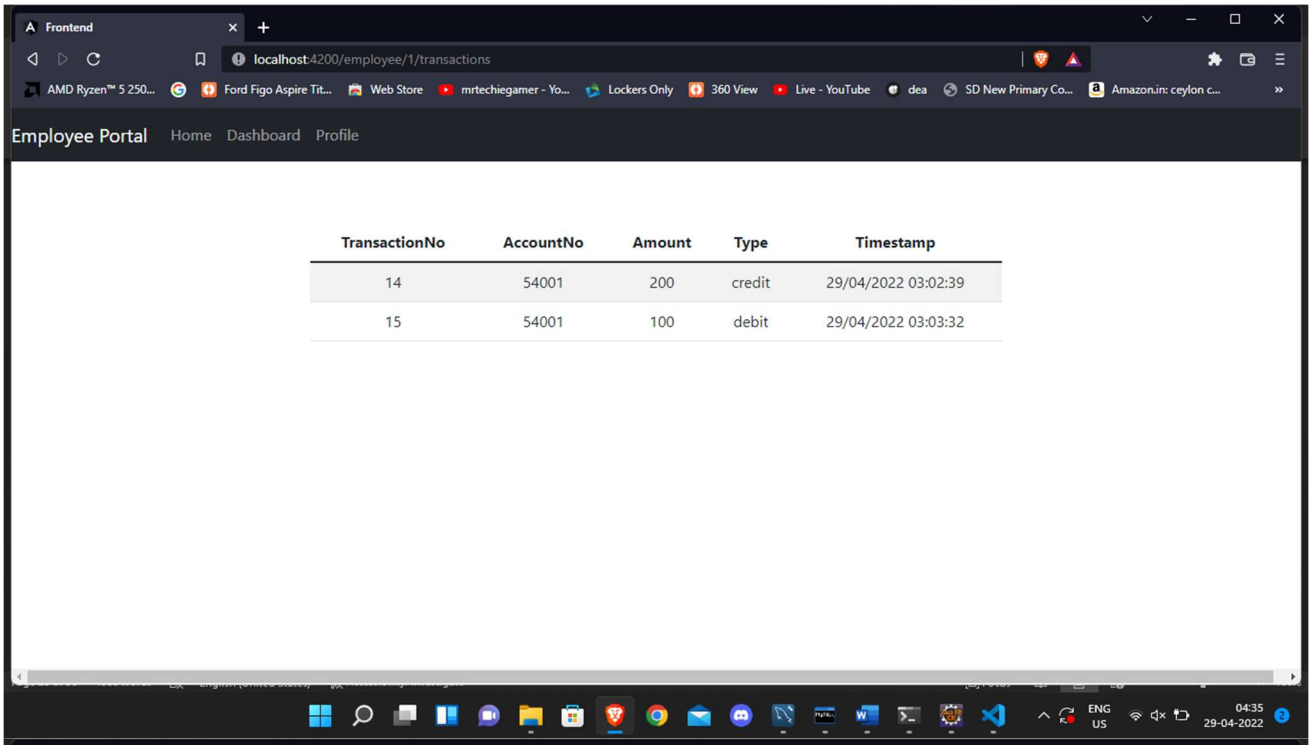

Frontend
localhost:4200/customer/1/transaction/deposit
AMD Ryzen™ 5 250...
Ford Figo Aspire Tit...
Web Store
mrtechiegamer - Yo...
Lockers Only
360 View
Live - YouTube
dea
SD New Primary Co...
Amazon.in: ceylon c...
Customer Portal
Home
Profile
Amount
200
Deposit
ENG
US
04:34
29-04-2022

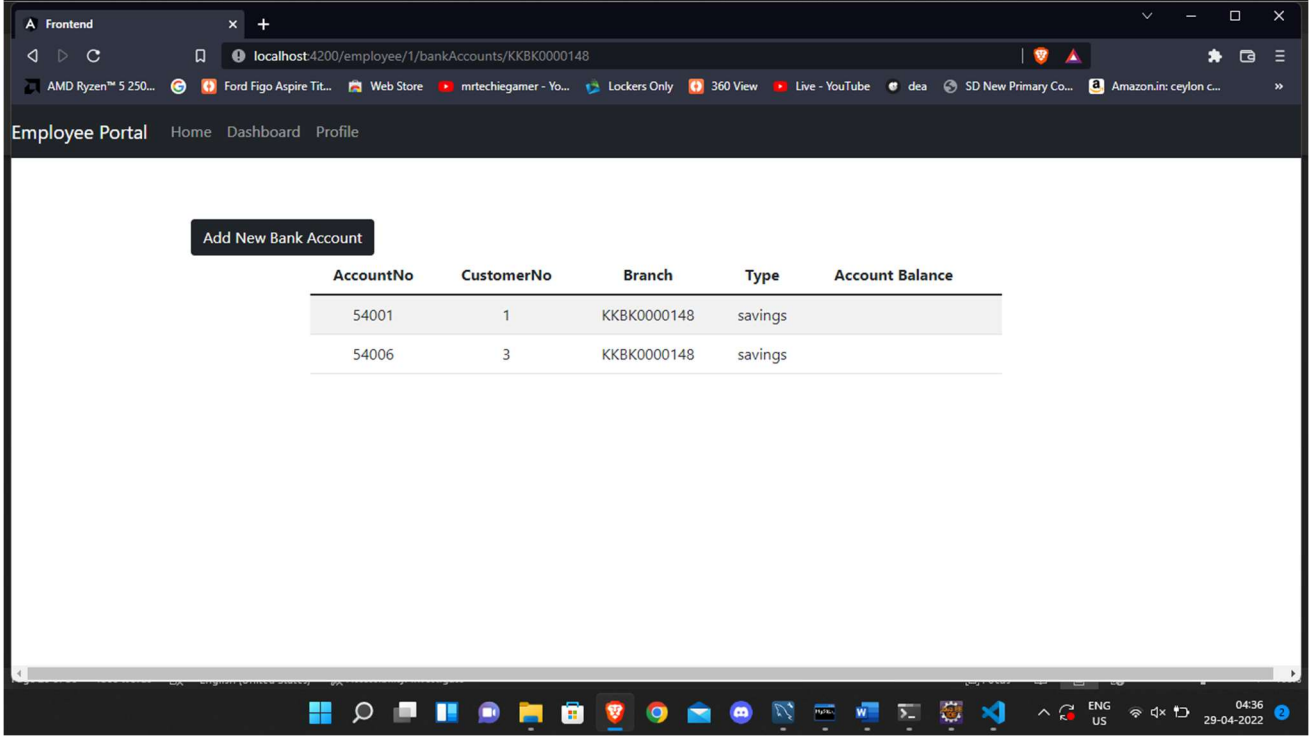

Frontend
localhost:4200/customer/1/transaction/withdraw
AMD Ryzen™ 5 250...
Ford Figo Aspire Tit...
Web Store
mrtechiegamer - Yo...
Lockers Only
360 View
Live - YouTube
dea
SD New Primary Co...
Amazon.in: ceylon c...
Customer Portal
Home
Profile
Amount
566
Withdraw
ENG
US
04:34
29-04-2022

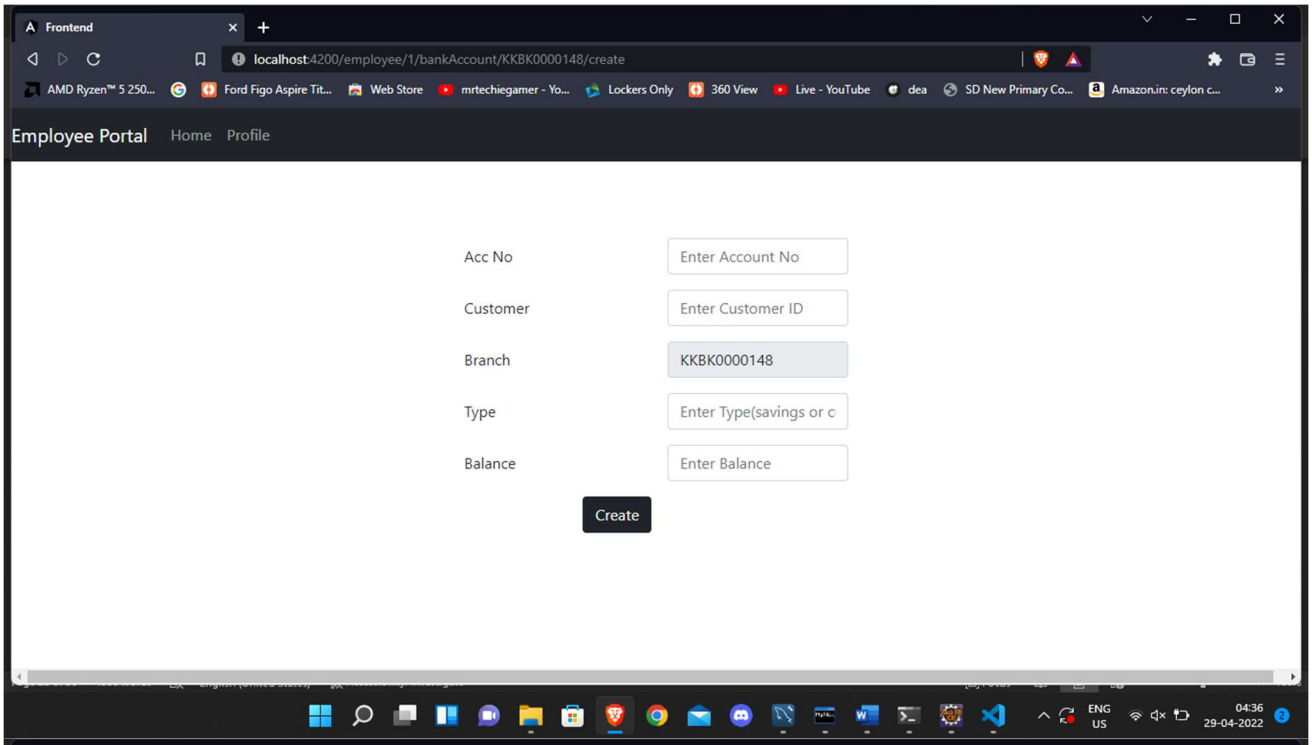

Frontend
localhost:4200/customer/1/transactions
AMD Ryzen™ 5 250...
Ford Figo Aspire Tit...
Web Store
mrtechiegamer - Yo...
Lockers Only
360 View
Live - YouTube
dea
SD New Primary Co...
Amazon.in: ceylon c...
Customer Portal
Home
Dashboard
Profile
TransactionNo
AccountNo
Amount
Type
Timestamp
14
54001
200
credit
29/04/2022 03:02:39
15
54001
100
debit
29/04/2022 03:03:32
ENG
US
04:35
29-04-2022
Frontend
localhost:4200/employee/index
AMD Ryzen™ 5 250...
Ford Figo Aspire Tit...
Web Store
mrtechiegamer - Yo...
Lockers Only
360 View
Live - YouTube
dea
SD New Primary Co...
Amazon.in: ceylon c...
Employee Portal
Select Bank
KKBK
Select Branch
HDFC0000148
Employee Id
1
Login
ENG
US
04:35
29-04-2022

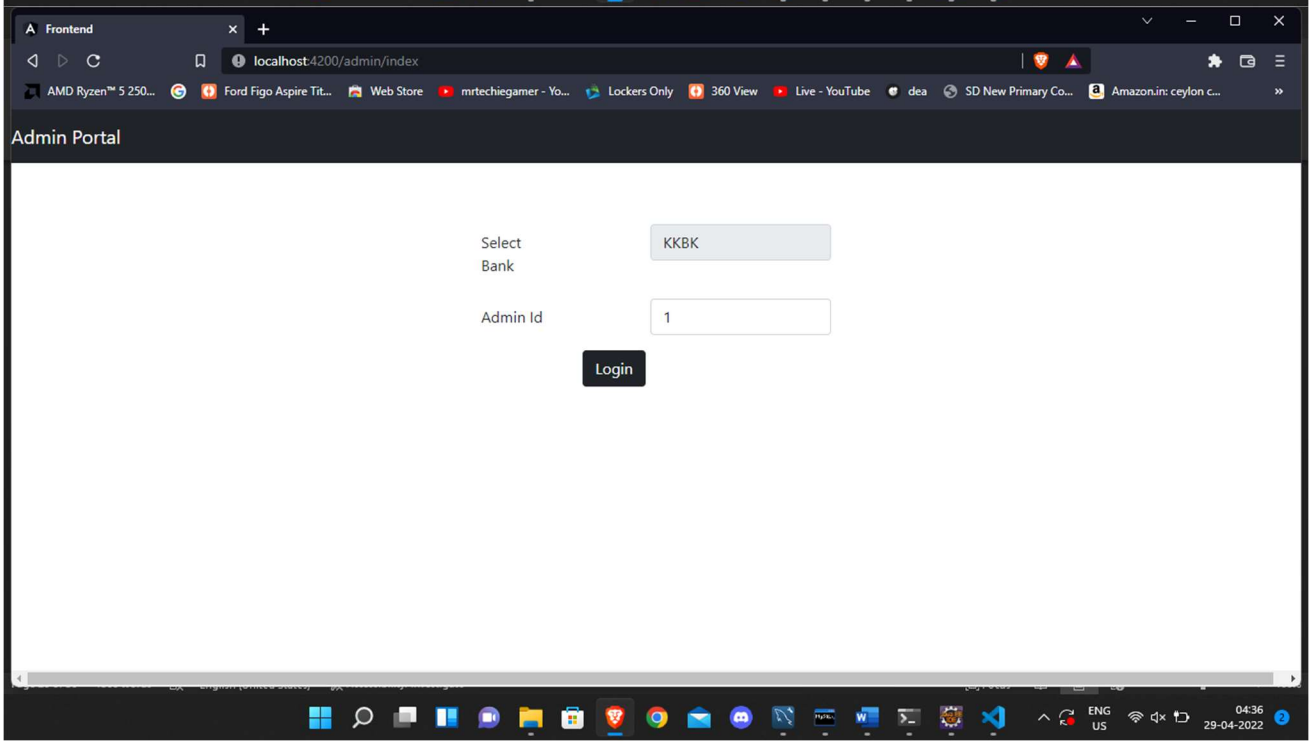

Frontend
localhost:4200/employee/1/dashboard
Employee Portal
Home
Profile
Branch Code : KKBK0000148
Bank : Kotak Mahindra Bank
Customers
Transactions
Bank Accounts

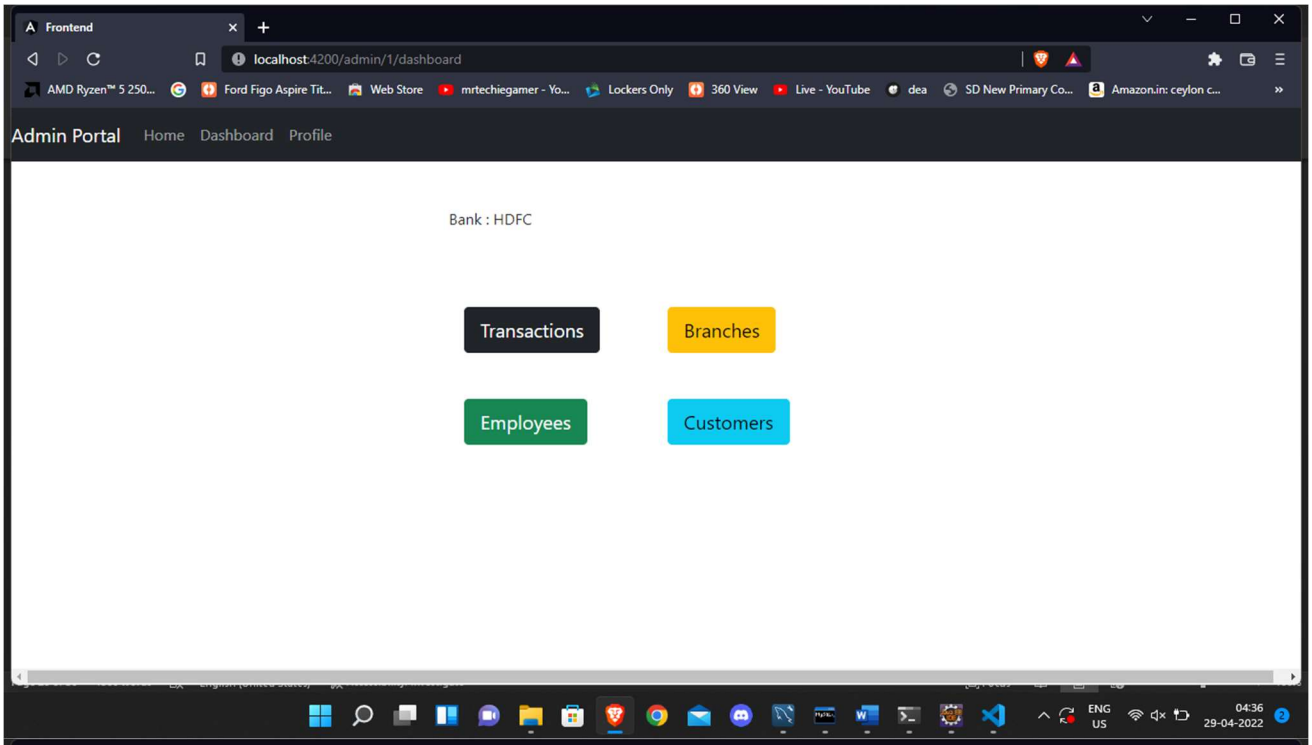

Frontend
localhost:4200/employee/1/customers/KKBK0000148
Employee Portal
Home
Dashboard
Profile
Add New Customer
Id
Name
Address
Action
1
anas
SVR Coloney, Kurnool, Andhra Pradesh, India, 518004
Edit
3
waheed
Ashok Nagar, Kurnool, Andhra Pradesh, India, 518001
Edit

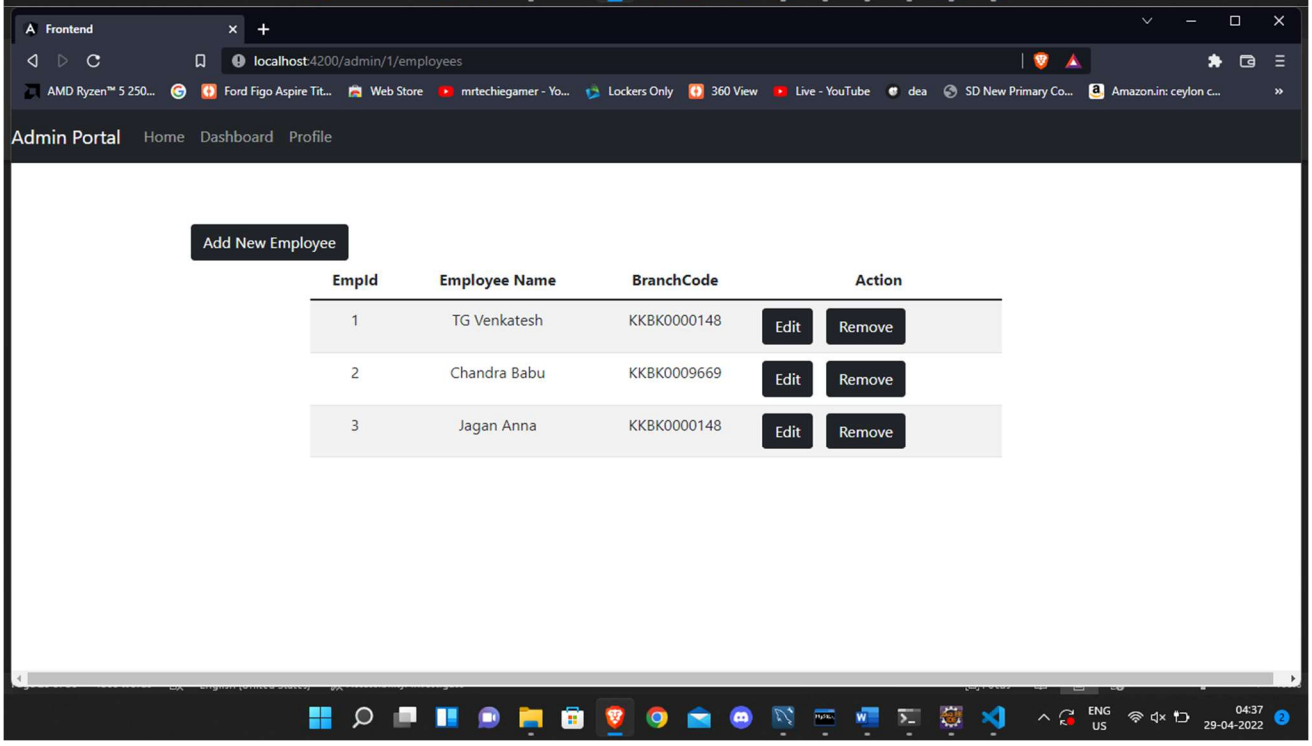

Frontend
localhost:4200/employee/1/transactions
Employee Portal
Home
Dashboard
Profile
TransactionNo
AccountNo
Amount
Type
Timestamp
14
54001
200
credit
29/04/2022 03:02:39
15
54001
100
debit
29/04/2022 03:03:32

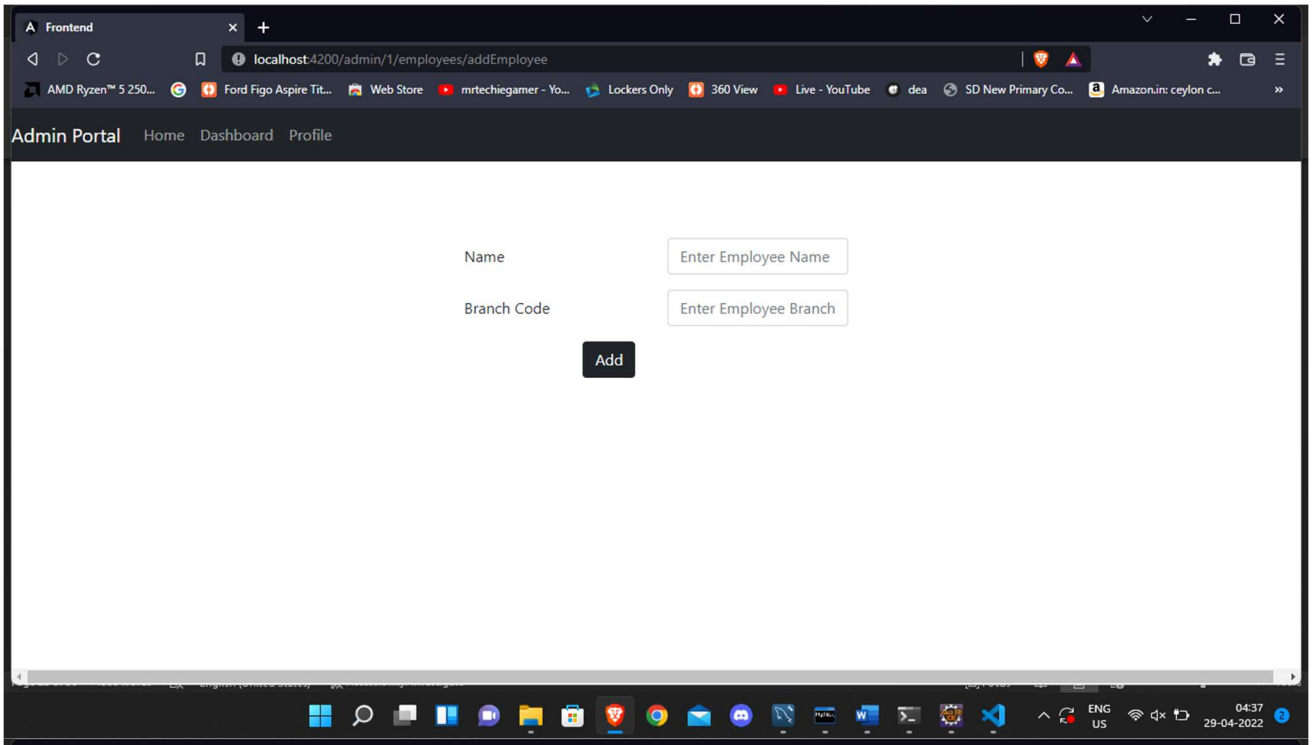

Frontend
localhost:4200/employee/1/bankAccounts/KKBK0000148
Employee Portal
Home
Dashboard
Profile
Add New Bank Account
AccountNo
CustomerNo
Branch
Type
Account Balance
54001
1
KKBK0000148
savings
54006
3
KKBK0000148
savings

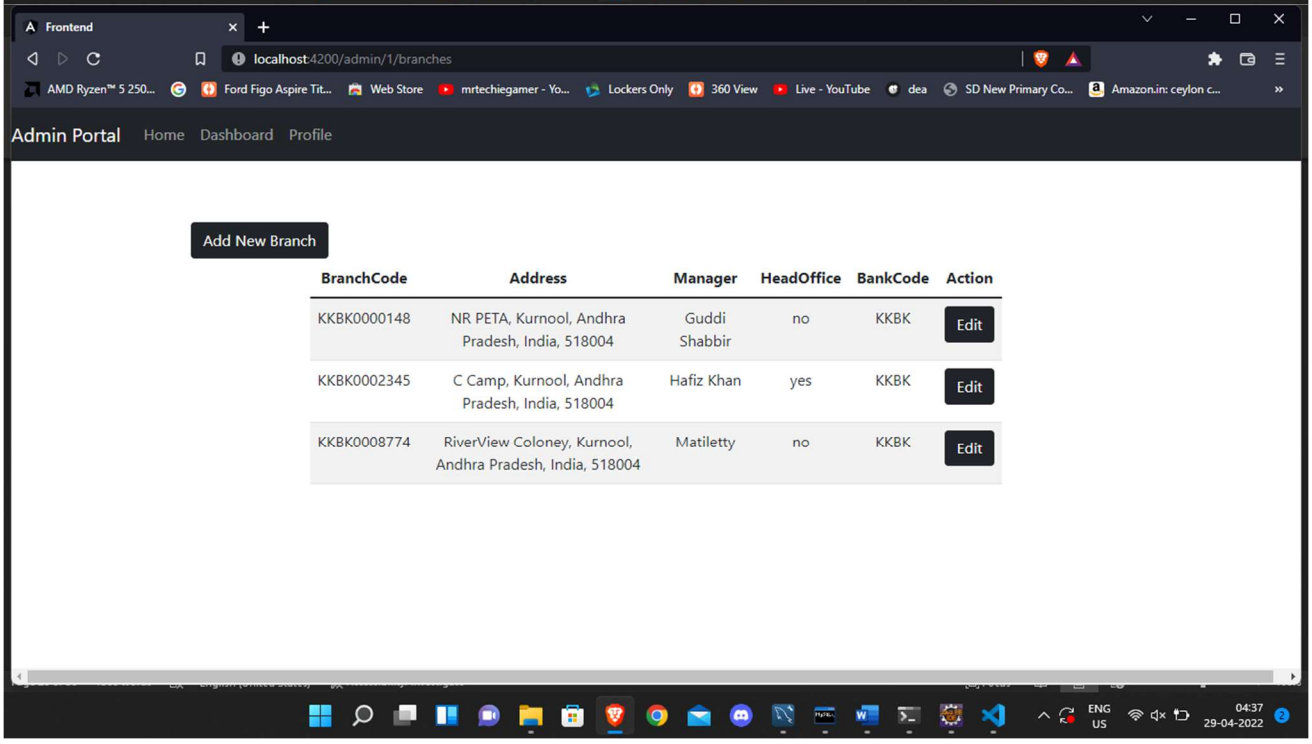

Frontend
localhost:4200/employee/1/bankAccount/KKBK0000148/create
AMD Ryzen™ 5 250...
Ford Figo Aspire Tit...
Web Store
mrtechiegamer - Yo...
Lockers Only
360 View
Live - YouTube
dea
SD New Primary Co...
Amazon.in: ceylon c...
Employee Portal
Home
Profile
Acc No
Enter Account No
Customer
Enter Customer ID
Branch
KKBK0000148
Type
Enter Type(savings or c
Balance
Enter Balance
Create
ENG
US
04:36
29-04-2022

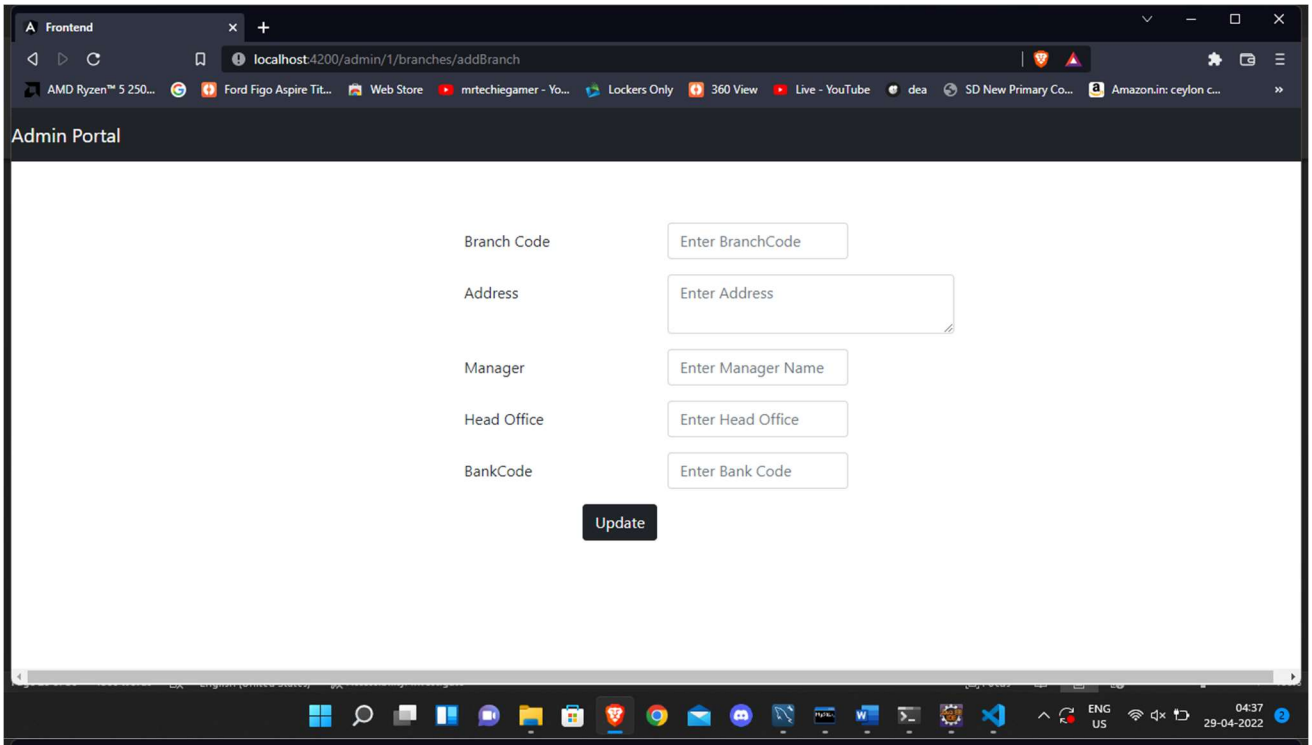

Frontend
localhost:4200/admin/index
AMD Ryzen™ 5 250...
Ford Figo Aspire Tit...
Web Store
mrtechiegamer - Yo...
Lockers Only
360 View
Live - YouTube
dea
SD New Primary Co...
Amazon.in: ceylon c...
Admin Portal
Select Bank
KKBK
Admin Id
1
Login
ENG
US
04:36
29-04-2022

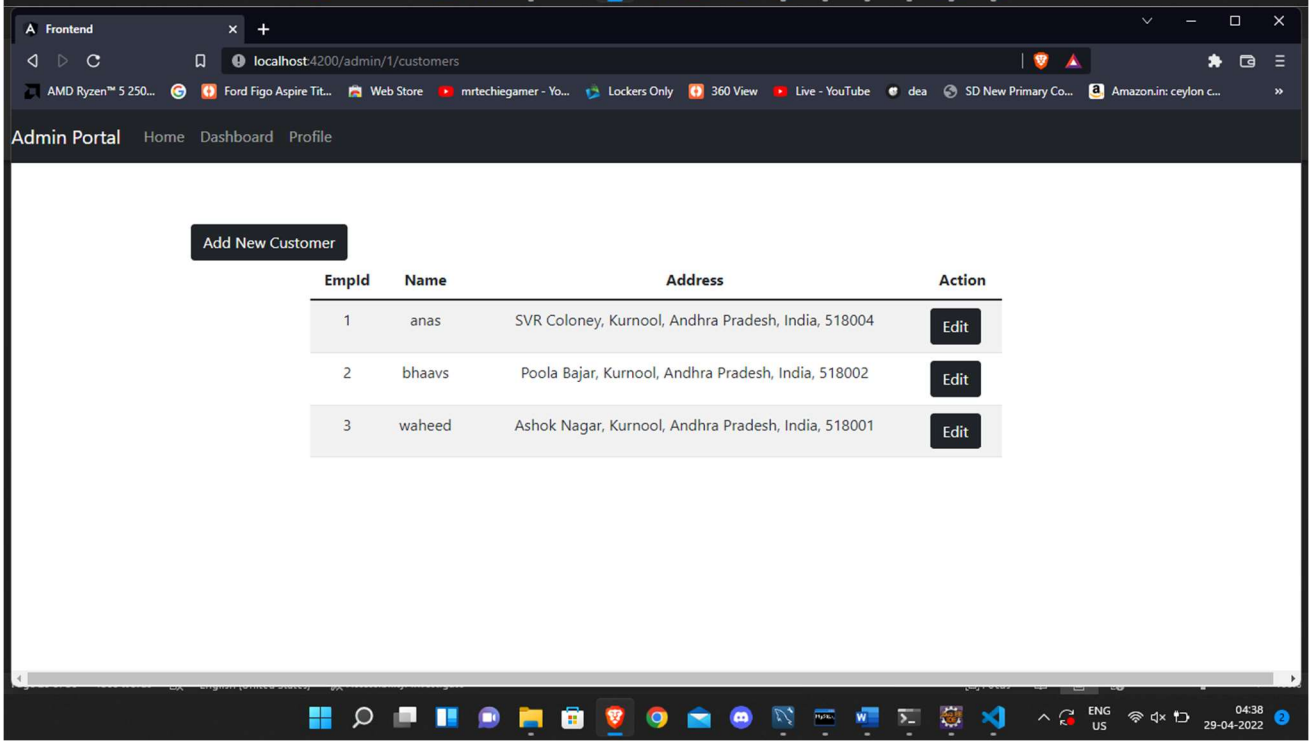

Frontend
localhost:4200/admin/1/dashboard
Admin Portal
Home
Dashboard
Profile
Bank : HDFC
Transactions
Branches
Employees
Customers

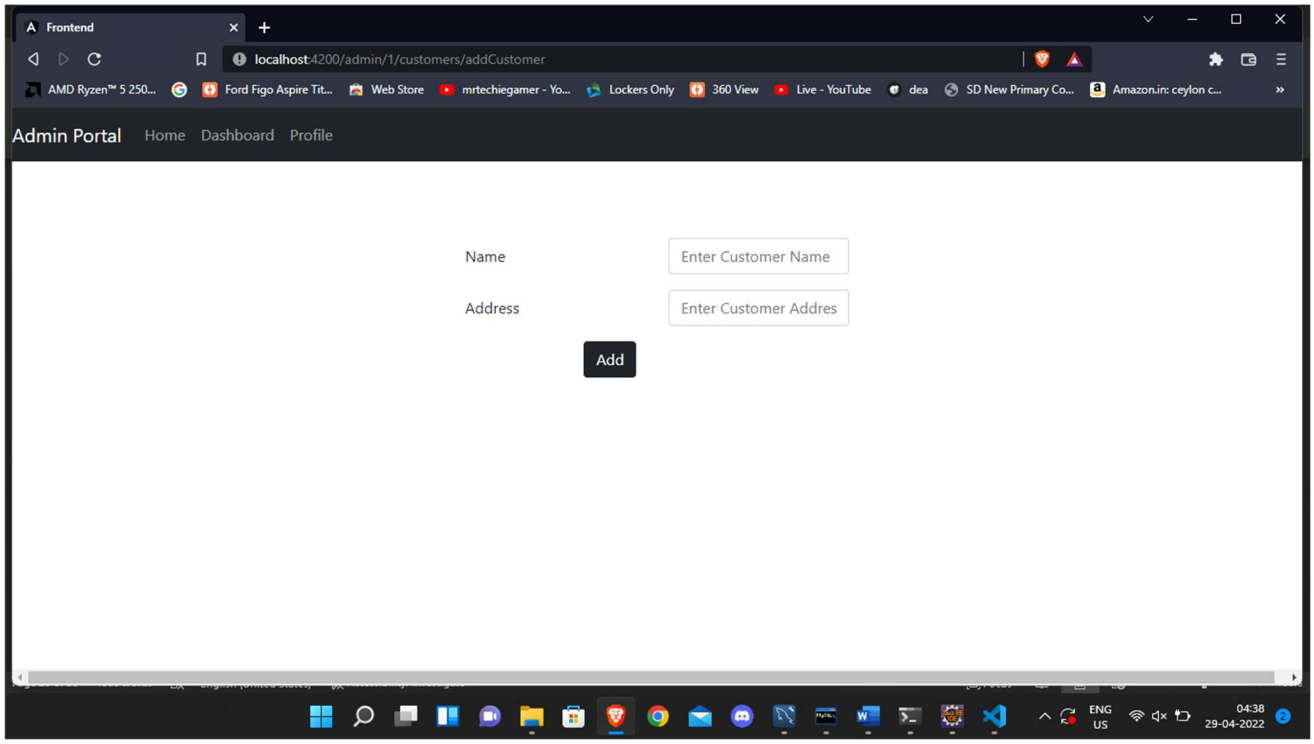

Frontend
localhost:4200/admin/1/employees
Admin Portal
Home
Dashboard
Profile
Add New Employee
EmpId
Employee Name
BranchCode
Action
1
TG Venkatesh
KKBK0000148
Edit
Remove
2
Chandra Babu
KKBK0009669
Edit
Remove
3
Jagan Anna
KKBK0000148
Edit
Remove


Frontend
localhost:4200/admin/1/employees/addEmployee
Admin Portal
Home
Dashboard
Profile
Name
Enter Employee Name
Branch Code
Enter Employee Branch
Add


Frontend
localhost:4200/admin/1/branches
Admin Portal
Home
Dashboard
Profile
Add New Branch
BranchCode
Address
Manager
HeadOffice
BankCode
Action
KKBK0000148
NR PETA, Kurnool, Andhra Pradesh, India, 518004
Guddi Shabbir
no
KKBK
Edit
KKBK0002345
C Camp, Kurnool, Andhra Pradesh, India, 518004
Hafiz Khan
yes
KKBK
Edit
KKBK0008774
RiverView Coloney, Kurnool, Andhra Pradesh, India, 518004
Matiletty
no
KKBK
Edit


Frontend
localhost:4200/admin/1/branches/addBranch
Admin Portal
Branch Code
Enter BranchCode
Address
Enter Address
Manager
Enter Manager Name
Head Office
Enter Head Office
BankCode
Enter Bank Code
Update


Frontend
localhost:4200/admin/1/customers
Admin Portal
Home
Dashboard
Profile
Add New Customer
EmpId
Name
Address
Action
1
anas
SVR Coloney, Kurnool, Andhra Pradesh, India, 518004
Edit
2
bhaavs
Poola Bajar, Kurnool, Andhra Pradesh, India, 518002
Edit
3
waheed
Ashok Nagar, Kurnool, Andhra Pradesh, India, 518001
Edit



CONCLUSION:

Customers can browse, edit, and update their profiles, as well as withdraw and deposit monies from their bank accounts, using this banking system. Employees and administrators can access all transactions as well as customer information in the system. The system's administrator can also create, update, and delete records.

REFERENCES:

8.1. stackoverflow.com

8.2. github.com/akash-coded