



Object Oriented Analysis and Design Using Java

UE20CS352

Bachelor of Technology

in

Computer Science & Engineering

Project Title : ATM Machine

TEAM MEMBERS:

Rajesh M Mysoremath	PES1UG20CS579
Ravi Kiran	PES1UG20CS580
Rohith H	PES1UG20CS581
Sagar S	PES1UG20CS587

January - May 2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

100ft Ring Road, Bengaluru – 560 085, Karnataka, India

Problem Statement:

An ATM (Automated Teller Machine) is an electronic device that allows bank account holders to access their accounts and carry out financial transactions such as withdrawals, deposits, and balance inquiries, without the need for a bank teller or cashier.

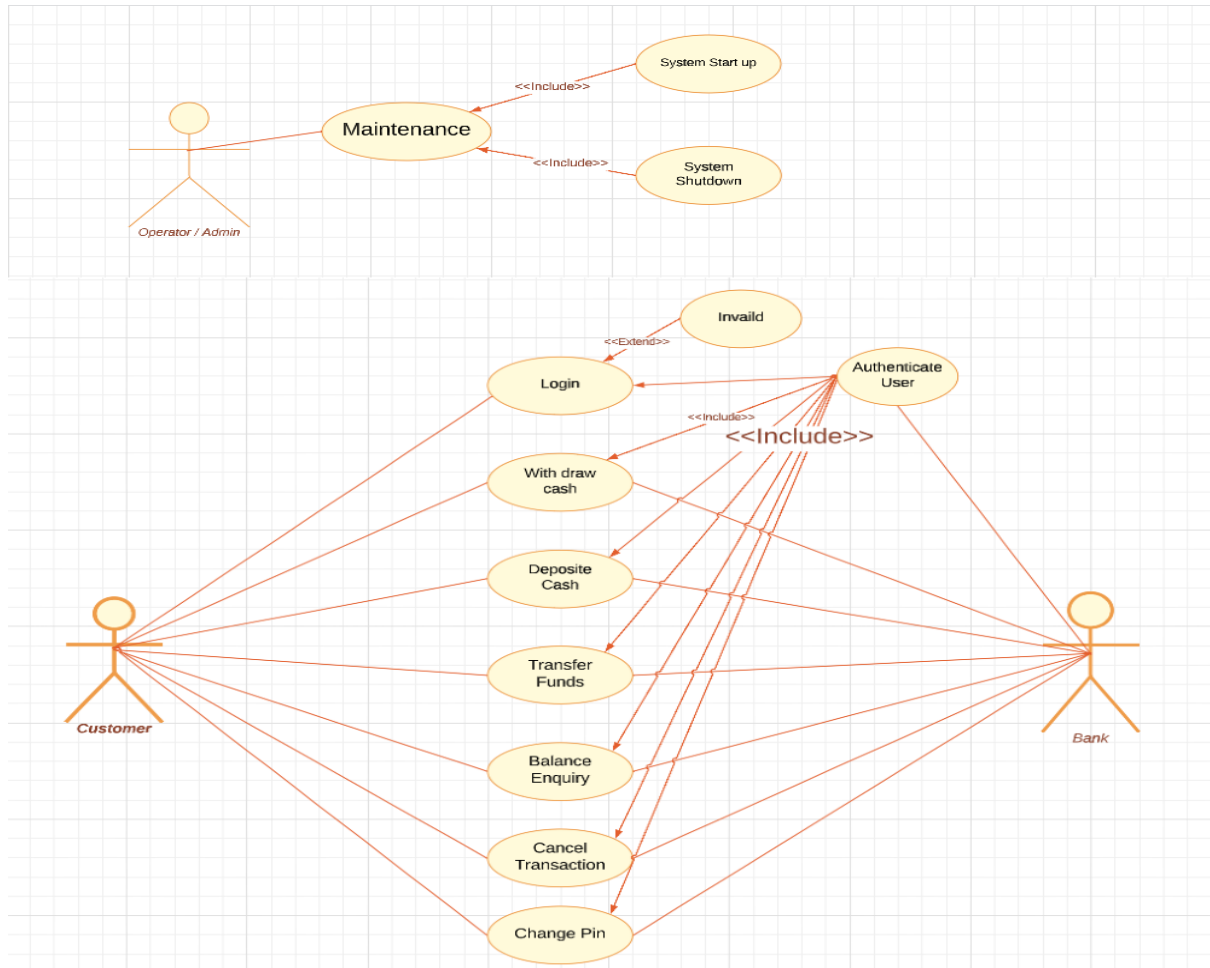
The objective of this project is to develop an ATM machine application using Java and following the MVC (Model-View-Controller) architecture. The application will allow bank account holders to perform various transactions by interacting with the user interface of the ATM machine. The features include the balance enquiry , cash withdrawal ,Deposit the money and also transfer the money to the other account.

Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

Models:

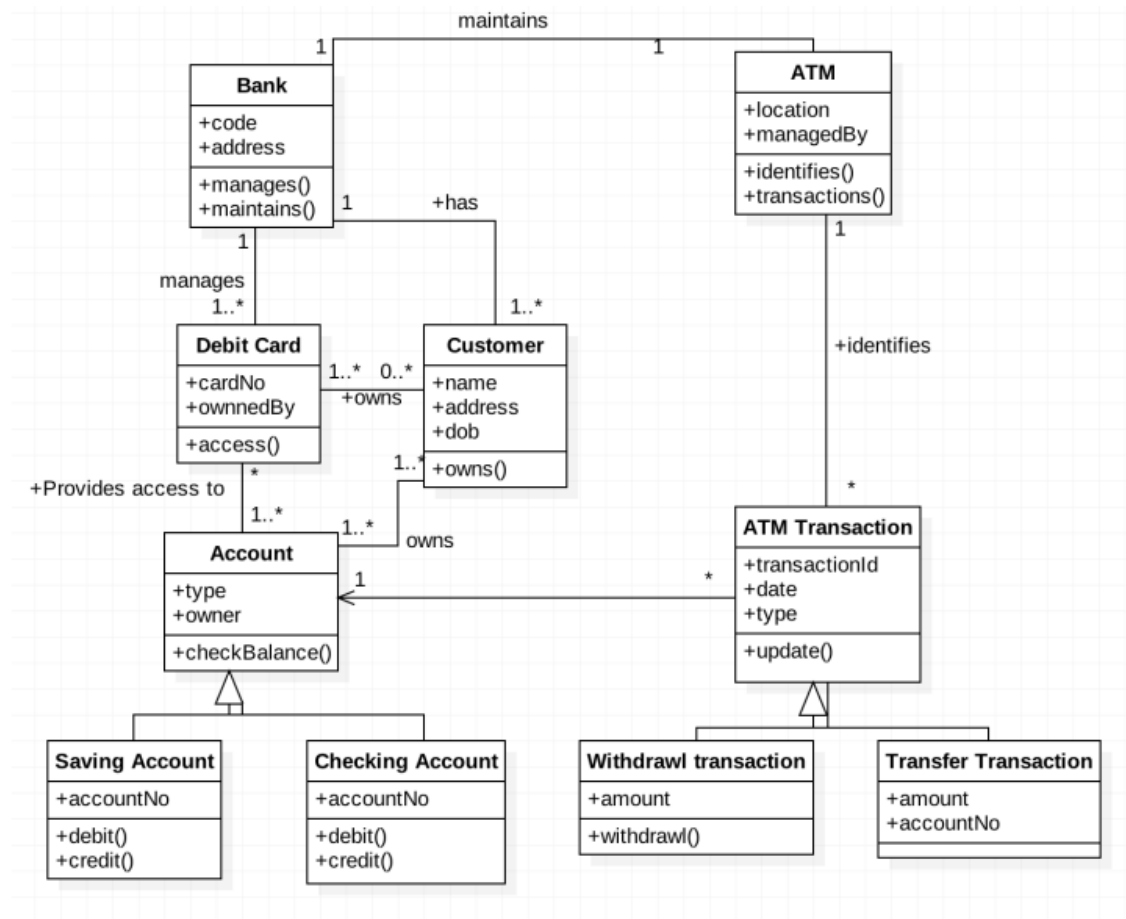
Use Case Model:



Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

Class Models:



Architectural patterns:

Design Principles and Design Patterns:

Based on the provided code files, the following GRASP and SOLID design principles are likely used in the following parts of the code:

1. High Cohesion:

- **TransactionService** class: It encapsulates the business logic for handling transactions, including saving transactions and retrieving transaction data from the repository.

Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

- AccountService class: It encapsulates the business logic for handling accounts, including creating accounts, retrieving account data, and updating account balances.

2. Low Coupling:

- TransactionService, AccountService, TransactionRepository, and AccountRepository classes: They rely on abstractions (interfaces) and are injected using the @Autowired annotation, allowing for flexibility in swapping out different implementations and reducing dependencies between classes.
- ATMController class: It receives HTTP requests from users and coordinates the appropriate actions in the system, but it does not contain business logic or interact directly with the repositories.

3. Creator:

- StartWebApplication class: It creates and runs the Spring Boot application using the SpringApplication.run() method, responsible for creating and initializing the application context.

4. Controller:

- ATMController class: It appears to handle user interactions with the ATM machine, receiving HTTP requests and coordinating the appropriate actions in the system.

5. Single Responsibility Principle (SRP):

- TransactionService class: It focuses on the business logic for handling transactions.
- AccountService class: It focuses on the business logic for handling accounts.
- TransactionRepository and AccountRepository classes: They are responsible for data access and manipulation for transactions and accounts, respectively.
- ATMController class: It handles user interactions with the ATM machine.

Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

6. Open/Closed Principle (OCP):

- The use of interfaces (abstractions) and dependency injection with @Autowired can promote extensibility and flexibility, allowing for new functionality to be added without modifying existing code. For example, if a new type of transaction or account is added, new classes implementing the relevant interfaces can be created without modifying the existing code.

7. Dependency Inversion Principle (DIP):

- The use of interfaces (abstractions) and dependency injection with @Autowired in the TransactionService, AccountService, TransactionRepository, and AccountRepository classes can promote loose coupling and flexibility in swapping out different implementations, adhering to the DIP.

Github link to the Code base

https://github.com/Rajesh-mm/ATM_Machine.git

Individual contributions of the team members:

Rohith H has implemented Transaction History and Deposit.

Sagar S has implemented Login and Change Pin.

Rajesh has implemented Withdraw and Transfer

Ravi Kiran has implemented Checking Balance, System Startup and Shutdown.

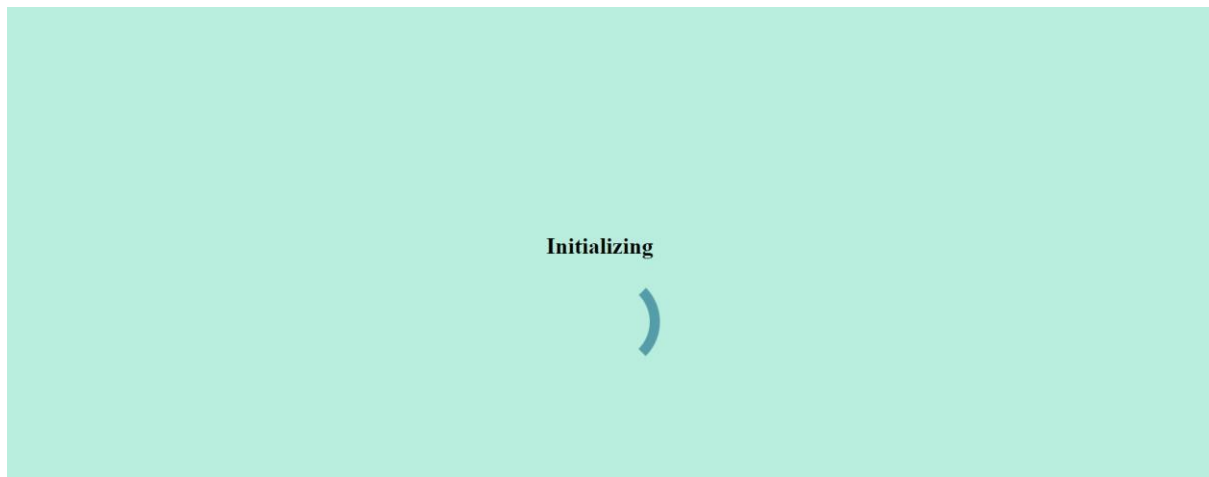
Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

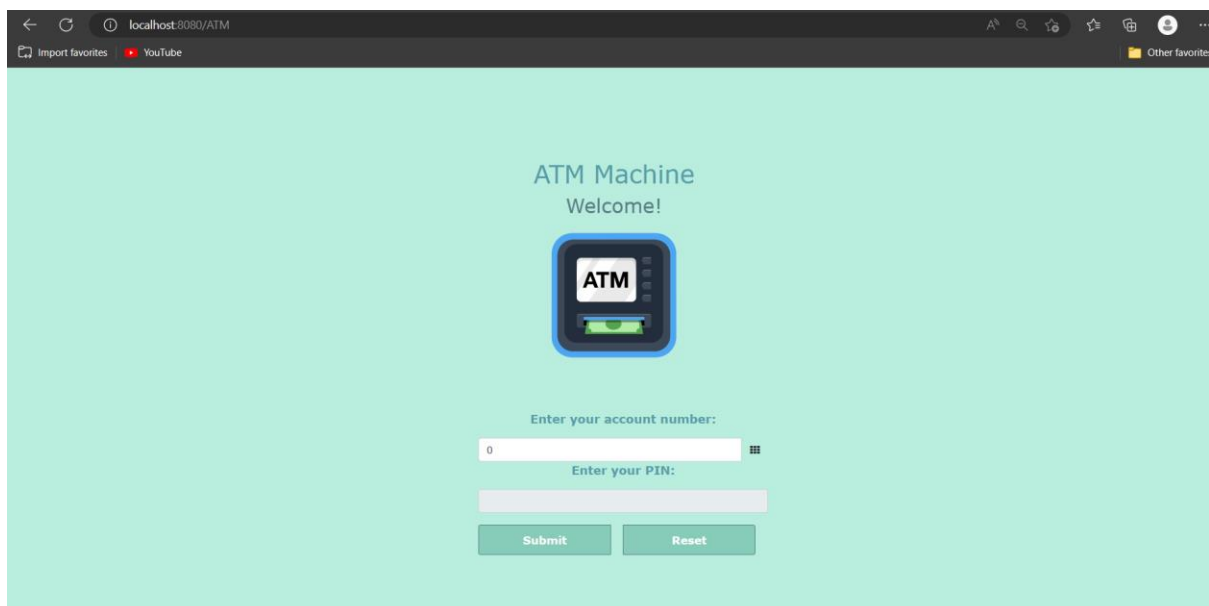
Screenshots with input values populated and output shown:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
tool.schema.internal.exec.ScriptSourceInputNonExistentImpl@1b275f5b'
2023-04-25 00:21:46.479 INFO 6984 --- [ restartedMain] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2023-04-25 00:21:46.509 INFO 6984 --- [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2023-04-25 00:21:47.226 INFO 6984 --- [ restartedMain] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2023-04-25 00:21:47.320 WARN 6984 --- [ restartedMain] aWebConfiguration$JpaWebMvcConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2023-04-25 00:21:48.257 INFO 6984 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2023-04-25 00:21:48.266 INFO 6984 --- [ restartedMain] com.atm.StartWebApplication : Started StartWebApplication in 7.486 seconds (JVM running for 8.296)
```

Startup Page:



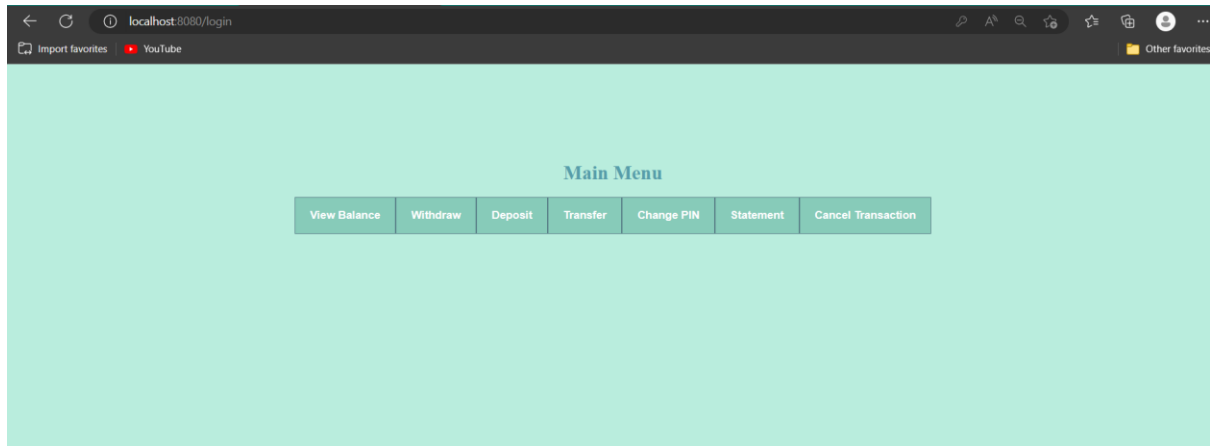
Login Page:



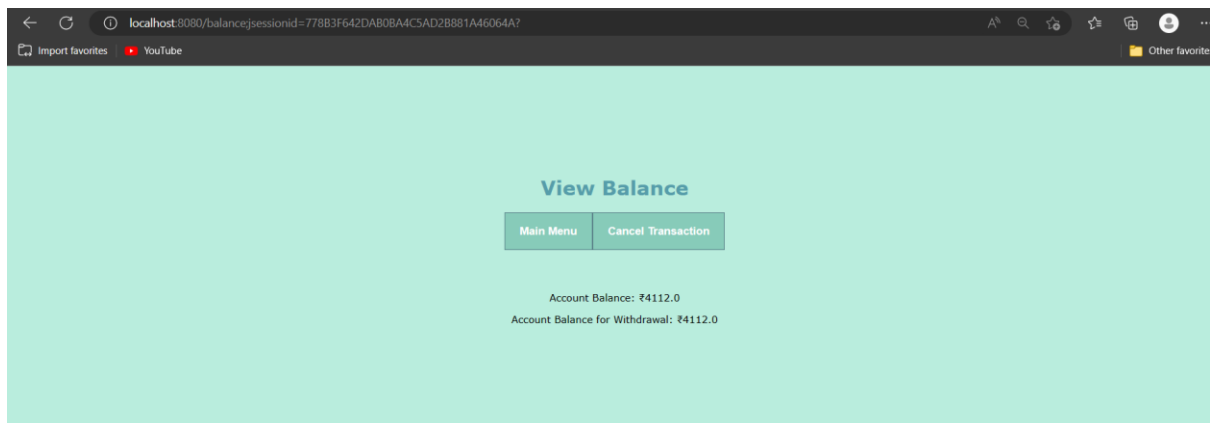
Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

Menu Page:



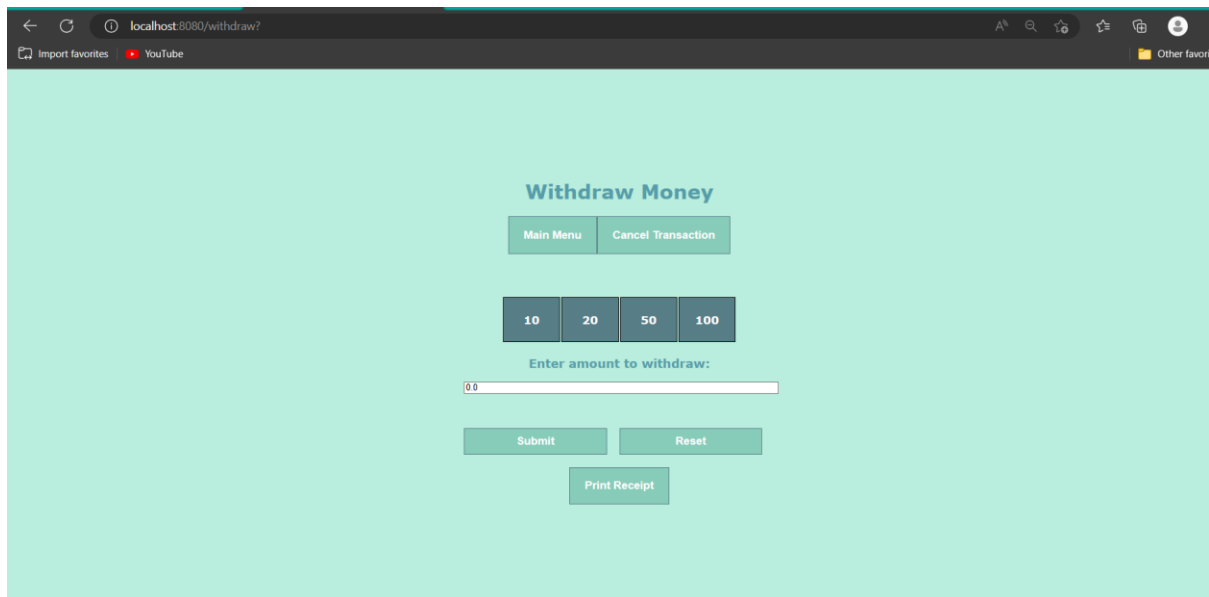
Balance Page:



Object Oriented Analysis and Design using Java (UE20CS352)

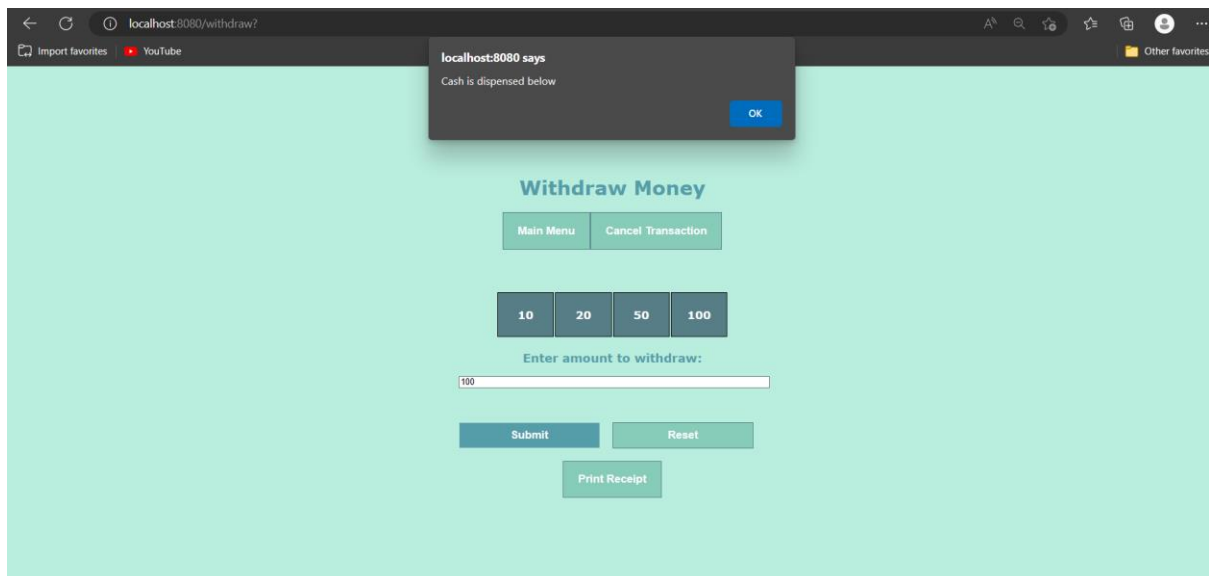
MINI PROJECT REPORT

Withdraw Page:



The screenshot shows a web browser window with the URL `localhost:8080/withdraw?`. The page has a light green background and a title "Withdraw Money" in blue. Below the title are two buttons: "Main Menu" and "Cancel Transaction". There are four buttons for denominations: "10", "20", "50", and "100". Below these is a text input field labeled "Enter amount to withdraw:" with a value of "0.0". At the bottom are three buttons: "Submit", "Reset", and "Print Receipt".

When we click on submit it will show message as “Cash is dispensed below”:



The screenshot shows the same web browser window as before, but with a message box displayed. The message box is titled "localhost:8080 says" and contains the text "Cash is dispensed below". There is an "OK" button in the bottom right corner of the message box. The background page is the same "Withdraw Money" form, but the "Enter amount to withdraw:" field now shows a value of "100".

Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

Withdrew 100 rupees:

Withdraw Money

Main Menu Cancel Transaction

10 20 50 100

Enter amount to withdraw:

100.0

You withdrew ₹100.0

Submit Reset

Print Receipt

When we click on Print Receipt showing balance and account number:

localhost:8080 says

Account number: 23456

Balance: ₹4012

OK

Withdraw Money

Main Menu Cancel Transaction

10 20 50 100

Enter amount to withdraw:

100.0

You withdrew ₹100.0

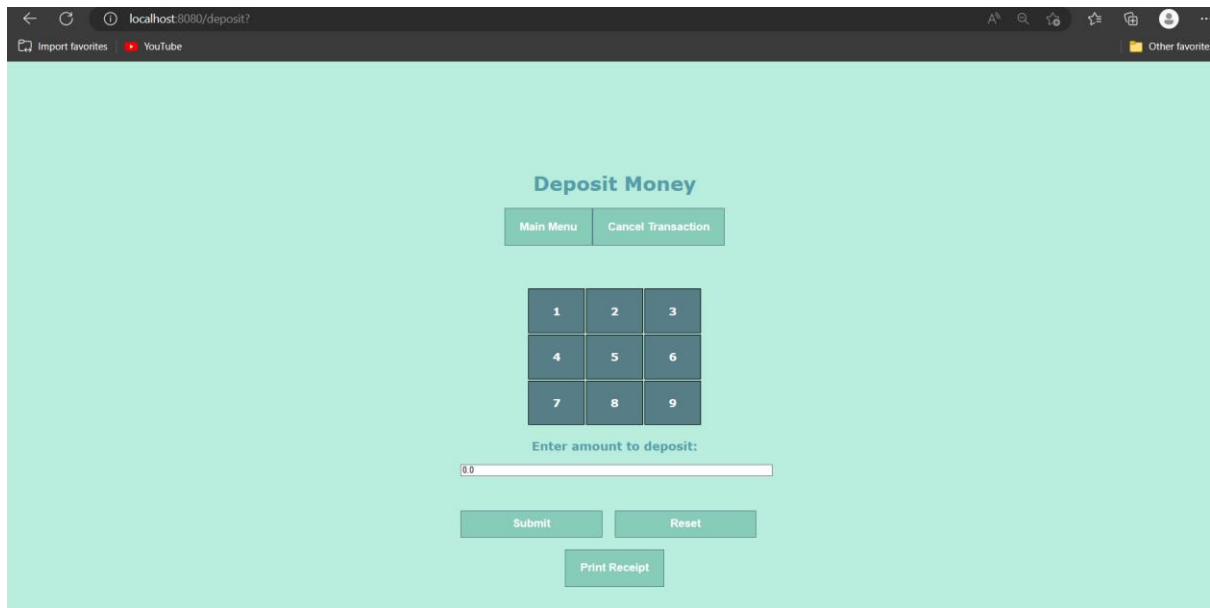
Submit Reset

Print Receipt

Object Oriented Analysis and Design using Java (UE20CS352)

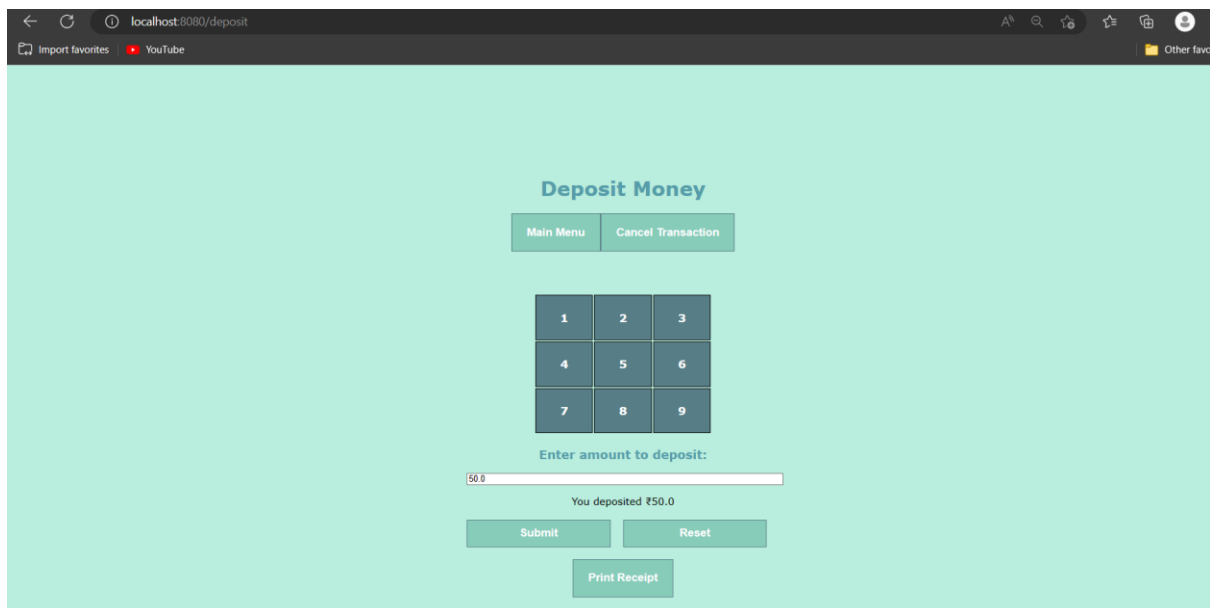
MINI PROJECT REPORT

Deposit Page:



The screenshot shows a web browser window with the address bar displaying "localhost:8080/deposit?". The page has a light green background and is titled "Deposit Money". At the top, there are two buttons: "Main Menu" and "Cancel Transaction". Below these is a numeric keypad with buttons for digits 1 through 9. Under the keypad is a text input field labeled "Enter amount to deposit:" with the value "0.0". At the bottom, there are three buttons: "Submit", "Reset", and "Print Receipt".

Deposited 50 rupees:



The screenshot shows the same web browser window as before, but the text input field now contains "50.0". Below the input field, a confirmation message reads "You deposited ₹50.0". The "Print Receipt" button is now highlighted with a green border. The other elements, including the "Main Menu", "Cancel Transaction", numeric keypad, and "Submit"/"Reset" buttons, remain the same.

Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

When we click on Print Receipt showing balance and account number:

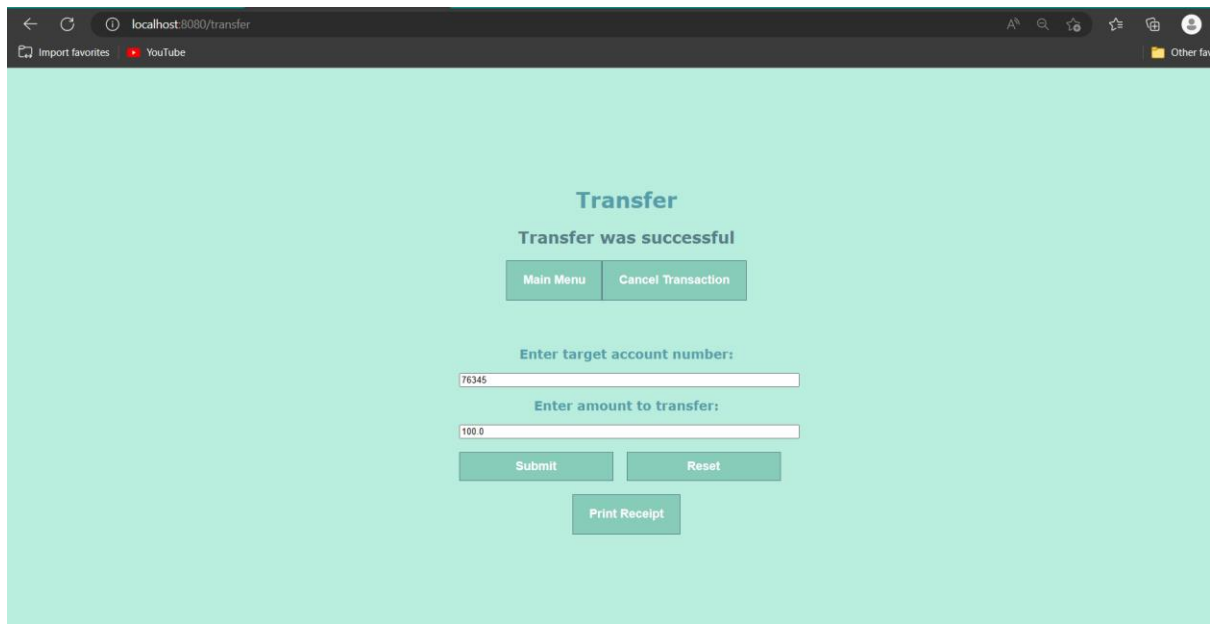
The screenshot shows a web browser at the URL `localhost:8080/deposit`. A modal dialog box is displayed in the center, titled "localhost:8080 says", containing the text "Account number: 23456" and "Balance: ₹4062", with an "OK" button. The background page has a light green background and is titled "Deposit Money". It features two buttons: "Main Menu" and "Cancel Transaction". Below these is a numeric keypad with buttons 1 through 9. Under the keypad is a text input field labeled "Enter amount to deposit:" containing the value "50.0". Below the input field, it says "You deposited ₹50.0". At the bottom are three buttons: "Submit", "Reset", and "Print Receipt".

Transfer Page:

The screenshot shows a web browser at the URL `localhost:8080/transfer?`. The page has a light green background and is titled "Transfer". It features two buttons: "Main Menu" and "Cancel Transaction". Below these are two text input fields. The first is labeled "Enter target account number:" and contains the value "0". The second is labeled "Enter amount to transfer:" and contains the value "0.0". At the bottom are three buttons: "Submit", "Reset", and "Print Receipt".

Object Oriented Analysis and Design using Java (UE20CS352)

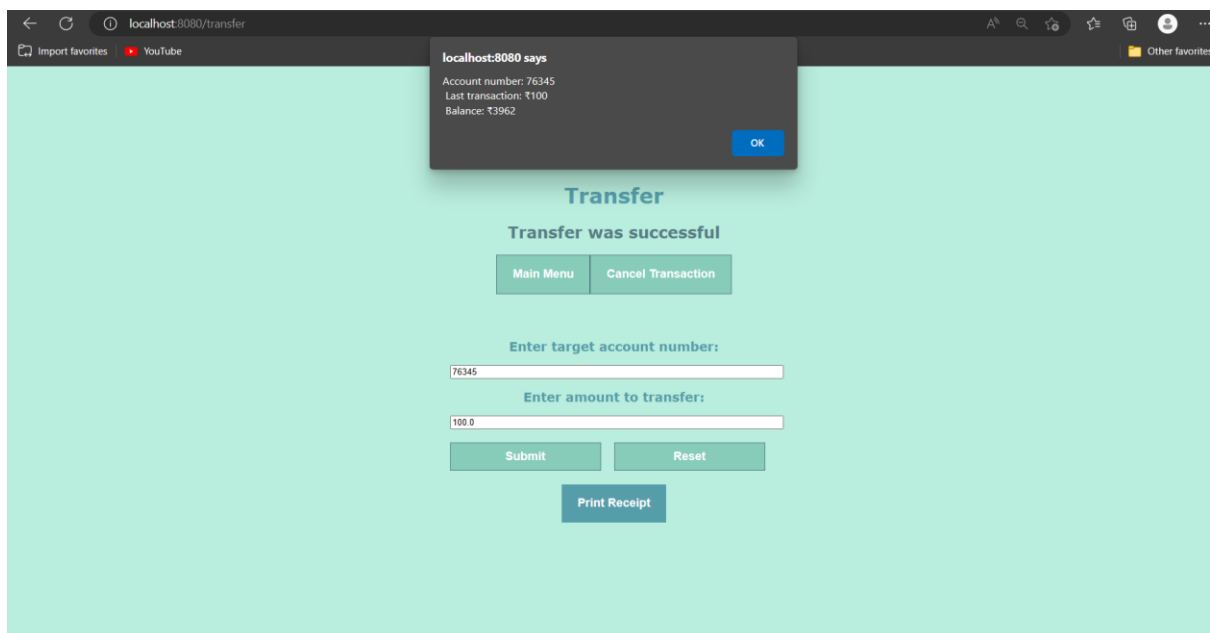
MINI PROJECT REPORT



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/transfer'. The page has a light green background and contains the following elements:

- Transfer** (Section Header)
- Transfer was successful** (Status Message)
- Two buttons: **Main Menu** and **Cancel Transaction**
- Enter target account number:** Input field containing '76345'
- Enter amount to transfer:** Input field containing '100.0'
- Two buttons: **Submit** and **Reset**
- Print Receipt** button

When we click on Print Receipt Showing balance and account number:



The screenshot shows the same web application as before, but with a modal dialog box open. The dialog box has a dark gray background and contains the following text:

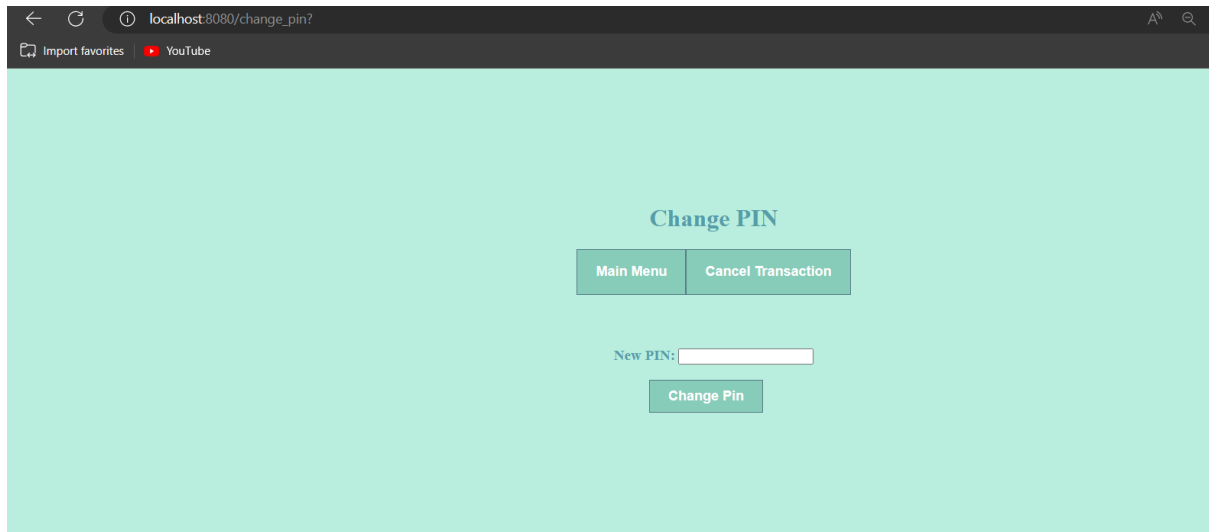
- localhost:8080 says**
- Account number: 76345
- Last transaction: ₹100
- Balance: ₹3962
- OK** button

The background page remains the same, showing the 'Transfer' section and the 'Print Receipt' button.

Object Oriented Analysis and Design using Java (UE20CS352)

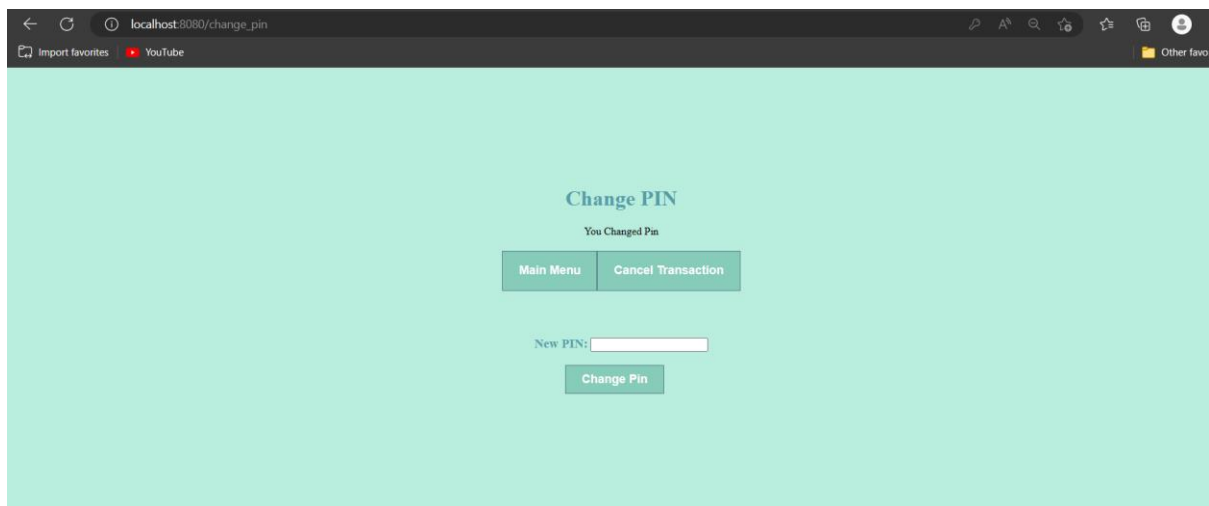
MINI PROJECT REPORT

Change PIN Page:



A screenshot of a web browser displaying the 'Change PIN' page. The browser's address bar shows 'localhost:8080/change_pin?'. The page has a light green background. At the top center, the text 'Change PIN' is displayed in a dark blue font. Below this, there are two buttons: 'Main Menu' and 'Cancel Transaction', both in a light green box. Further down, the text 'New PIN:' is followed by a white input field. Below the input field is a 'Change Pin' button in a light green box.

Pin changed from 1212 to 4567

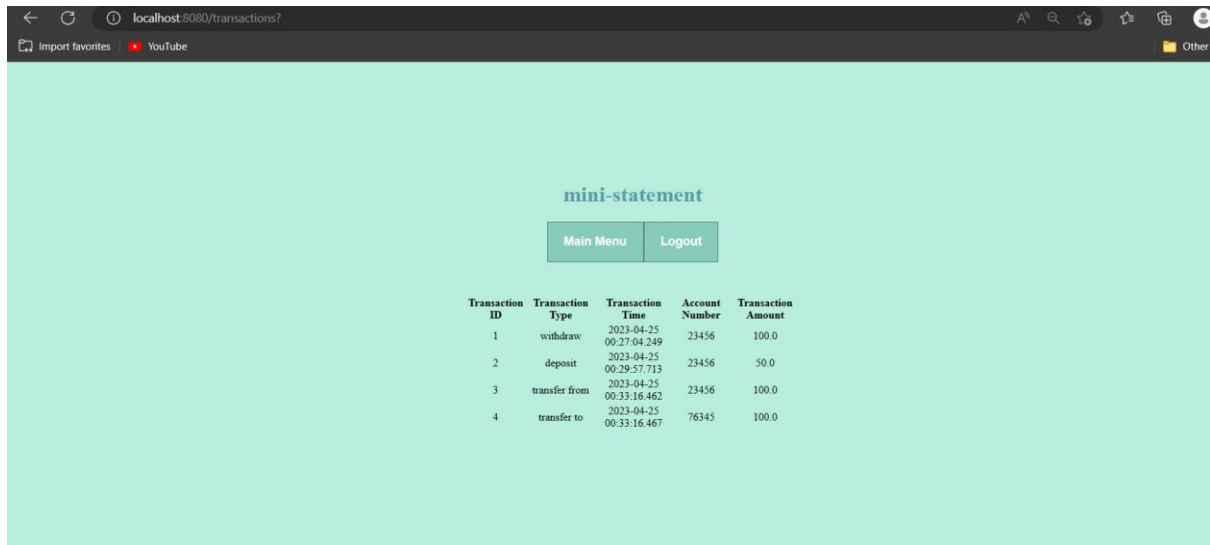


A screenshot of the same web browser displaying the 'Change PIN' page. The browser's address bar shows 'localhost:8080/change_pin'. The page has a light green background. At the top center, the text 'Change PIN' is displayed in a dark blue font. Below this, the text 'You Changed Pin' is displayed in a smaller dark blue font. Below this, there are two buttons: 'Main Menu' and 'Cancel Transaction', both in a light green box. Further down, the text 'New PIN:' is followed by a white input field. Below the input field is a 'Change Pin' button in a light green box.

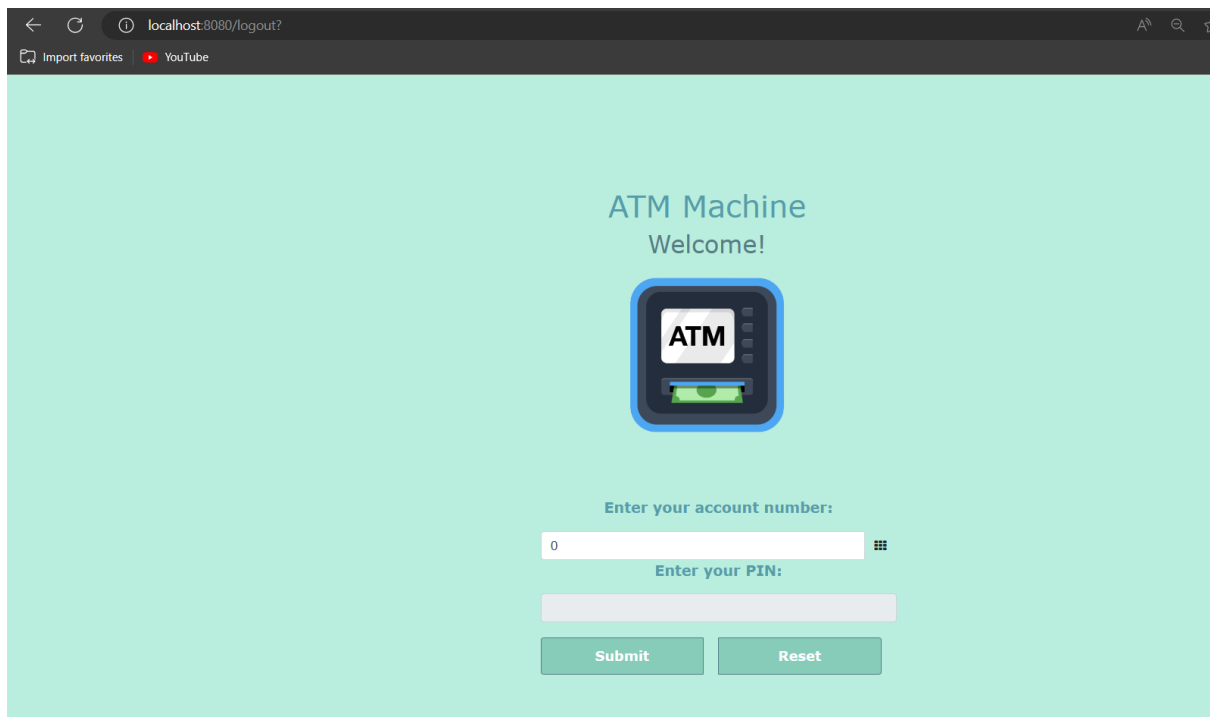
Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

Transaction History or Mini-Statement Page:



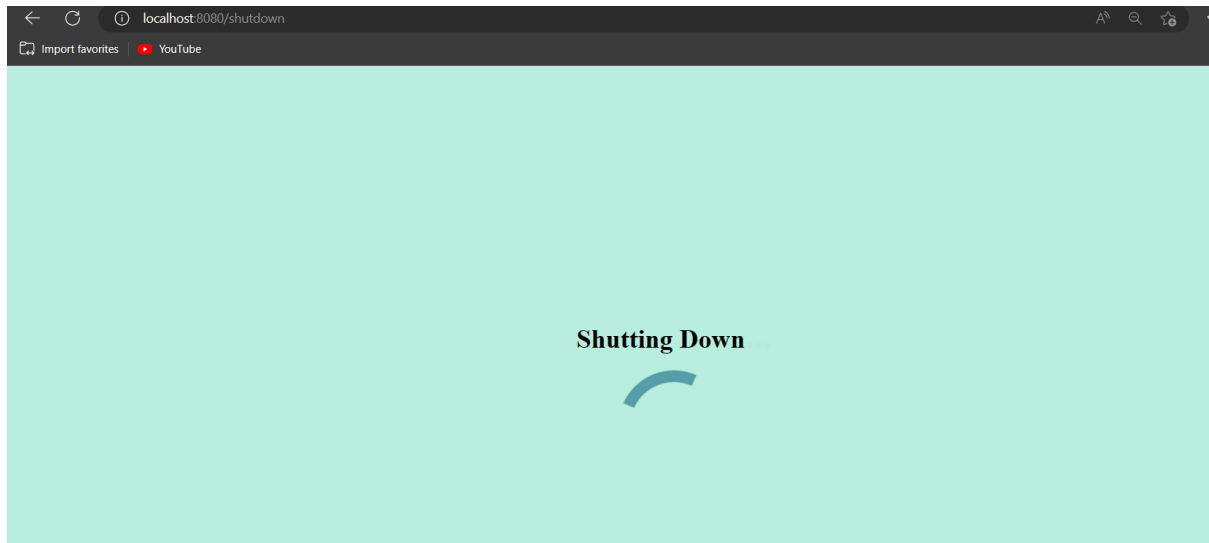
After clicking cancel Transaction:



Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

After 10 minutes when not in use it will shut down:



H2 Database:

Accounts Table:

A screenshot of the H2 Database console. The left sidebar shows the database structure: jdbc:h2:mem:testdb, ACCOUNTS, TRANSACTIONS, USERS, INFORMATION_SCHEMA, Sequences, and Users. The main area shows the SQL statement 'SELECT * FROM ACCOUNTS' and the resulting table data.

ACCOUNT_NO	PIN	ACCOUNT_TYPE	BALANCE	USER_ID	LOCKED	INCORRECT_ATTEMPTS
23456	4567	SAVINGS	3962.0	1	FALSE	0
76345	8001	CHECKING	5334.0	2	FALSE	0

(2 rows, 102 ms)

Edit

Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

Transaction Table:

The screenshot shows the H2 console interface. The left sidebar displays the database structure: jdbc:h2:mem:testdb, ACCOUNTS, TRANSACTIONS, USERS, INFORMATION_SCHEMA, Sequences, and Users. The main area shows the SQL statement 'SELECT * FROM TRANSACTIONS' and its results. The results are displayed in a table with 5 columns: TRANSACTION_ID, TYPE, TRANSACTION_TIME, ACCOUNT_NO, and AMOUNT. There are 4 rows of data.

TRANSACTION_ID	TYPE	TRANSACTION_TIME	ACCOUNT_NO	AMOUNT
1	withdraw	2023-04-25 00:27:04.249	23456	100.0
2	deposit	2023-04-25 00:29:57.713	23456	50.0
3	transfer from	2023-04-25 00:33:16.462	23456	100.0
4	transfer to	2023-04-25 00:33:16.467	76345	100.0

(4 rows, 55 ms)

User Table:

The screenshot shows the H2 console interface. The left sidebar displays the database structure: jdbc:h2:mem:testdb, ACCOUNTS, TRANSACTIONS, USERS, INFORMATION_SCHEMA, Sequences, and Users. The main area shows the SQL statement 'SELECT * FROM USERS' and its results. The results are displayed in a table with 3 columns: ID, FIRST_NAME, and LAST_NAME. There are 4 rows of data.

ID	FIRST_NAME	LAST_NAME
1	Rohit	H
2	Sagar	S
3	Ravi	Kiran
4	Rajesh	MM

(4 rows, 14 ms)