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:

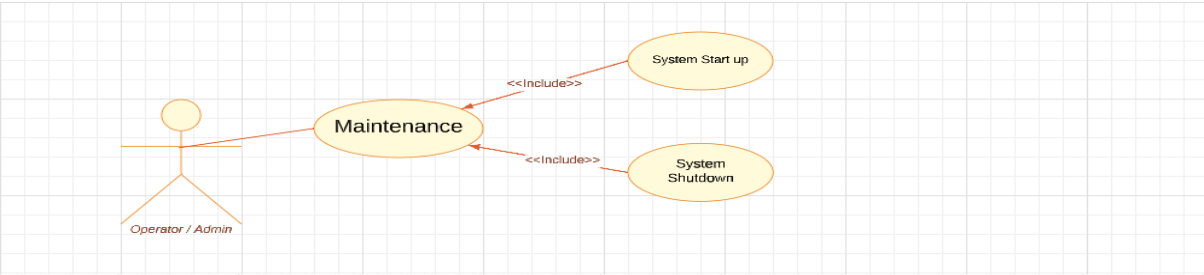
An ATM (Automated Teller Machine) is an electronic device that allows bank account holders to access their accounts and carry out financial transactions such as withdrawals, deposits, and balance inquiries, without the need for a bank teller or cashier.

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

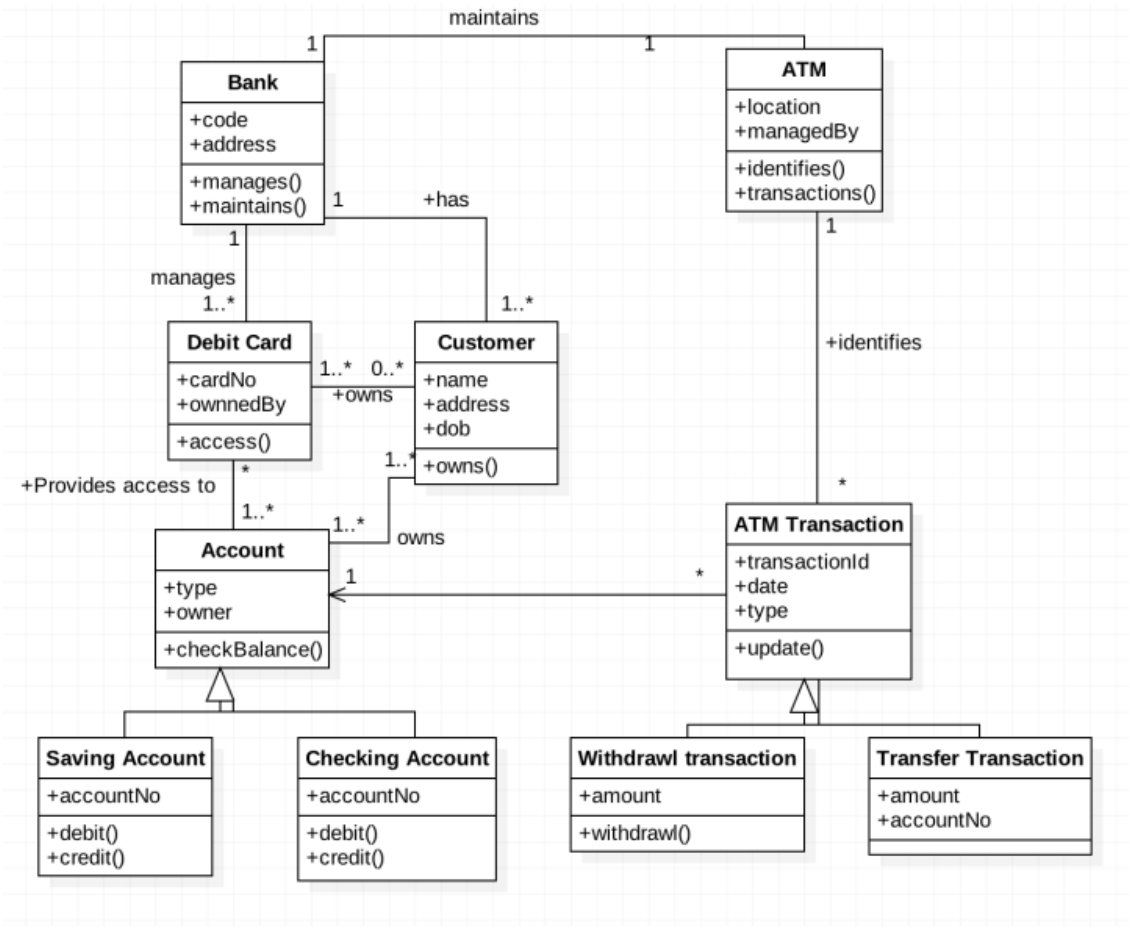
Models:

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.

- 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.

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.

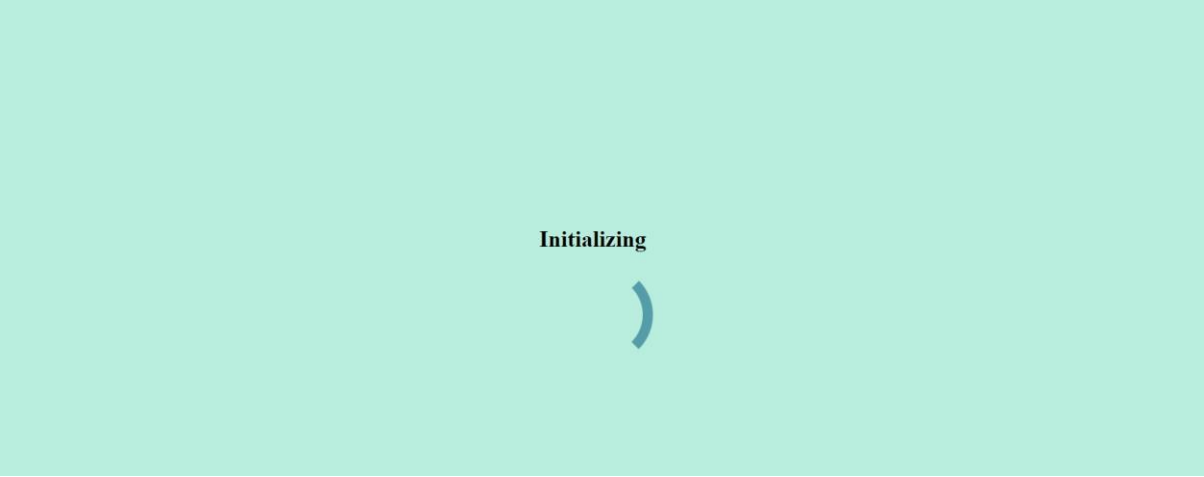
Object Oriented Analysis and Design using Java (UE20CS352)
MINI PROJECT REPORT

Screenshots with input values populated and output shown:

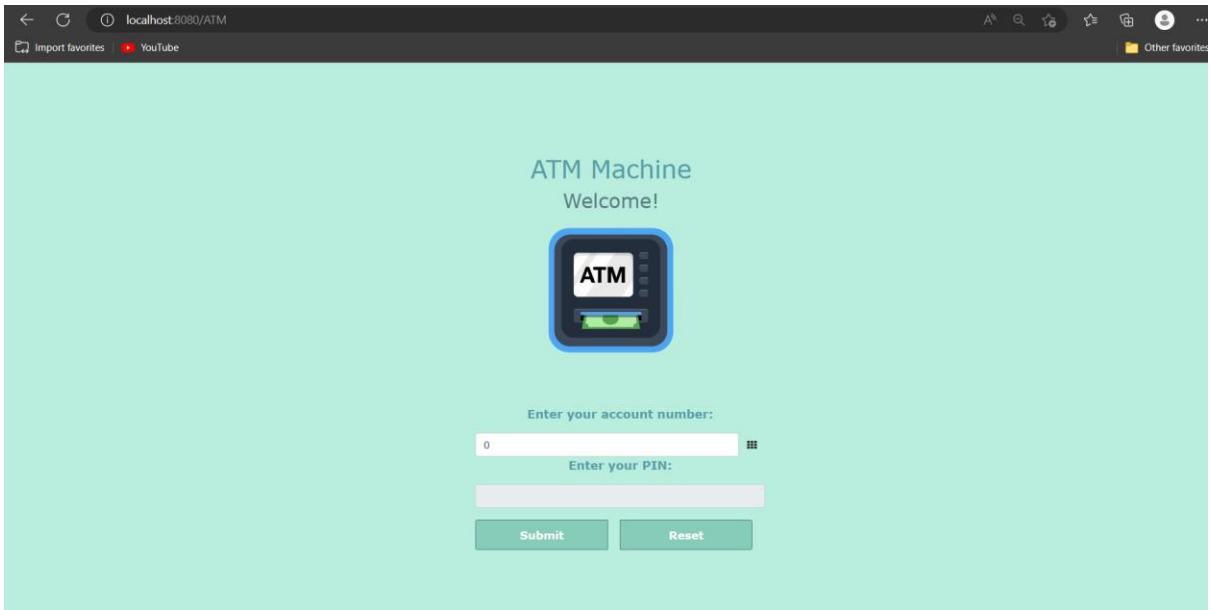
```
PROBLEMS 5   OUTPUT   DEBUG CONSOLE   TERMINAL                                   Run: StartWebApplication

tool.schema.internal.exec.ScriptSourceInputNonExistentImpl@1b275f5b'
2023-04-25 00:21:46.479  INFO 6984 --- [  restartedMain] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persisten
ce 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 'applicationTaskExecu
tor'
2023-04-25 00:21:47.320  WARN 6984 --- [  restartedMain] aWebConfiguration$JpaWebMvcConfiguration : spring.jpa.open-in-view is enabled by default. The
refore, 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 contex
t 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:

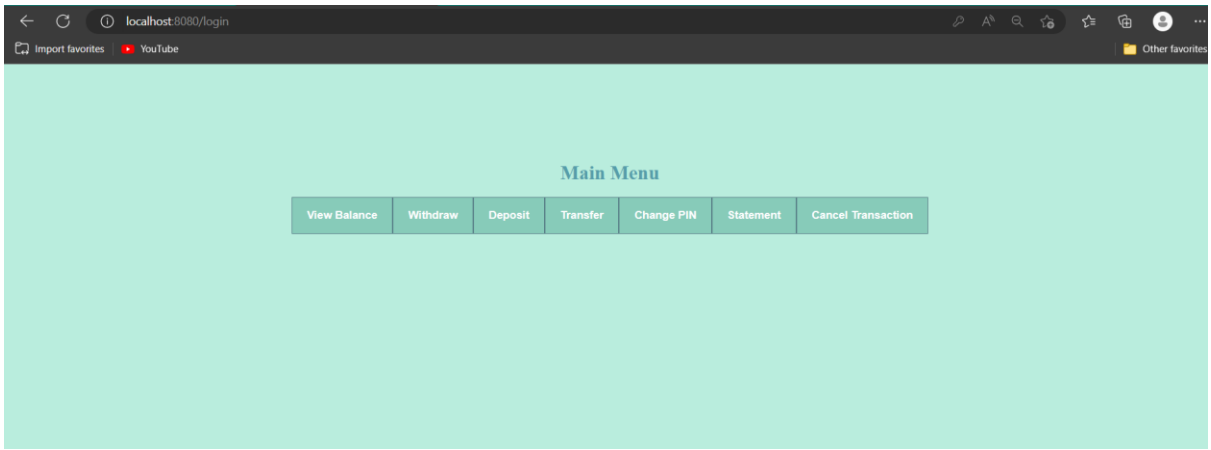


Login Page:

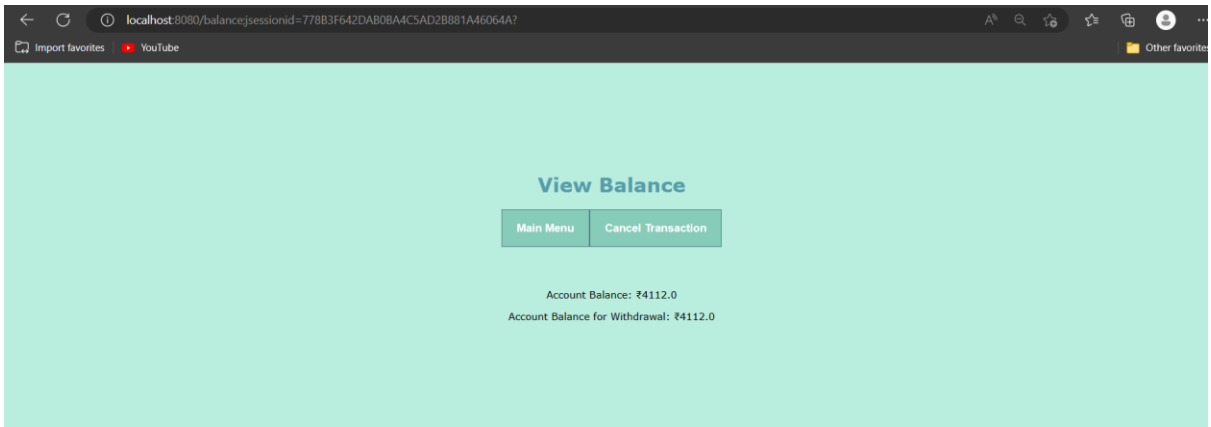


Object Oriented Analysis and Design using Java (UE20CS352)
MINI PROJECT REPORT

Menu Page:

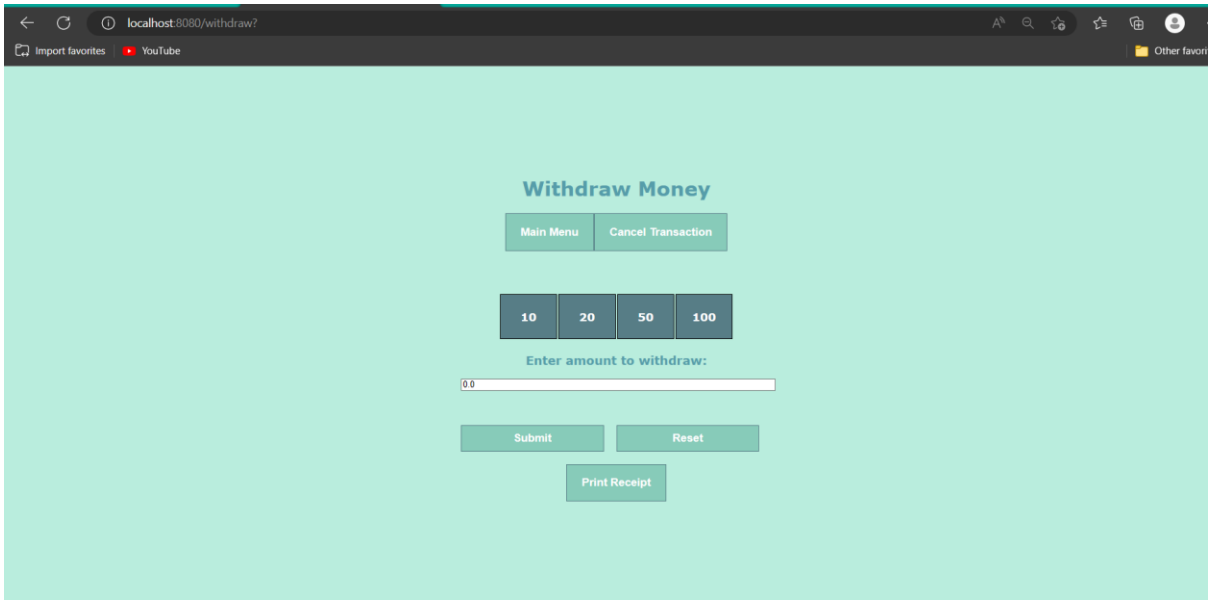


Balance Page:

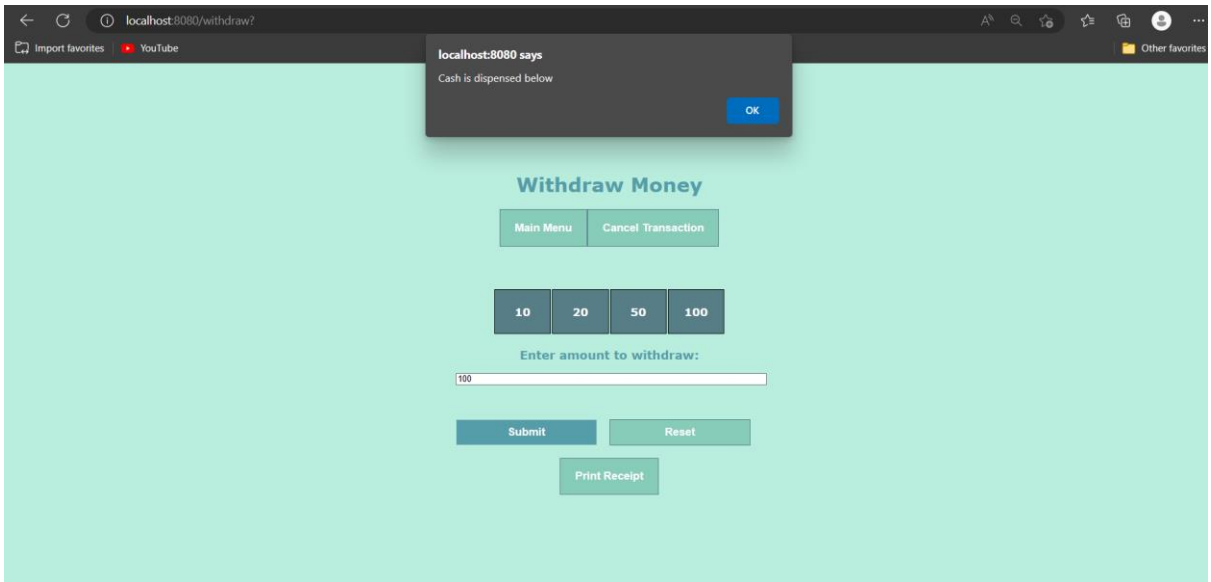


Object Oriented Analysis and Design using Java (UE20CS352)
MINI PROJECT REPORT

Withdraw Page:

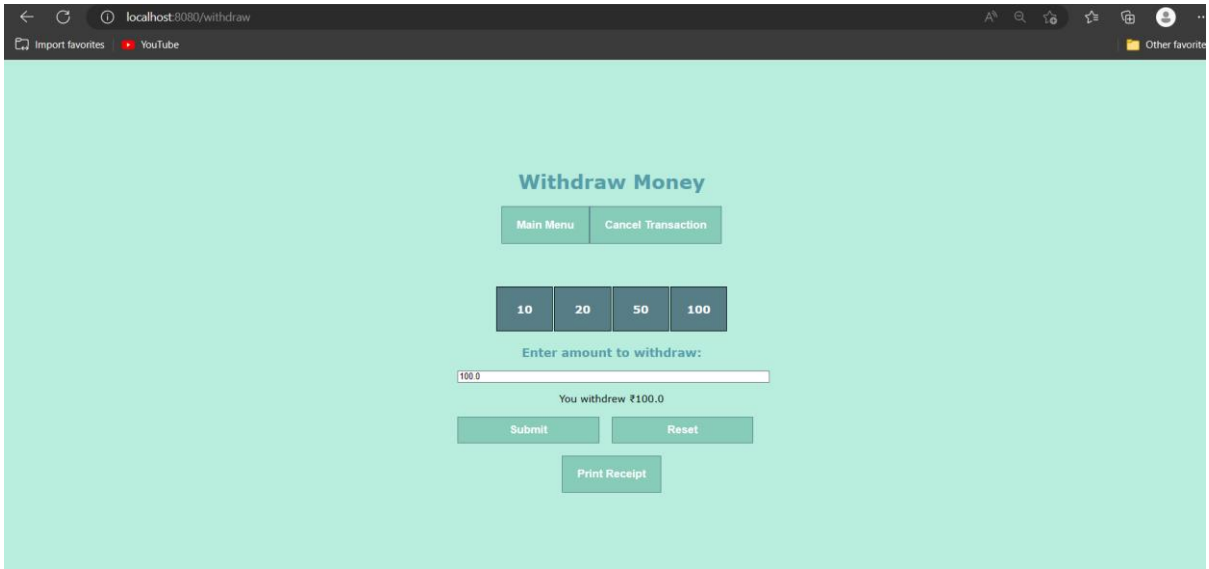


When we click on submit it will show message as “Cash is dispensed below”:

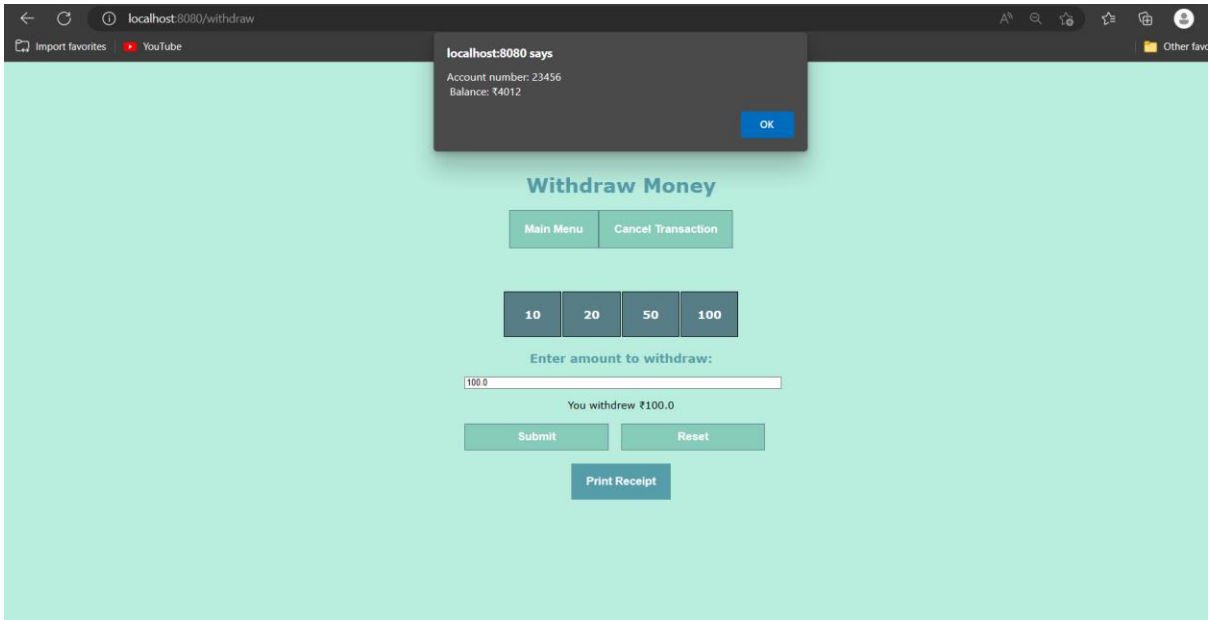


Object Oriented Analysis and Design using Java (UE20CS352)
MINI PROJECT REPORT

Withdrew 100 rupees:

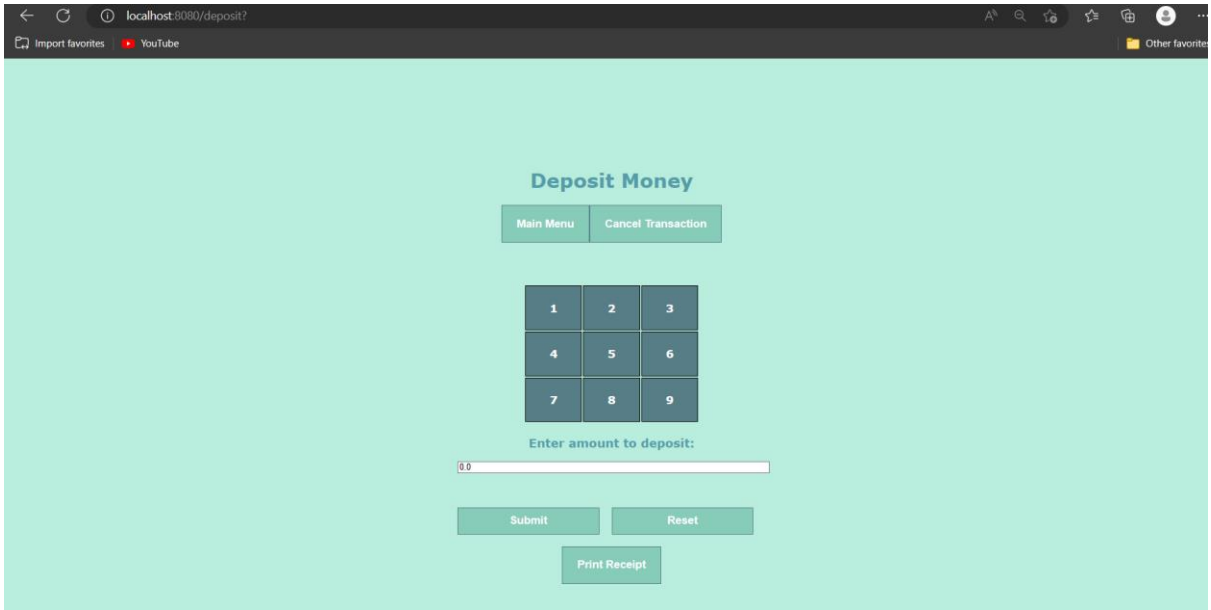


When we click on Print Receipt showing balance and account number:

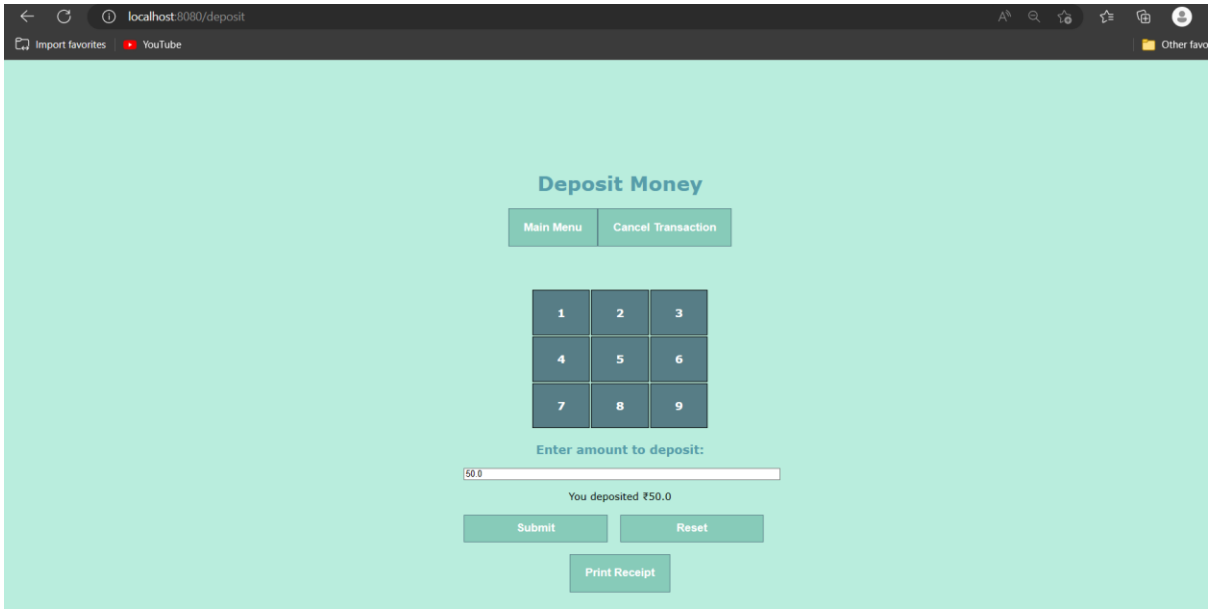


Object Oriented Analysis and Design using Java (UE20CS352)
MINI PROJECT REPORT

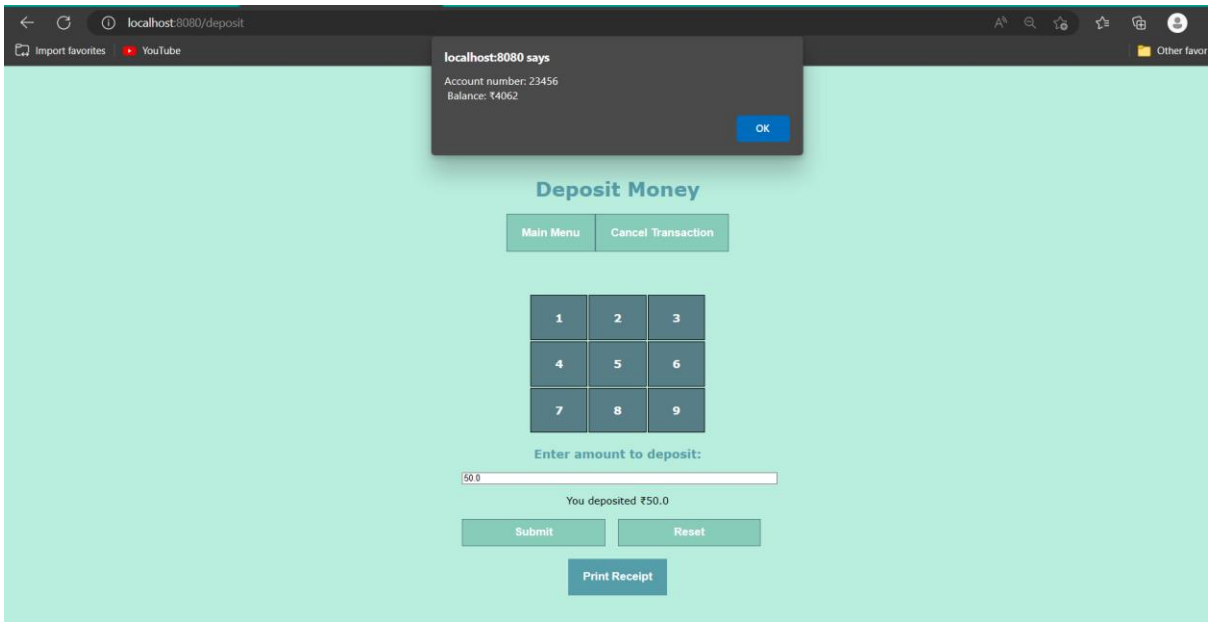
Deposit Page:



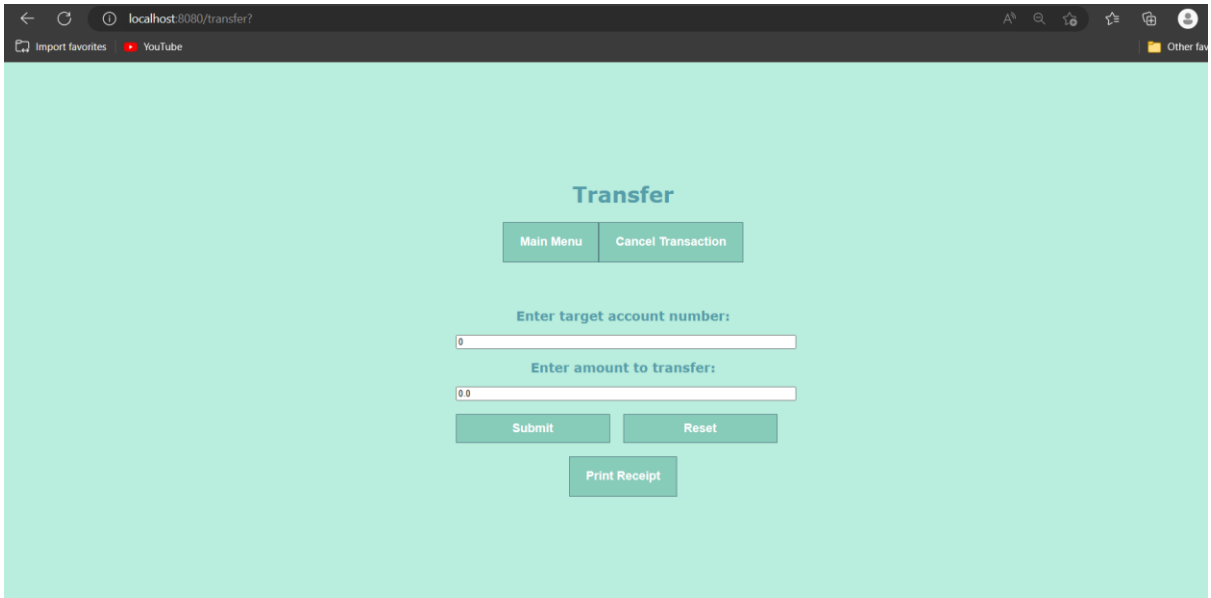
Deposited 50 rupees:



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

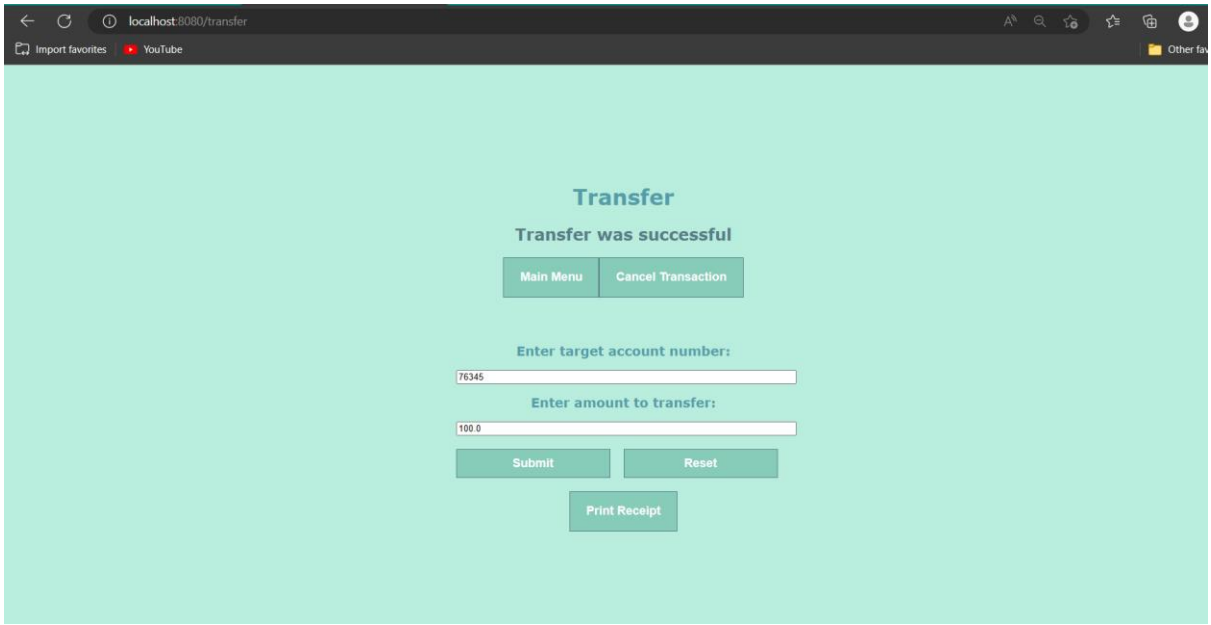


Transfer Page:

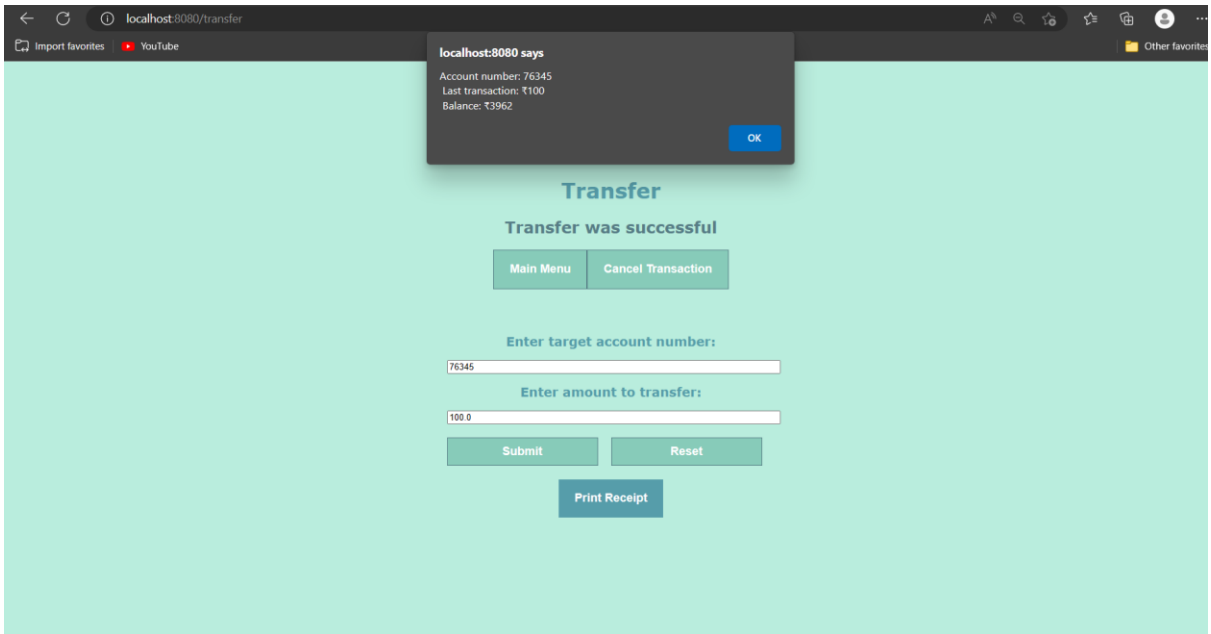


Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT



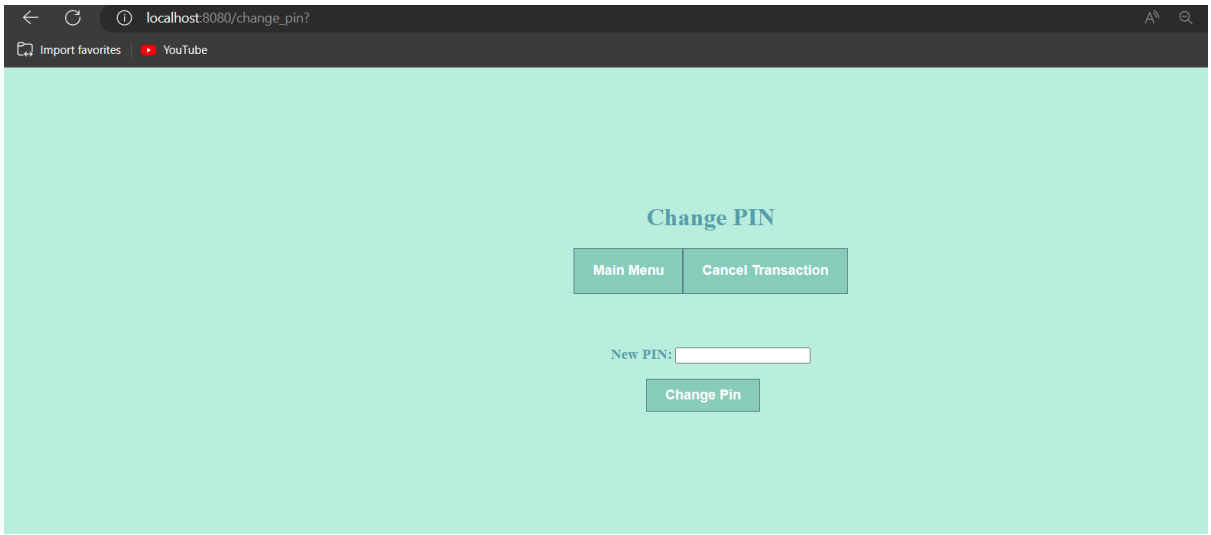
When we click on Print Receipt Showing balance and account number:



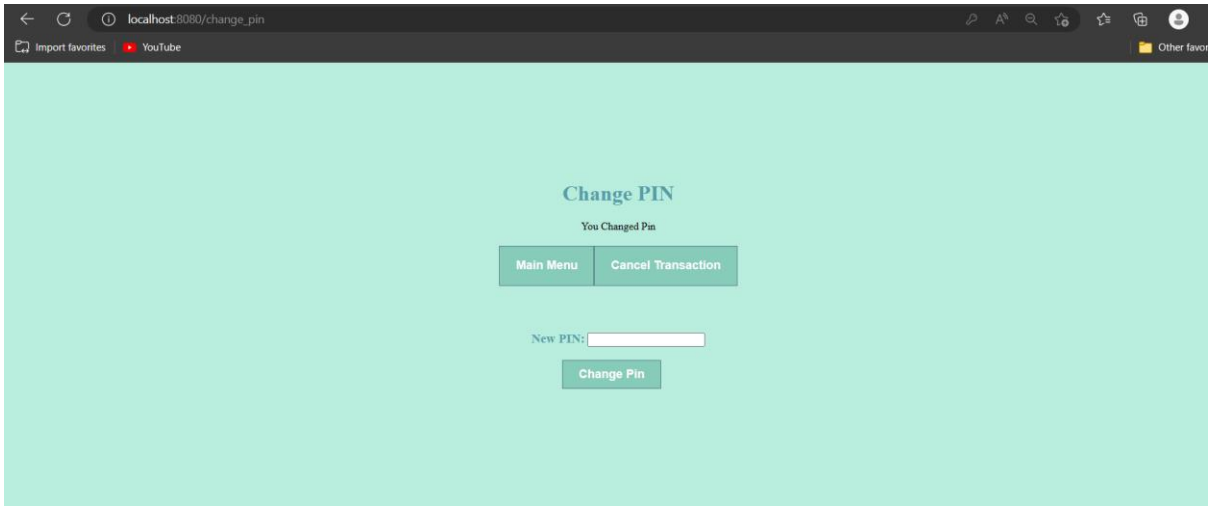
Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

Change PIN Page:



Pin changed from 1212 to 4567



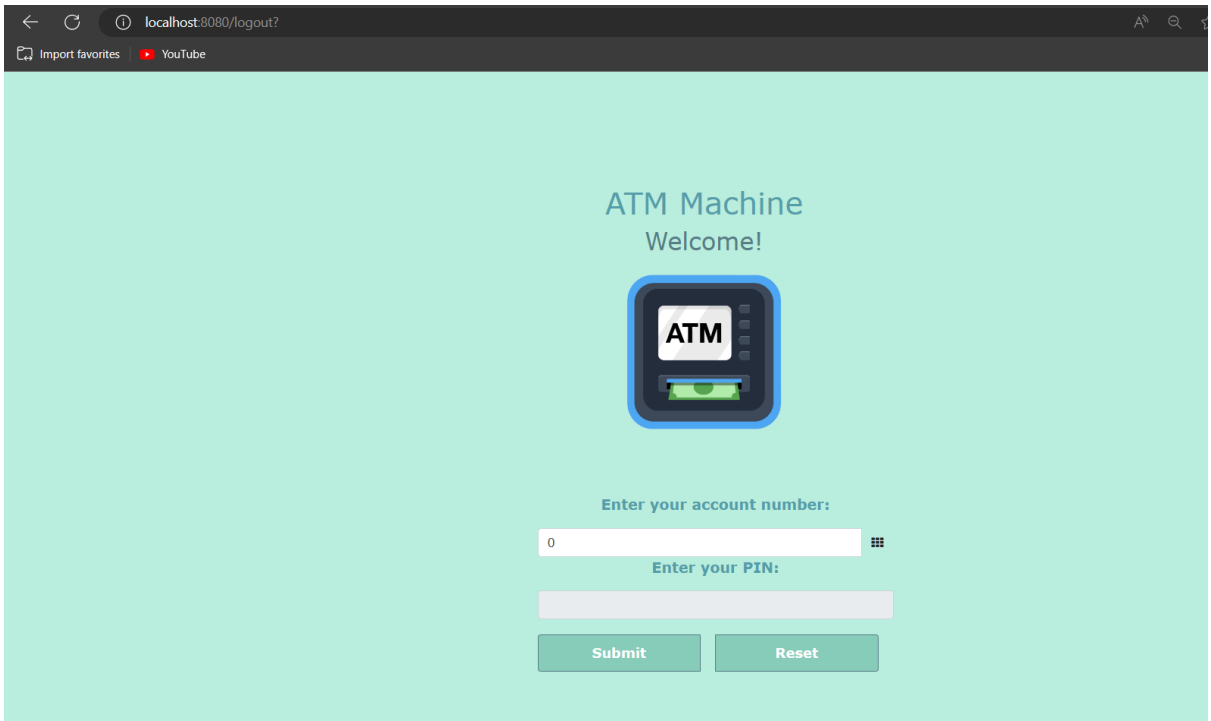
Object Oriented Analysis and Design using Java (UE20CS352)

MINI PROJECT REPORT

Transaction History or Mini-Statement Page:

Transaction ID	Transaction Type	Transaction Time	Account Number	Transaction 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

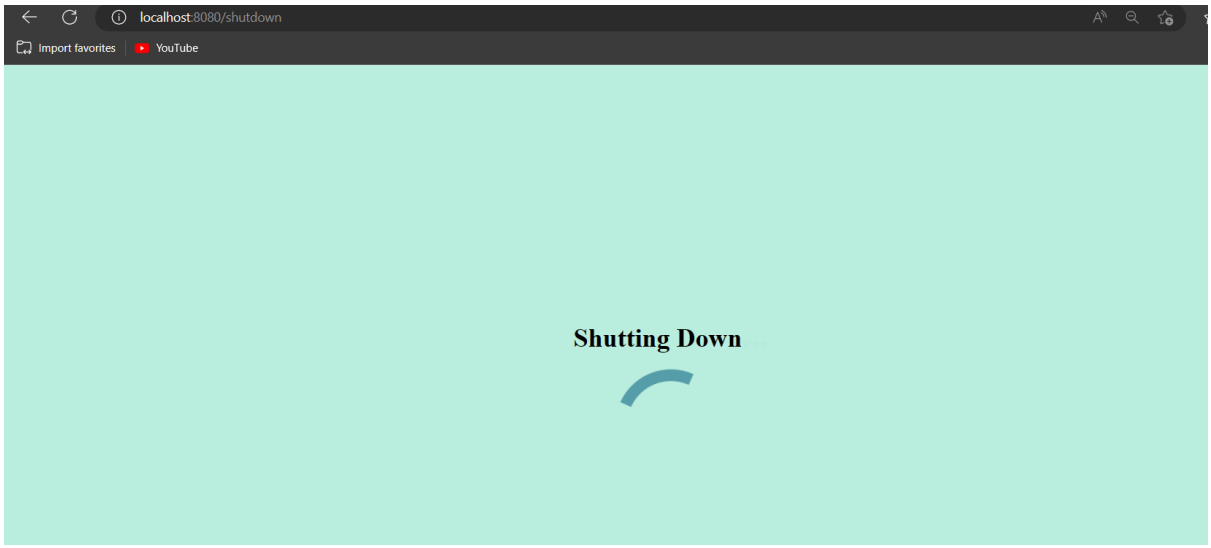
After clicking cancel Transaction:



Object Oriented Analysis and Design using Java (UE20CS352)

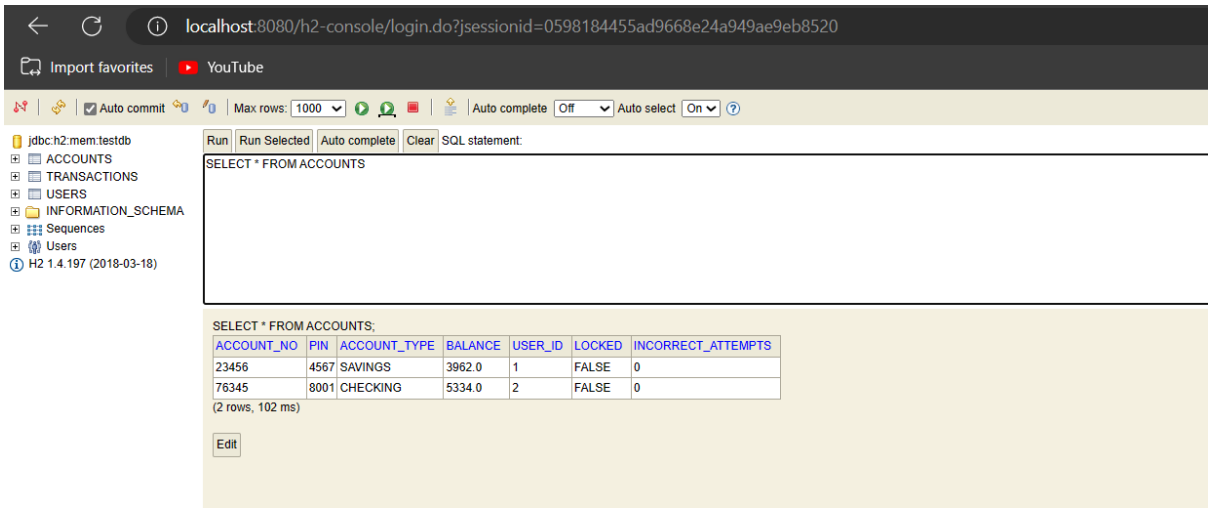
MINI PROJECT REPORT

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



H2 Database:

Accounts Table:

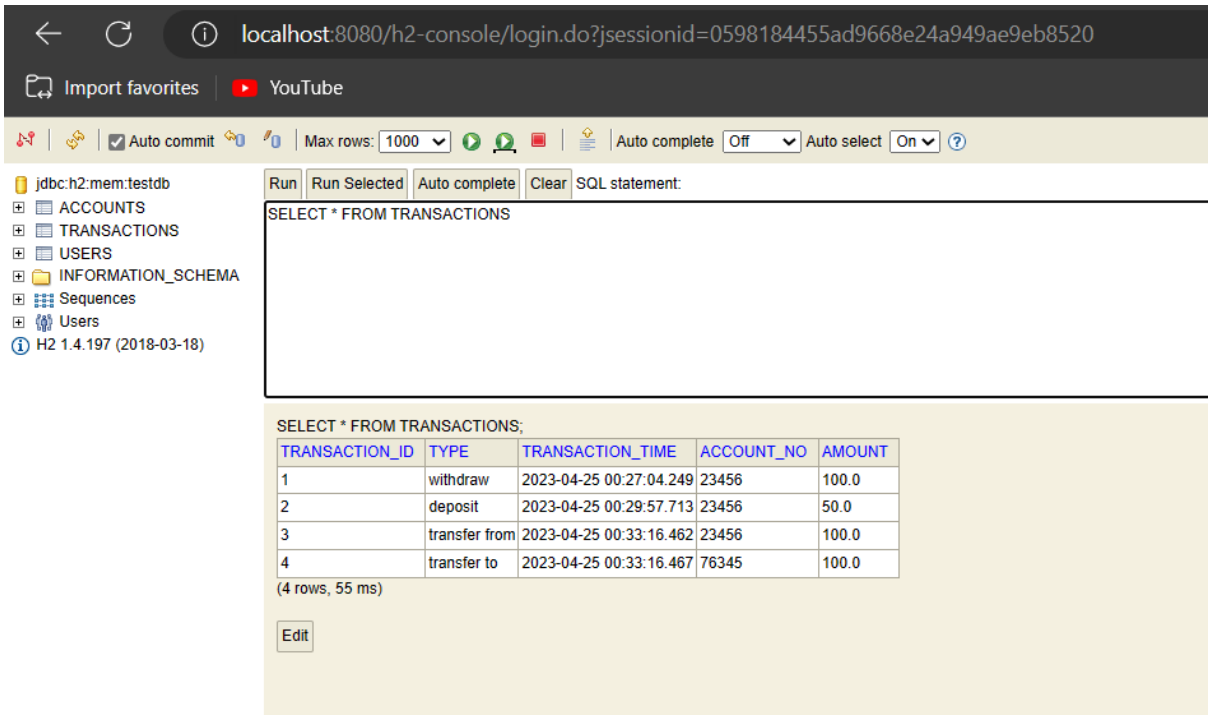


ACCOUNT_NO	PIN	ACCOUNT_TYPE	BALANCE	USER_ID	LOCKED	INCORRECT_ATTEMPTS
23456	4567	SAVINGS	3962.0	1	FALSE	0
76345	8001	CHECKING	5334.0	2	FALSE	0

Object Oriented Analysis and Design using Java (UE20CS352)

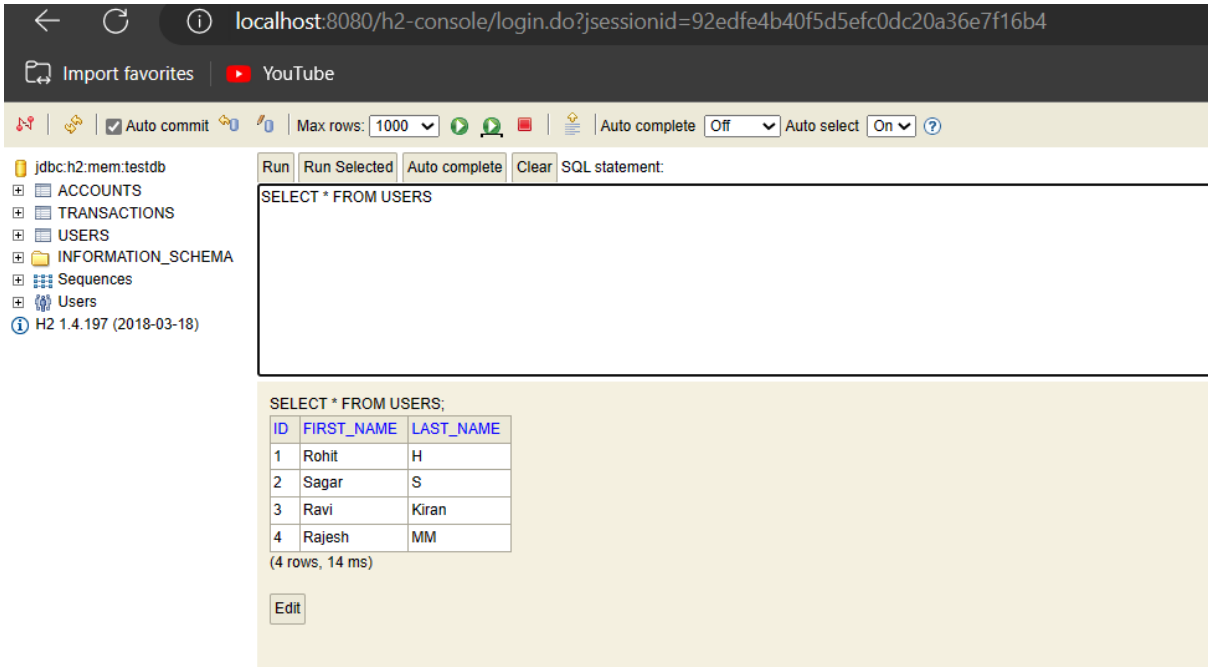
MINI PROJECT REPORT

Transaction Table:



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

User Table:



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