CAPSTONE PROJECT: BANKING SYSTEMS

1. SYSTEM REQUIREMENT SPECIFICATION:

1.1. Software Requirements:

- Operating System: ubuntu 20.04
- Coding language: Java
- Data Base: MySQL
- Front end: browser (recent version)

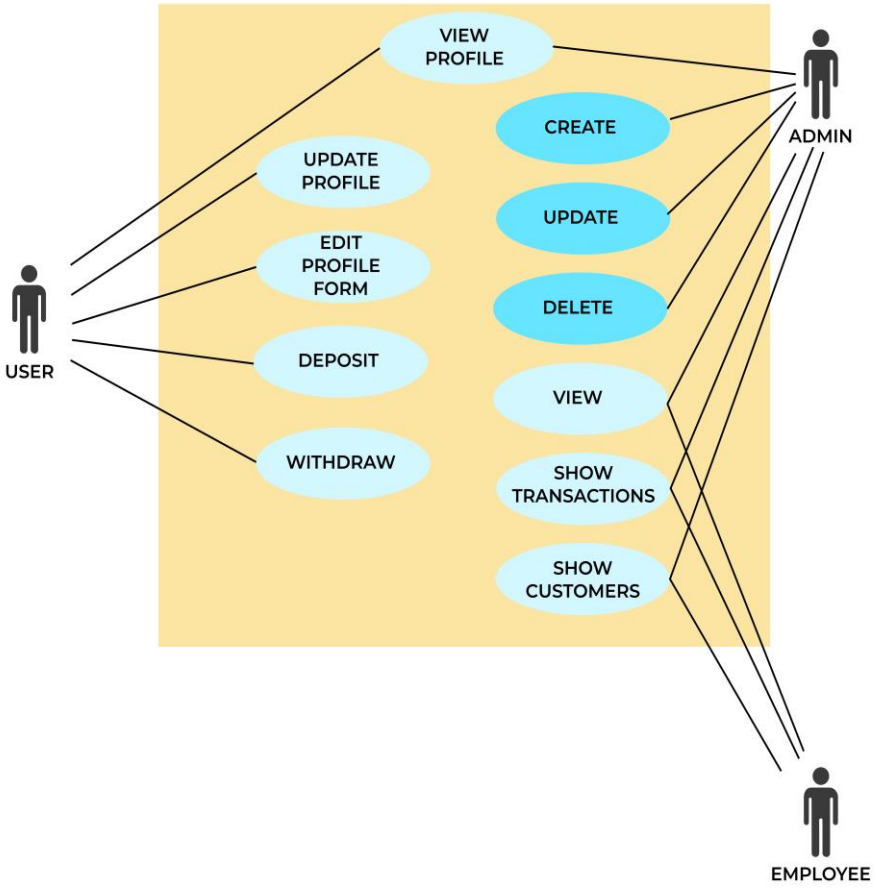
1.2. Hardware Requirements:

- Personal computer with keyboard and mouse maintained with uninterrupted power supply.
- Processor: AMD Ryzen 5
- Installed Memory (RAM): 8.00 GB

2. PROJECT SCOPE:

- Front-end for the bank employees who can view the various transactional details and perform the required operations.
- 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.
- Front-end for the Admin to view all the details of all customers and employees.

3. USE CASE DIAGRAM:



Use Case diagram is used for representing the problem statement, the actors in it, and their functionality in a behavioral manner. In the above figure and the system, there are 3 different actors:

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 customer 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 details such as password, etc.)

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 in the basic flow, the details entered by the user/admin/employee are invalid, then the user/admin/employee is informed that their login attempt has failed. The user may re-enter correct details or create a new account.

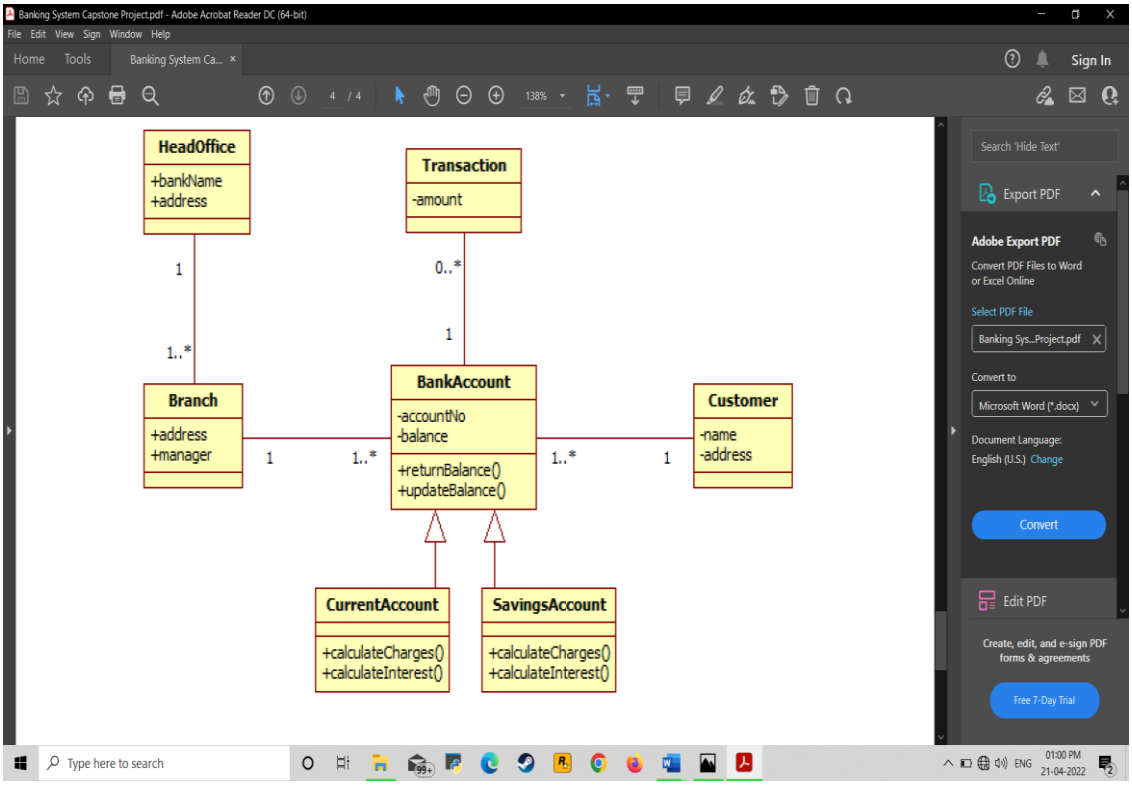
- Pre-conditions:

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

- Post-conditions:

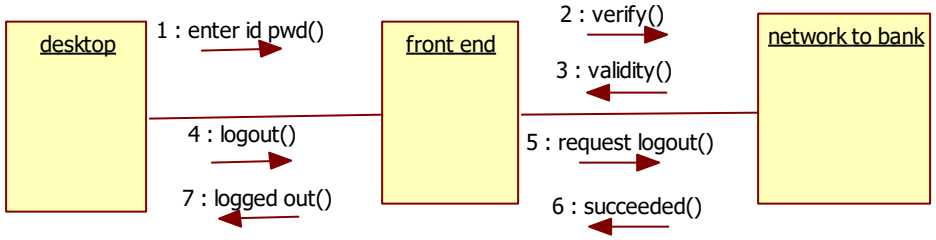
4.8. The database is modified after transaction.

5. CLASS DIAGRAM:

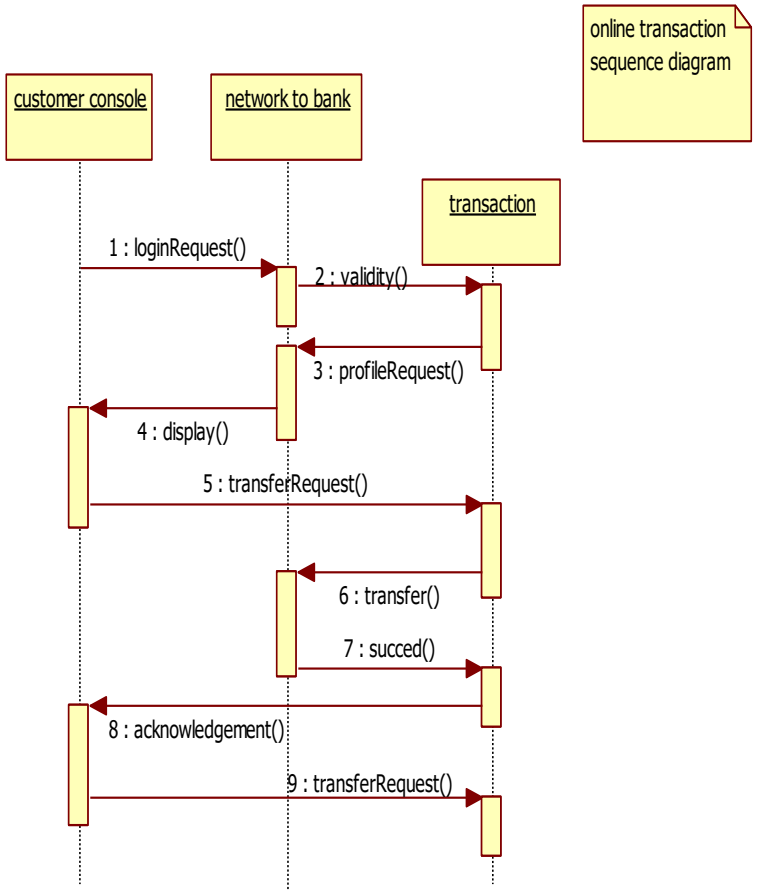


6. SEQUENCE DIAGRAM:

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:

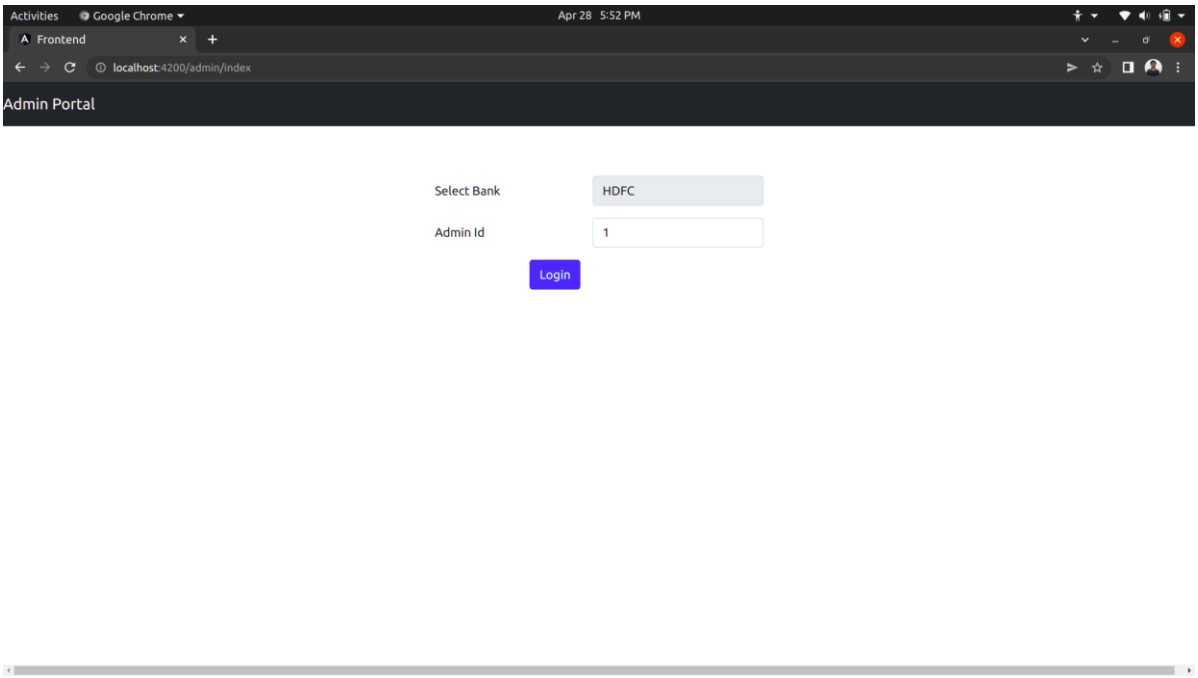
CLASS NAME: BankService		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For Creating Bank and Viewing The details.		Bank model. Bank Repository.

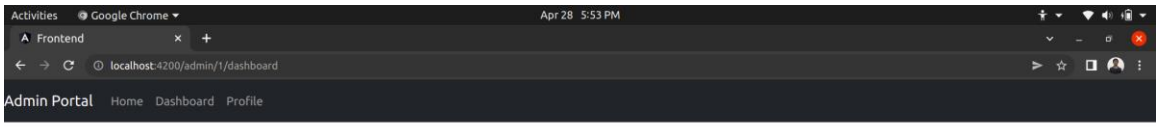
CLASS NAME: BranchService		
RESPONSIBILITIES	DEPENDABLES	DEPENDENCIES
For creating Branch, get the branches, get the branches by IFSC, deleting Branch by IFSC, updating Branch manager name and getting Customer by Branches.		Branch and Customer models. Branch and Customer Repository.

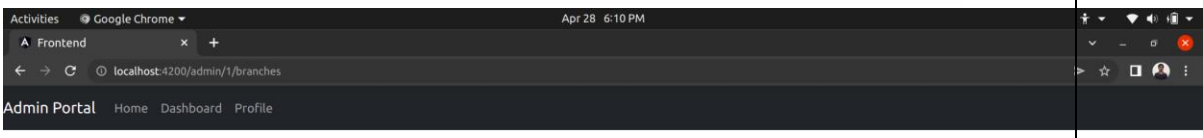
CLASS NAME: TransactionService

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

8. IMPLEMENTATION:

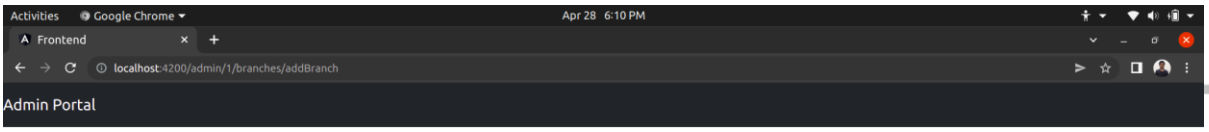






Add New Branch

BranchCode	Address	Manager	HeadOffice	BankCode	Action
HDFC0000148	East Street, Pune, Maharashtra, India, 411001	Jesse Pinkman	no	HDFC	Edit
HDFC0009116	Akurdi, Pune, Maharashtra, India, 411035	Gus Fring	yes	HDFC	Edit
HDFC0009669	Yerwada, Pune, Maharashtra, India, 411006	Saul Goodman	no	HDFC	Edit



Branch Code — Enter BranchCode

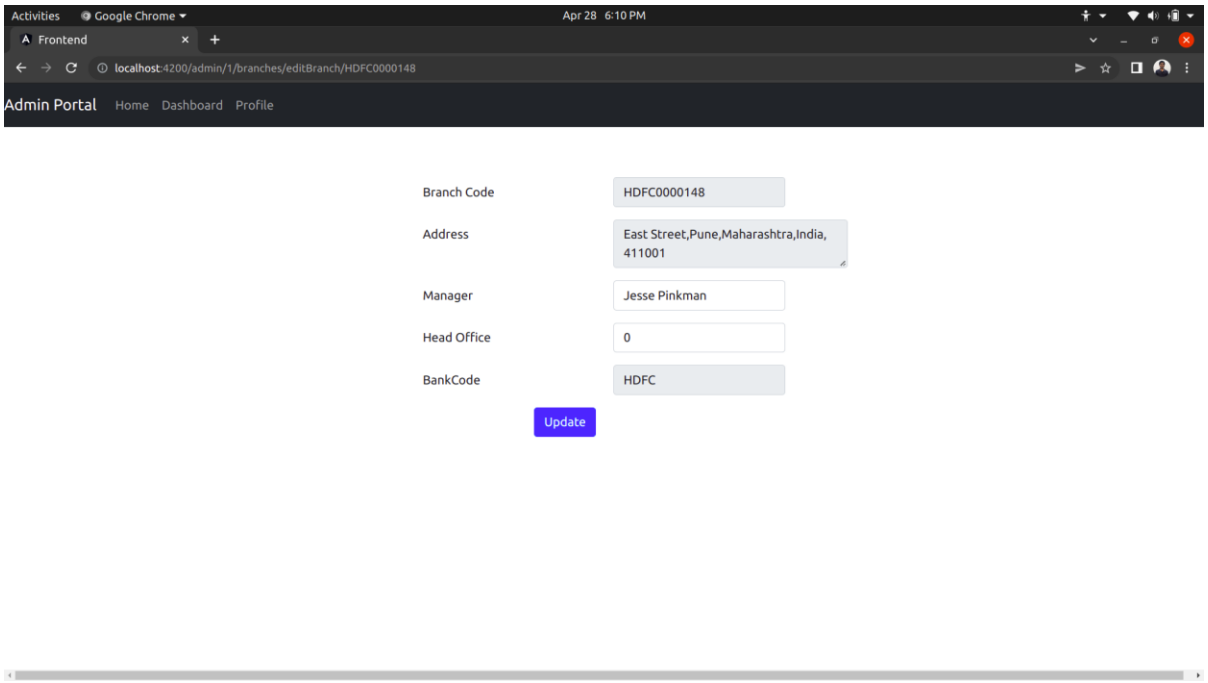
Address — Enter Address

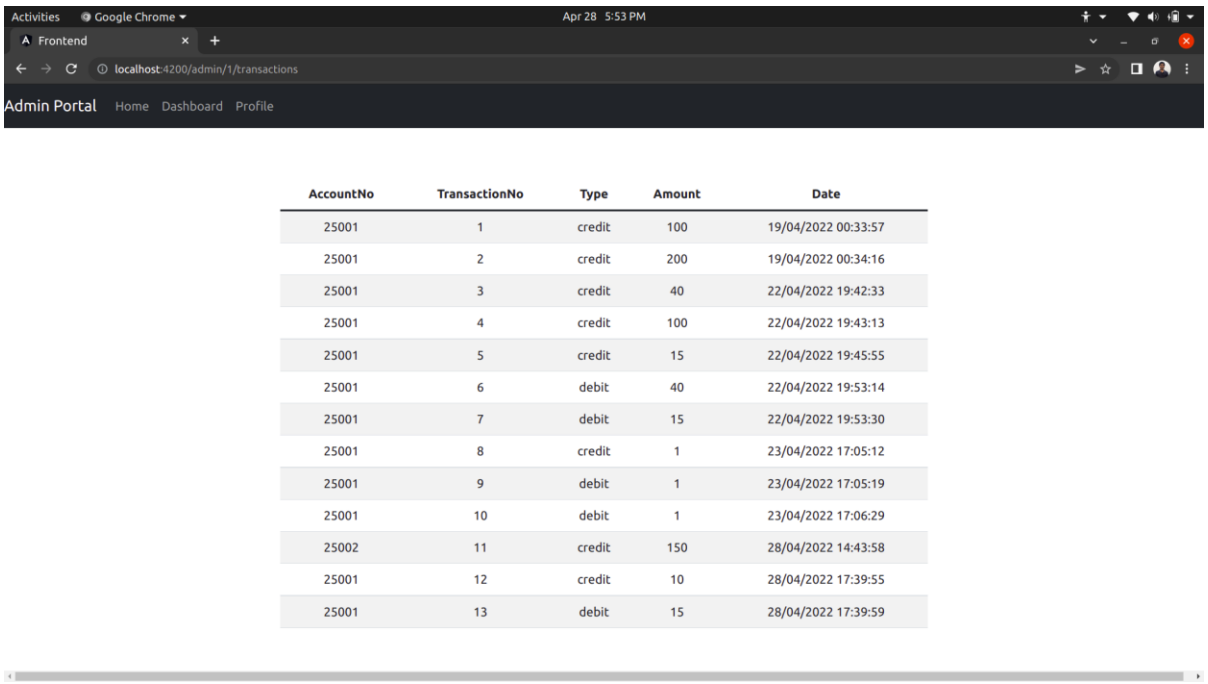
Manager — Enter Manager Name

Head Office — Enter Head Office

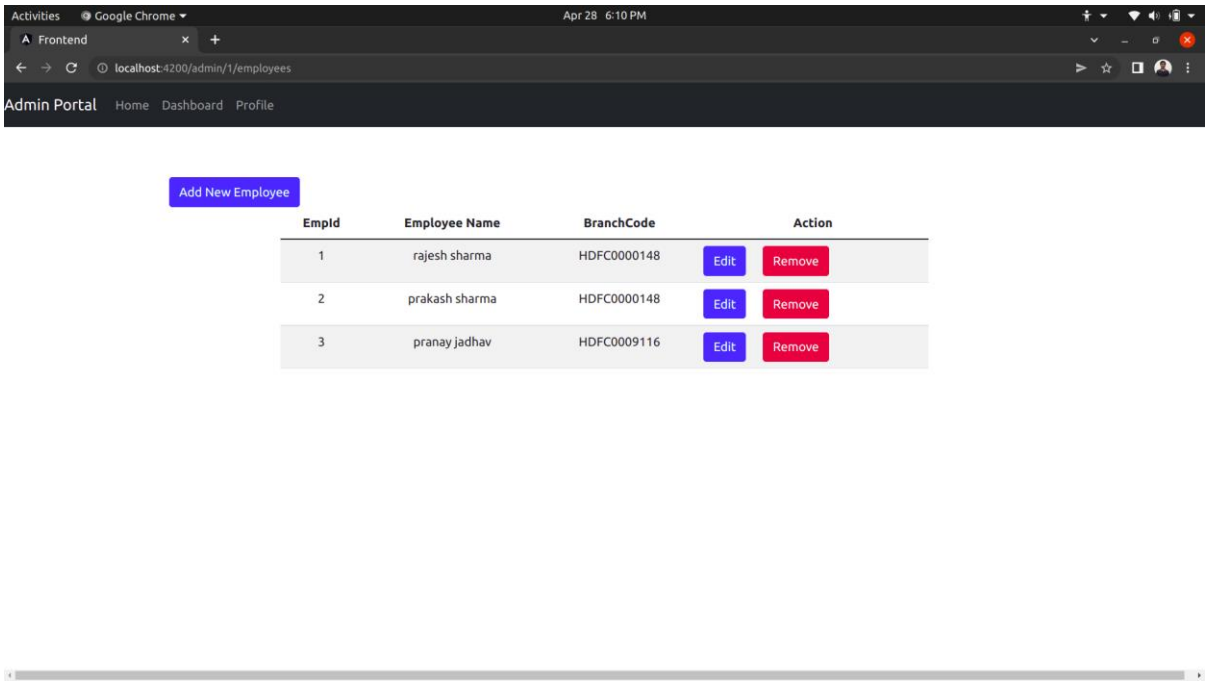
BankCode — Enter Bank Code

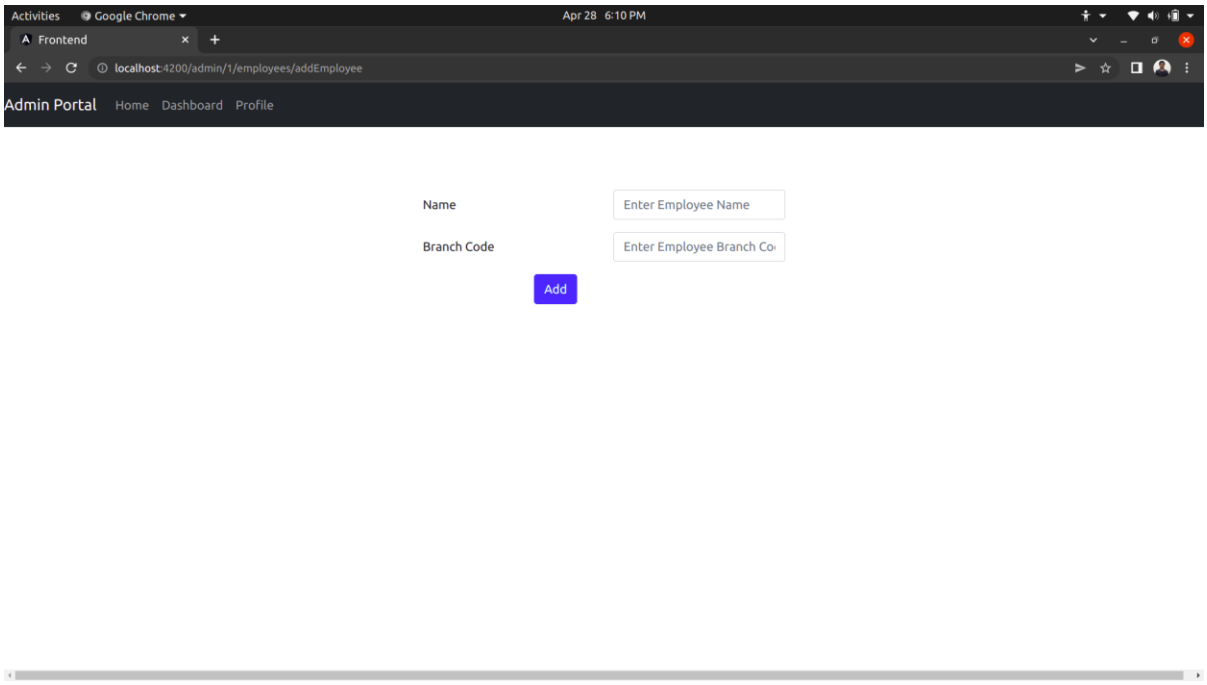
Update

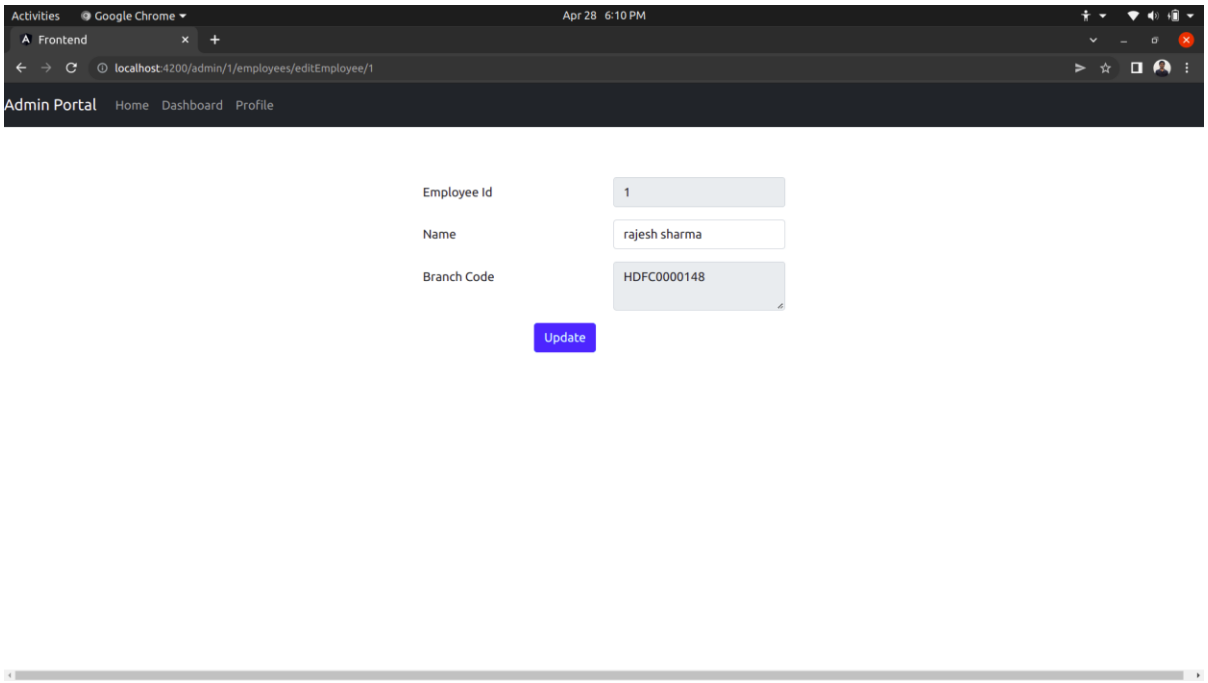


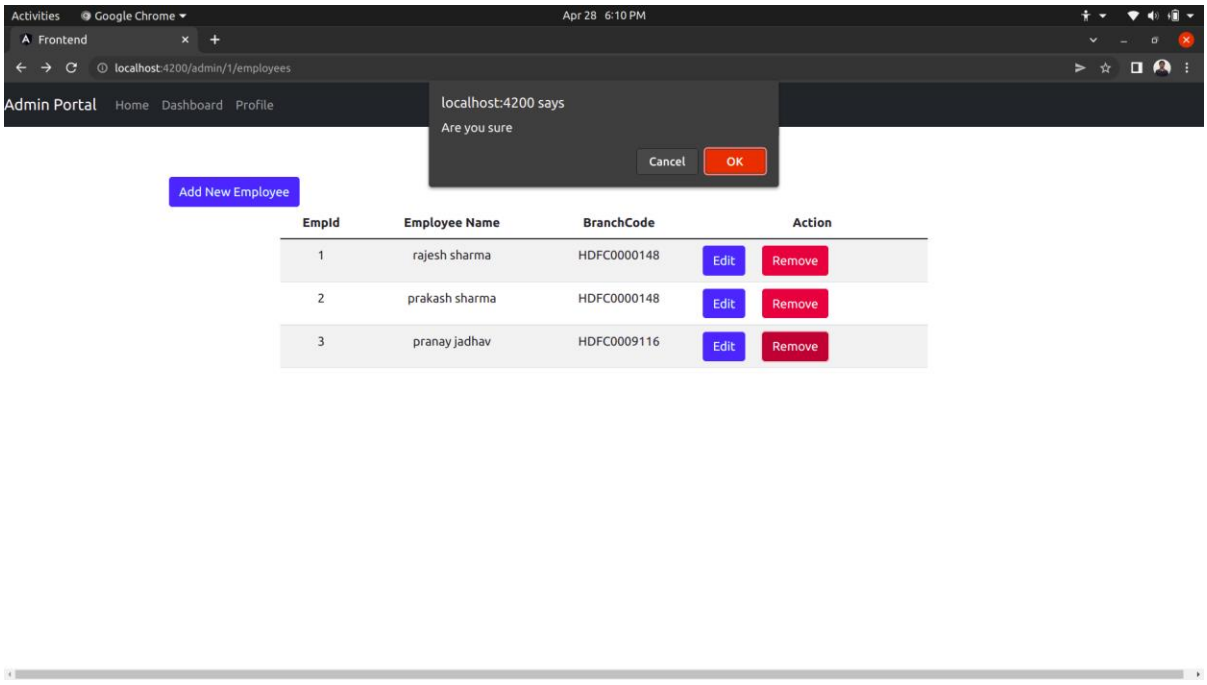


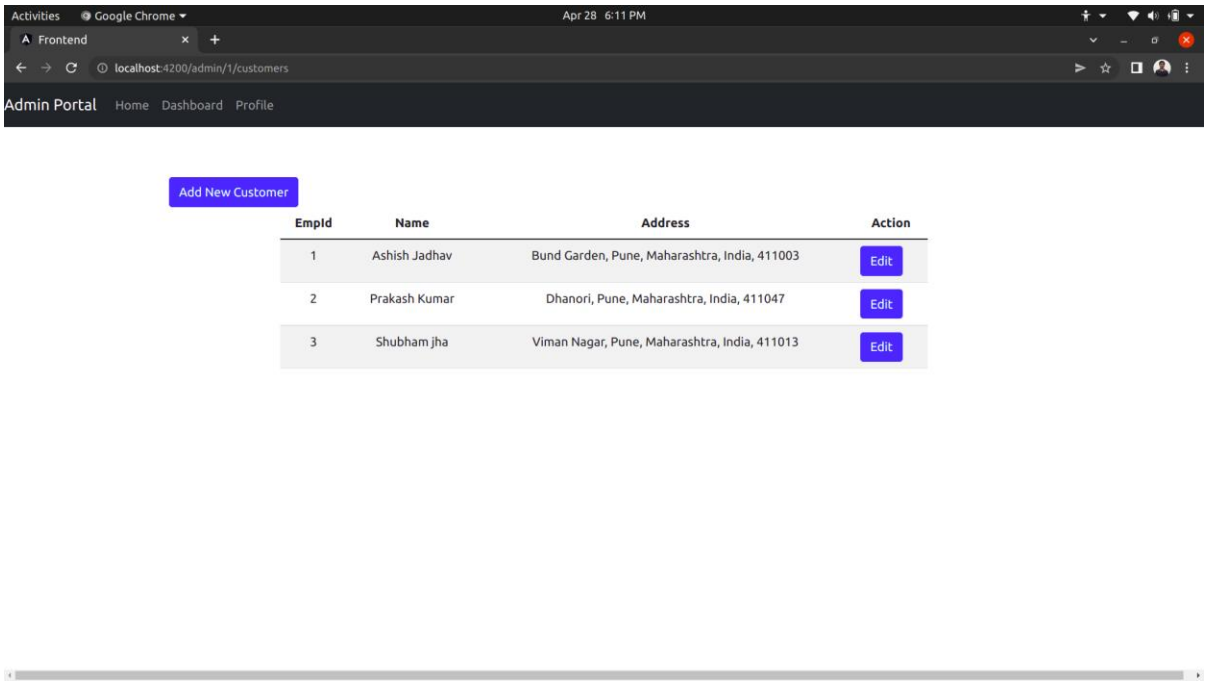
AccountNo	TransactionNo	Type	Amount	Date
25001	1	credit	100	19/04/2022 00:33:57
25001	2	credit	200	19/04/2022 00:34:16
25001	3	credit	40	22/04/2022 19:42:33
25001	4	credit	100	22/04/2022 19:43:13
25001	5	credit	15	22/04/2022 19:45:55
25001	6	debit	40	22/04/2022 19:53:14
25001	7	debit	15	22/04/2022 19:53:30
25001	8	credit	1	23/04/2022 17:05:12
25001	9	debit	1	23/04/2022 17:05:19
25001	10	debit	1	23/04/2022 17:06:29
25002	11	credit	150	28/04/2022 14:43:58
25001	12	credit	10	28/04/2022 17:39:55
25001	13	debit	15	28/04/2022 17:39:59

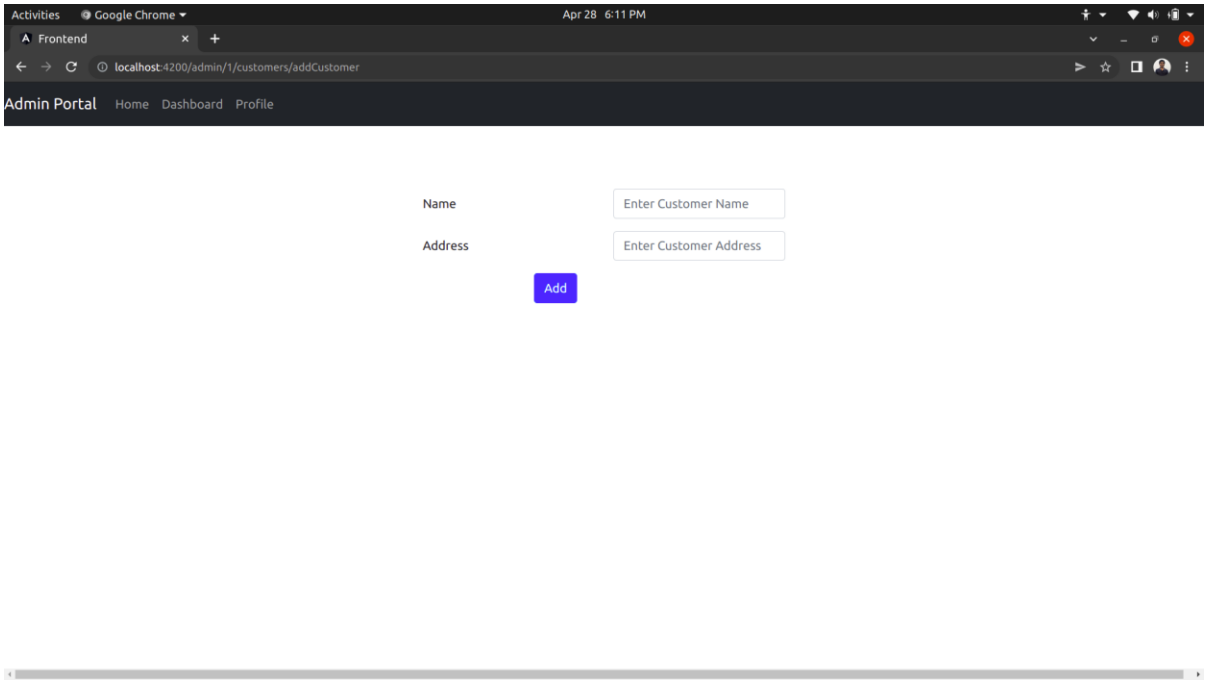

Activities
Google Chrome
Apr 28 6:10 PM
Frontend
localhost:4200/admin/1/employees
Admin Portal
Home
Dashboard
Profile
Add New Employee
EmpId
Employee Name
BranchCode
Action
1
rajesh sharma
HDFC0000148
Edit
Remove
2
prakash sharma
HDFC0000148
Edit
Remove
3
pranay jadhav
HDFC0009116
Edit
Remove

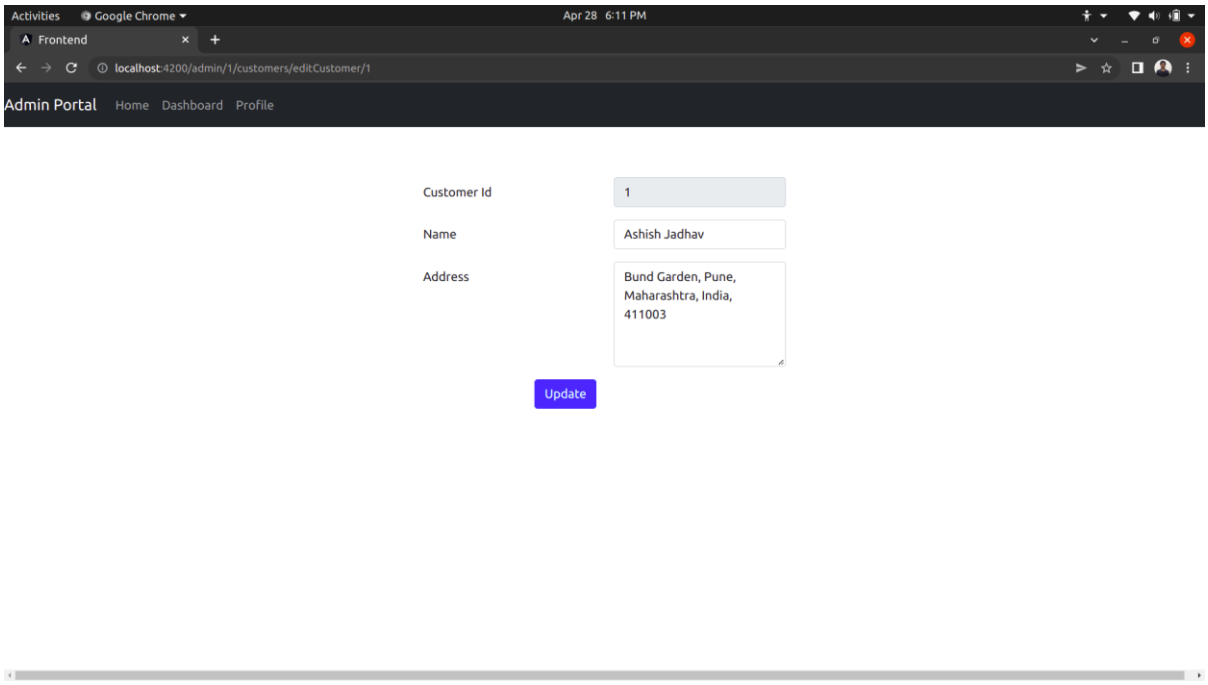

Activities
Google Chrome
Apr 28 6:10 PM
Frontend
localhost:4200/admin/1/employees/addEmployee
Admin Portal
Home
Dashboard
Profile
Name
Enter Employee Name
Branch Code
Enter Employee Branch Co
Add

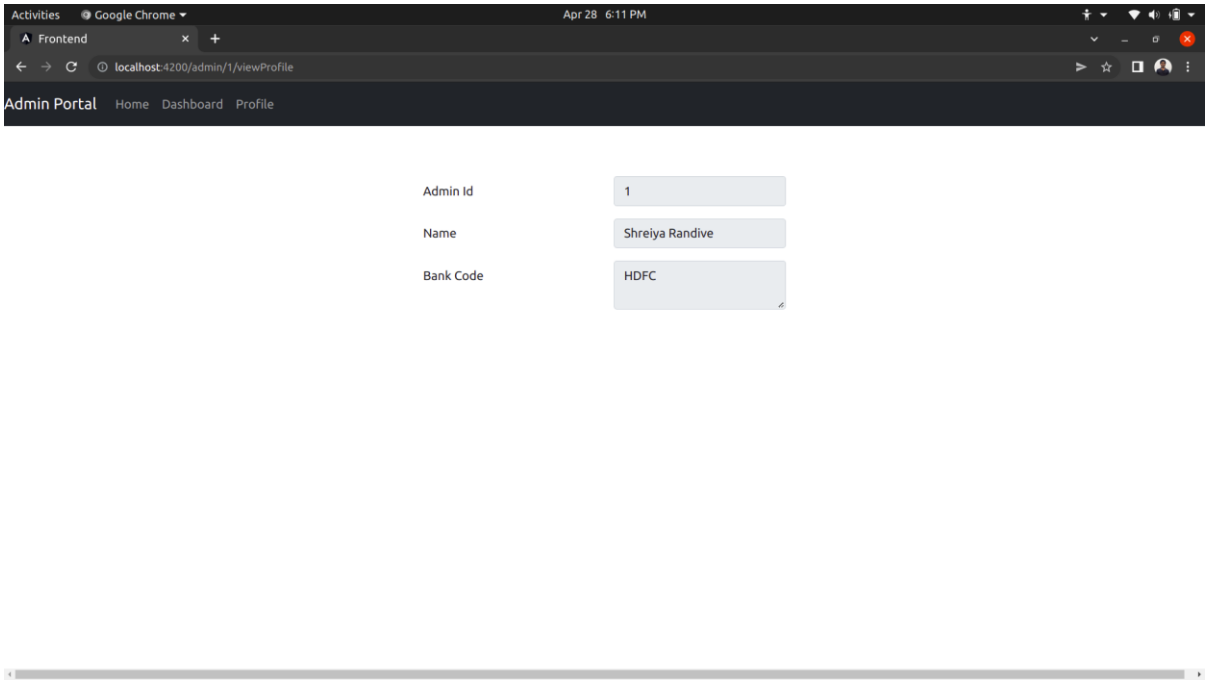

Activities
Google Chrome
Apr 28 6:10 PM
Frontend
localhost:4200/admin/1/employees/editEmployee/1
Admin Portal
Home
Dashboard
Profile
Employee Id
1
Name
rajesh sharma
Branch Code
HDFC0000148
Update

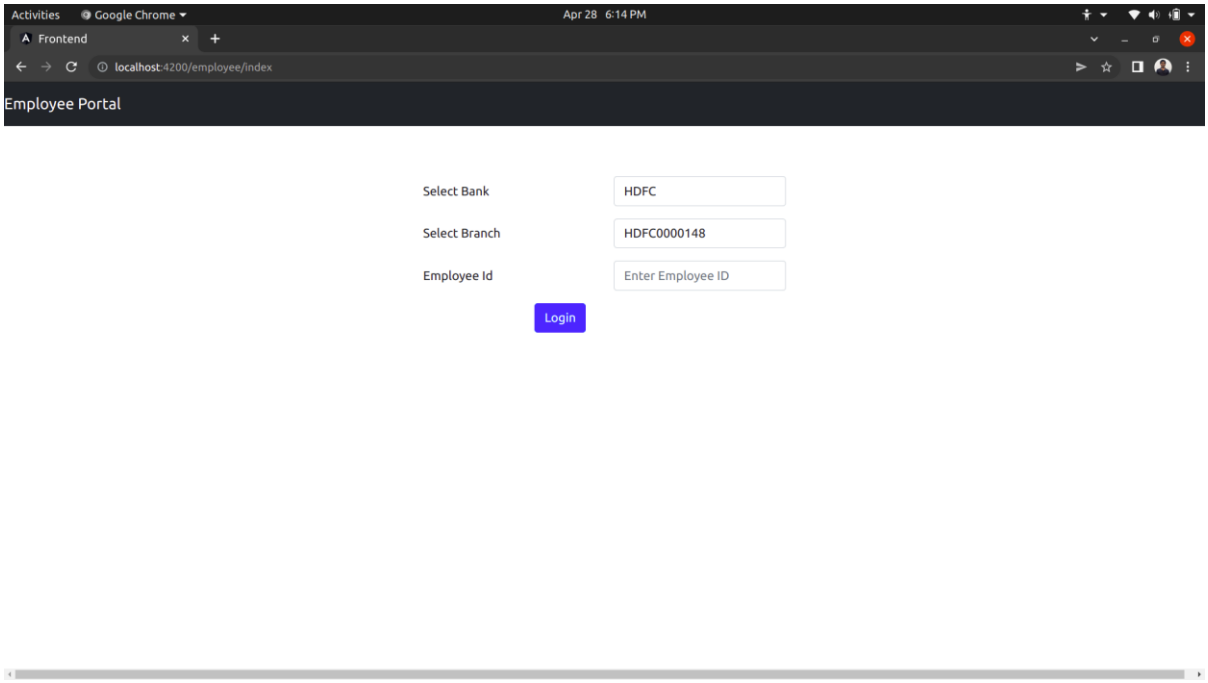

Activities
Google Chrome
Apr 28 6:10 PM
Frontend
localhost:4200/admin/1/employees
Admin Portal
Home
Dashboard
Profile
localhost:4200 says
Are you sure
Cancel
OK
Add New Employee
EmpId
Employee Name
BranchCode
Action
1
rajesh sharma
HDFC0000148
Edit
Remove
2
prakash sharma
HDFC0000148
Edit
Remove
3
pranay jadhav
HDFC0009116
Edit
Remove

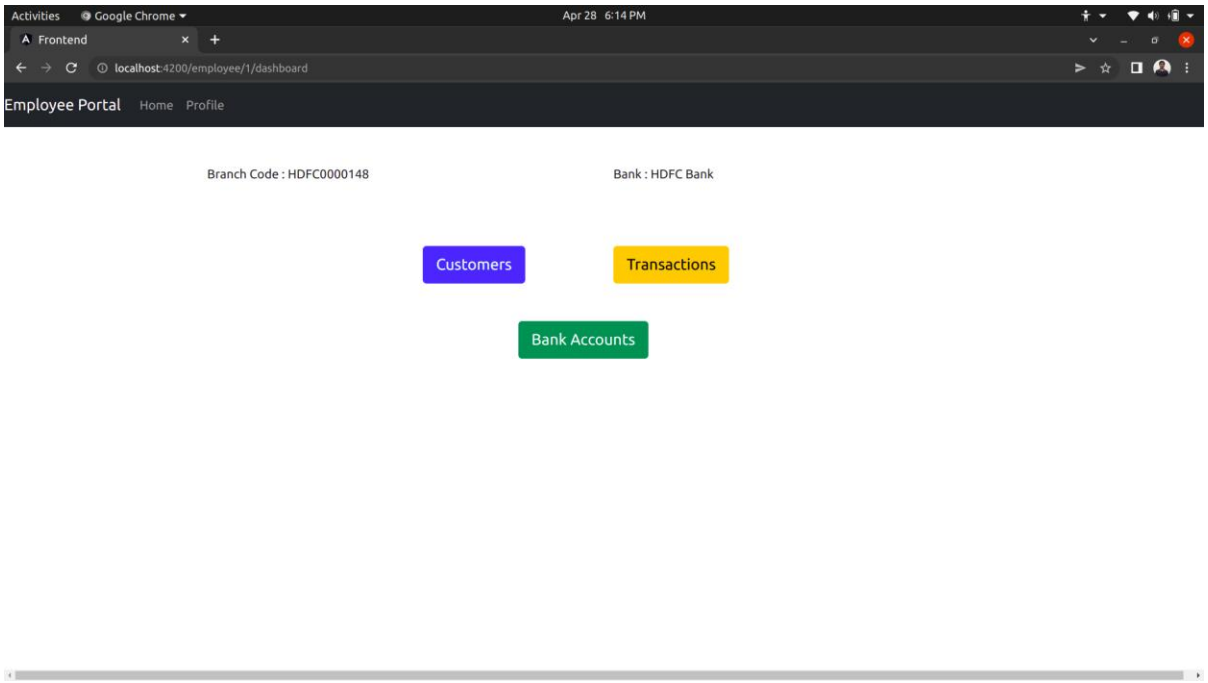

Activities
Google Chrome
Apr 28 6:11 PM
Frontend
localhost:4200/admin/1/customers
Admin Portal
Home
Dashboard
Profile
Add New Customer
EmpId
Name
Address
Action
1
Ashish Jadhav
Bund Garden, Pune, Maharashtra, India, 411003
Edit
2
Prakash Kumar
Dhanori, Pune, Maharashtra, India, 411047
Edit
3
Shubham jha
Viman Nagar, Pune, Maharashtra, India, 411013
Edit

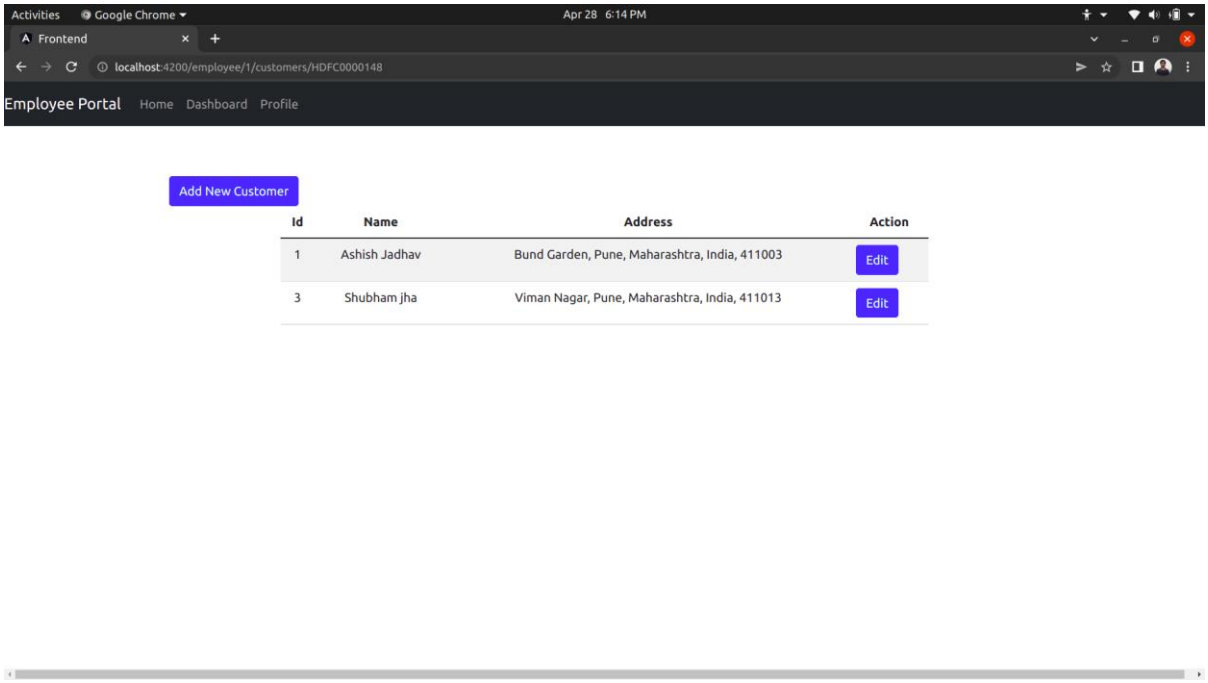

Activities
Google Chrome
Apr 28 6:11 PM
Frontend
localhost:4200/admin/1/customers/addCustomer
Admin Portal
Home
Dashboard
Profile
Name
Enter Customer Name
Address
Enter Customer Address
Add

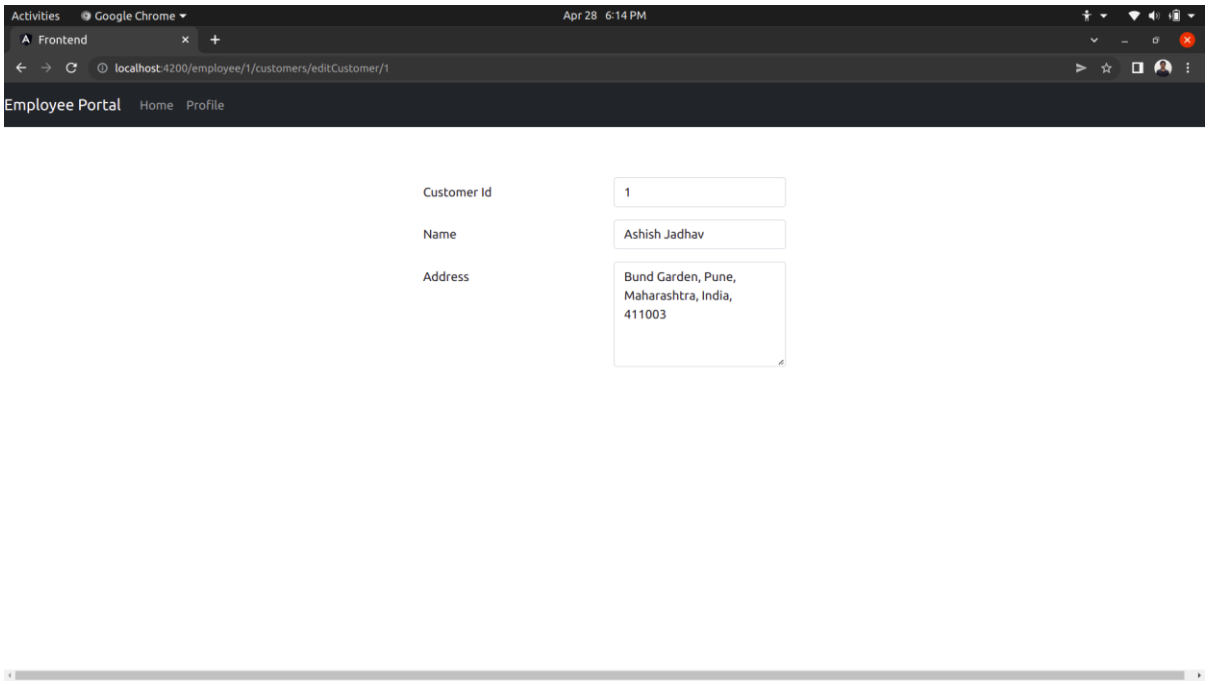

Activities
Google Chrome
Apr 28 6:11 PM
Frontend
localhost:4200/admin/1/customers/editCustomer/1
Admin Portal
Home
Dashboard
Profile
Customer Id
1
Name
Ashish Jadhav
Address
Bund Garden, Pune,
Maharashtra, India,
411003
Update

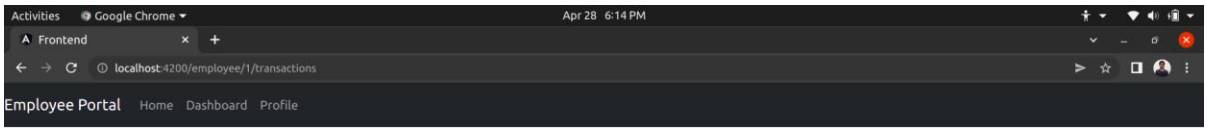

Activities
Google Chrome
Apr 28 6:11 PM
Frontend
localhost:4200/admin/1/viewProfile
Admin Portal
Home
Dashboard
Profile
Admin Id
1
Name
Shreiya Randive
Bank Code
HDFC


Activities
Google Chrome
Apr 28 6:14 PM
Frontend
localhost:4200/employee/index
Employee Portal
Select Bank
HDFC
Select Branch
HDFC0000148
Employee Id
Enter Employee ID
Login

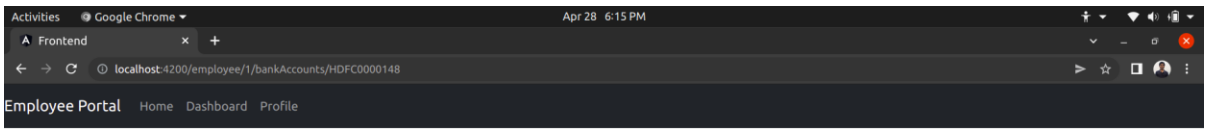

Activities
Google Chrome
Apr 28 6:14 PM
Frontend
localhost:4200/employee/1/dashboard
Employee Portal
Home
Profile
Branch Code : HDFC0000148
Bank : HDFC Bank
Customers
Transactions
Bank Accounts


Activities
Google Chrome
Apr 28 6:14 PM
Frontend
localhost:4200/employee/1/customers/HDFC0000148
Employee Portal
Home
Dashboard
Profile
Add New Customer
Id
Name
Address
Action
1
Ashish Jadhav
Bund Garden, Pune, Maharashtra, India, 411003
Edit
3
Shubham jha
Viman Nagar, Pune, Maharashtra, India, 411013
Edit


Activities
Google Chrome
Apr 28 6:14 PM
Frontend
localhost:4200/employee/1/customers/editCustomer/1
Employee Portal
Home
Profile
Customer Id
1
Name
Ashish Jadhav
Address
Bund Garden, Pune, Maharashtra, India, 411003

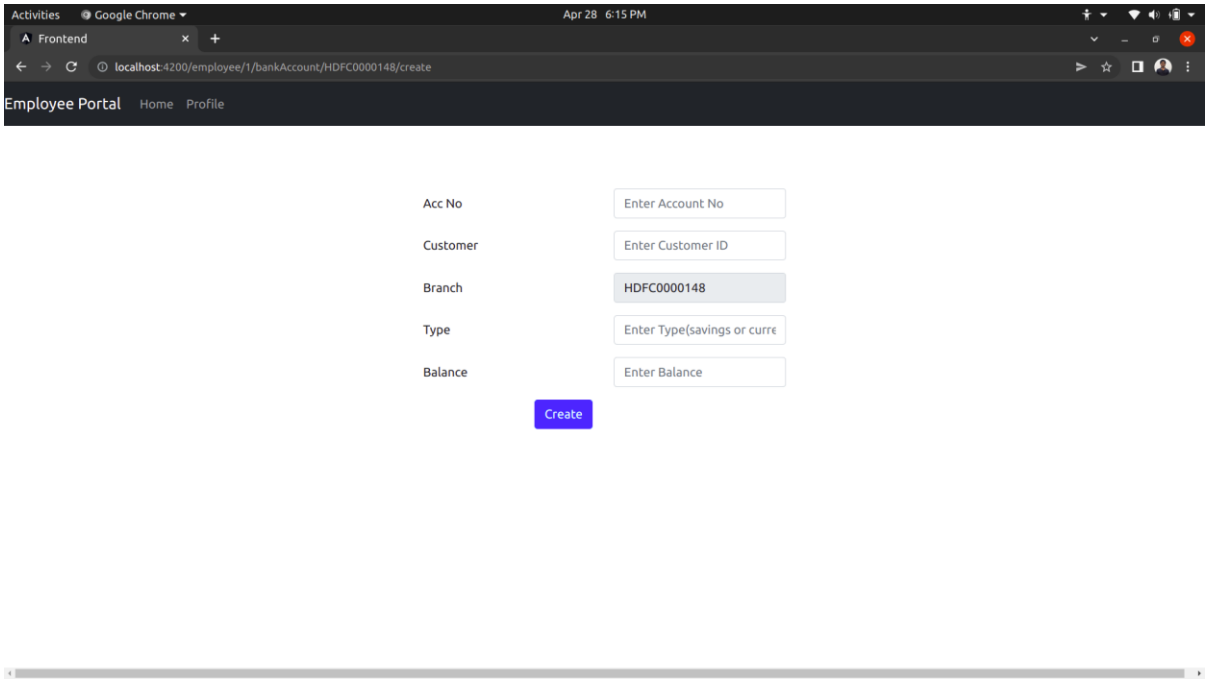


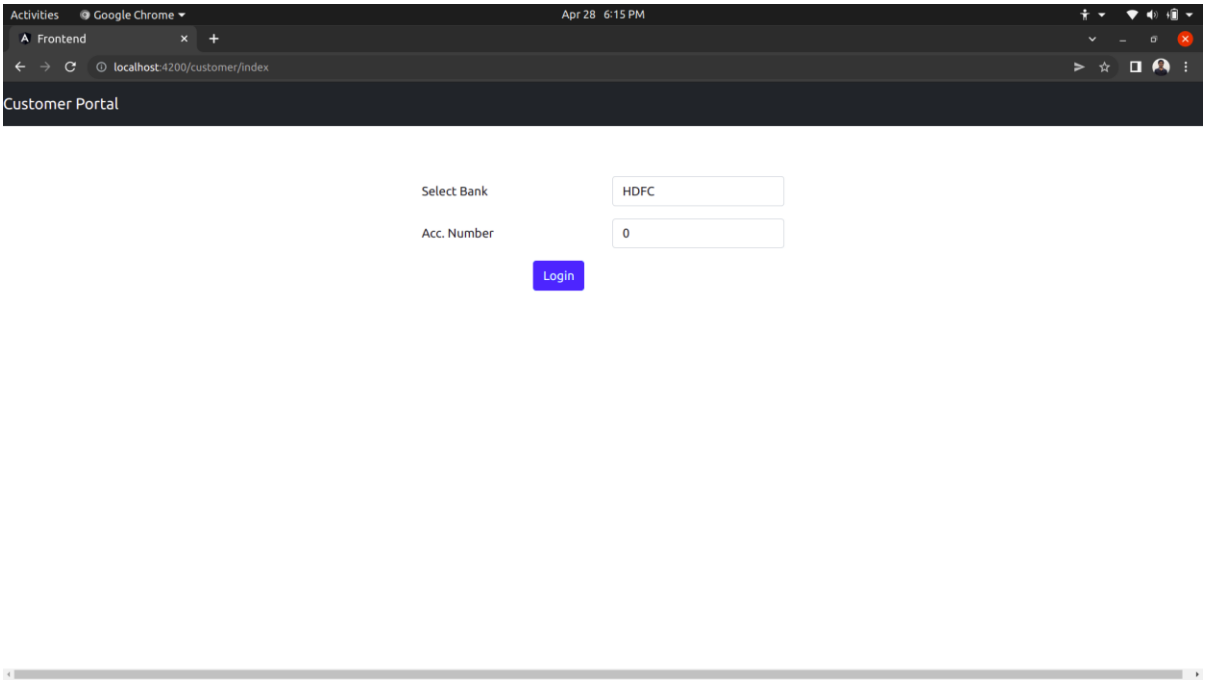
TransactionNo	AccountNo	Amount	Type	Timestamp
1	25001	100	credit	19/04/2022 00:33:57
2	25001	200	credit	19/04/2022 00:34:16
3	25001	40	credit	22/04/2022 19:42:33
4	25001	100	credit	22/04/2022 19:43:13
5	25001	15	credit	22/04/2022 19:45:55
6	25001	40	debit	22/04/2022 19:53:14
7	25001	15	debit	22/04/2022 19:53:30
8	25001	1	credit	23/04/2022 17:05:12
9	25001	1	debit	23/04/2022 17:05:19
10	25001	1	debit	23/04/2022 17:06:29
12	25001	10	credit	28/04/2022 17:39:55
13	25001	15	debit	28/04/2022 17:39:59

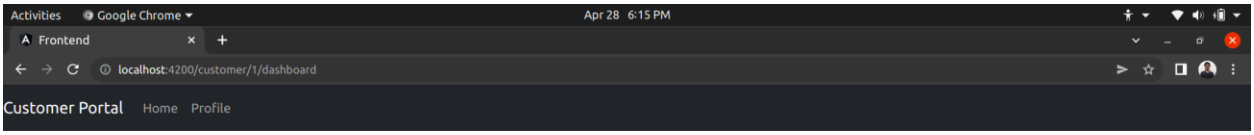


Add New Bank Account

AccountNo	CustomerNo	Branch	Type	Account Balance
25001	1	HDFC0000148	savings	
25006	3	HDFC0000148	savings	

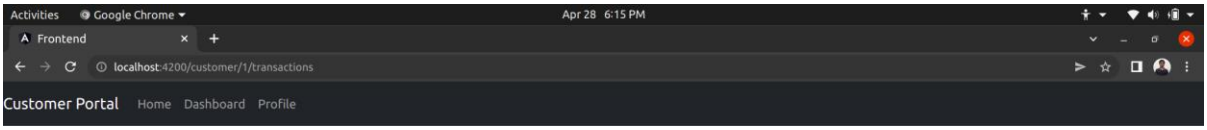

Activities
Google Chrome
Apr 28 6:15 PM
Frontend
localhost:4200/employee/1/bankAccount/HDFC0000148/create
Employee Portal
Home
Profile
Acc No
Enter Account No
Customer
Enter Customer ID
Branch
HDFC0000148
Type
Enter Type(savings or curre
Balance
Enter Balance
Create


Activities
Google Chrome
Apr 28 6:15 PM
Frontend
localhost:4200/customer/index
Customer Portal
Select Bank
HDFC
Acc. Number
0
Login

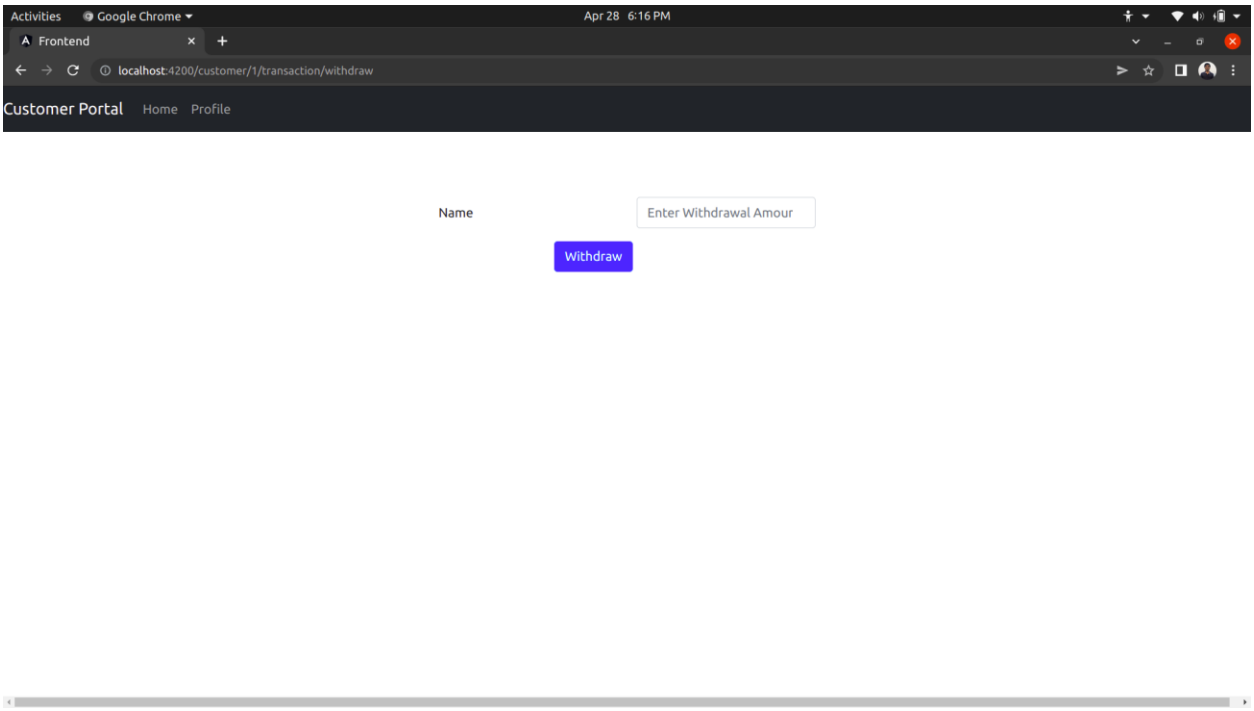


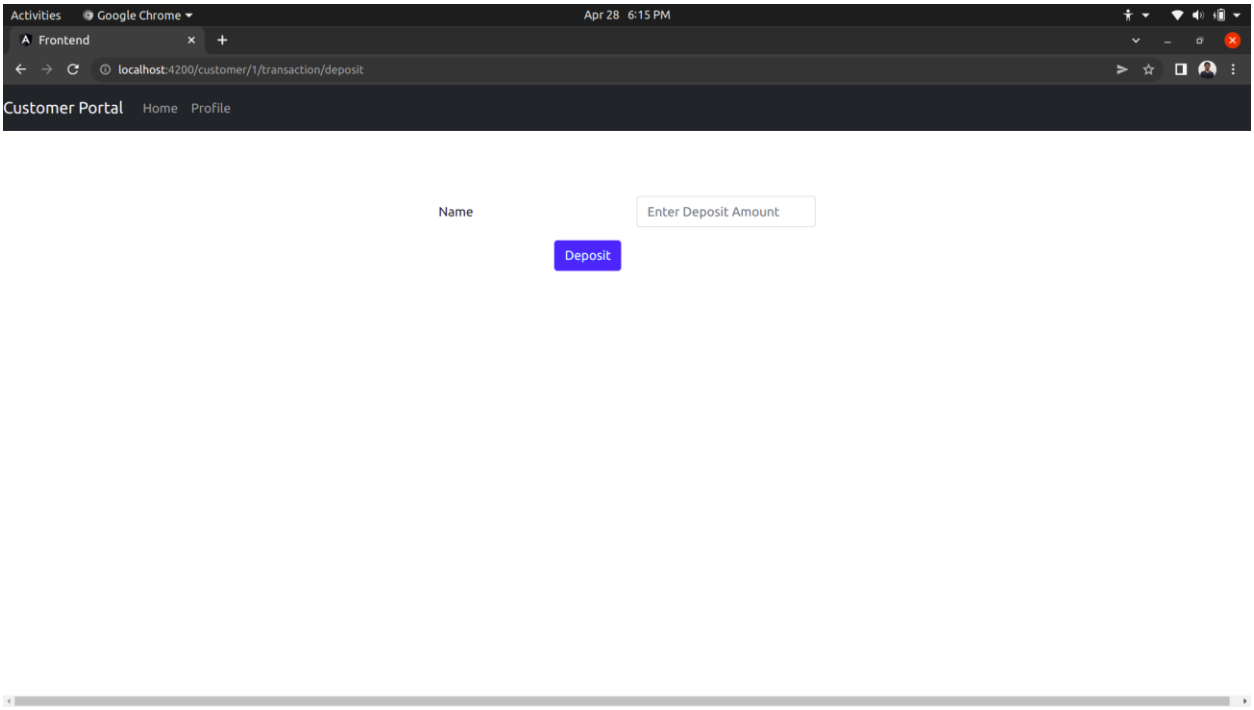
Transactions Deposit

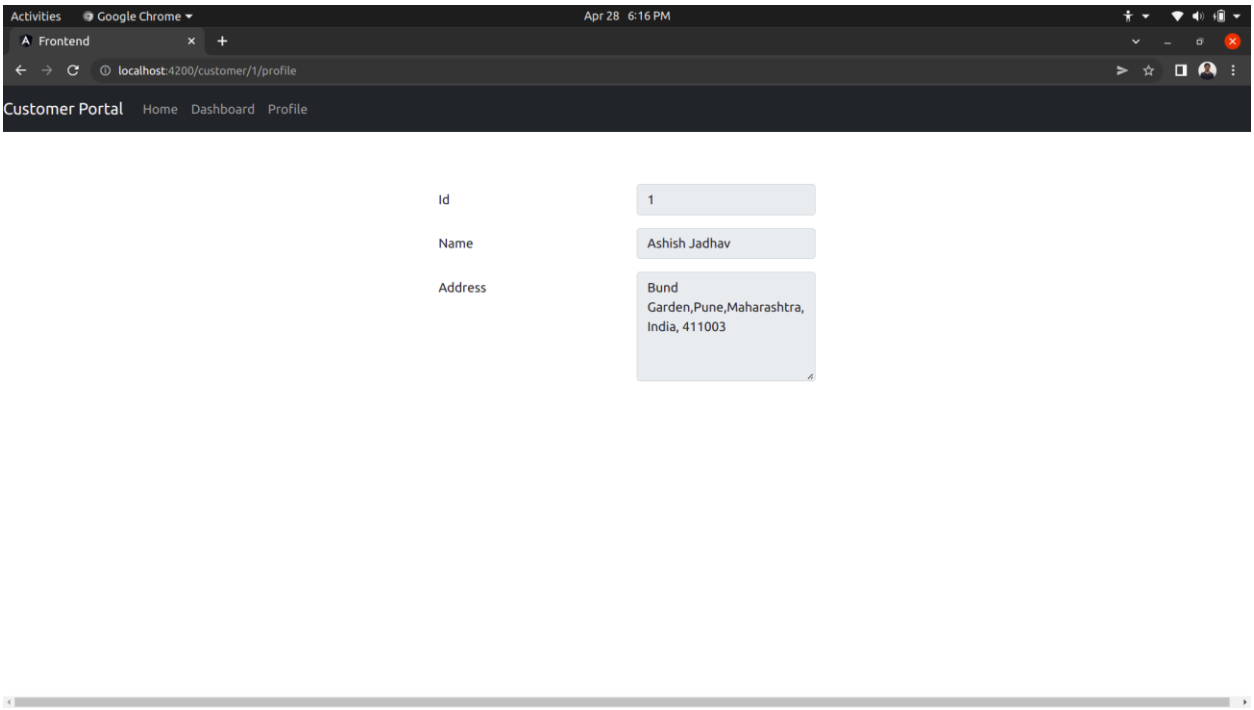
Withdraw



TransactionNo	AccountNo	Amount	Type	Timestamp
1	25001	100	credit	19/04/2022 00:33:57
2	25001	200	credit	19/04/2022 00:34:16
3	25001	40	credit	22/04/2022 19:42:33
4	25001	100	credit	22/04/2022 19:43:13
5	25001	15	credit	22/04/2022 19:45:55
6	25001	40	debit	22/04/2022 19:53:14
7	25001	15	debit	22/04/2022 19:53:30
8	25001	1	credit	23/04/2022 17:05:12
9	25001	1	debit	23/04/2022 17:05:19
10	25001	1	debit	23/04/2022 17:06:29
12	25001	10	credit	28/04/2022 17:39:55
13	25001	15	debit	28/04/2022 17:39:59


Activities
Google Chrome
Apr 28 6:16 PM
Frontend
localhost:4200/customer/1/transaction/withdraw
Customer Portal
Home
Profile
Name
Enter Withdrawal Amour
Withdraw


Activities
Google Chrome
Apr 28 6:15 PM
Frontend
localhost:4200/customer/1/transaction/deposit
Customer Portal
Home
Profile
Name
Enter Deposit Amount
Deposit



9. CONCLUSION:

This banking system allows customers to view, edit, and update their profile along with withdrawal and deposit of funds in their bank accounts. The system allows the employees and the administrator to view all transactions as well as customer details. The admin of the system can also create, update, or delete.

10. REFERENCES:

10.1. stackoverflow.com