Experiment - 9

Student Name: Kumar adya **UID:** 23BCS80040
Branch: CSE **Section/Group:** 636/A
Semester: 6th **Date of Performance:** 06.04.25
Subject Name: JAVA **Subject Code:** 22CSH-359

1. **Aim:** To Develop a Spring-based application integrated with Hibernate to manage transactions. Create a banking system where users can transfer money between accounts, ensuring transaction consistency.

2. **Objective:** To develop a Spring-based banking application integrated with Hibernate that enables users to securely transfer money between accounts, ensuring data integrity and transaction consistency through proper transaction management.

3. **Implementation/Code:**

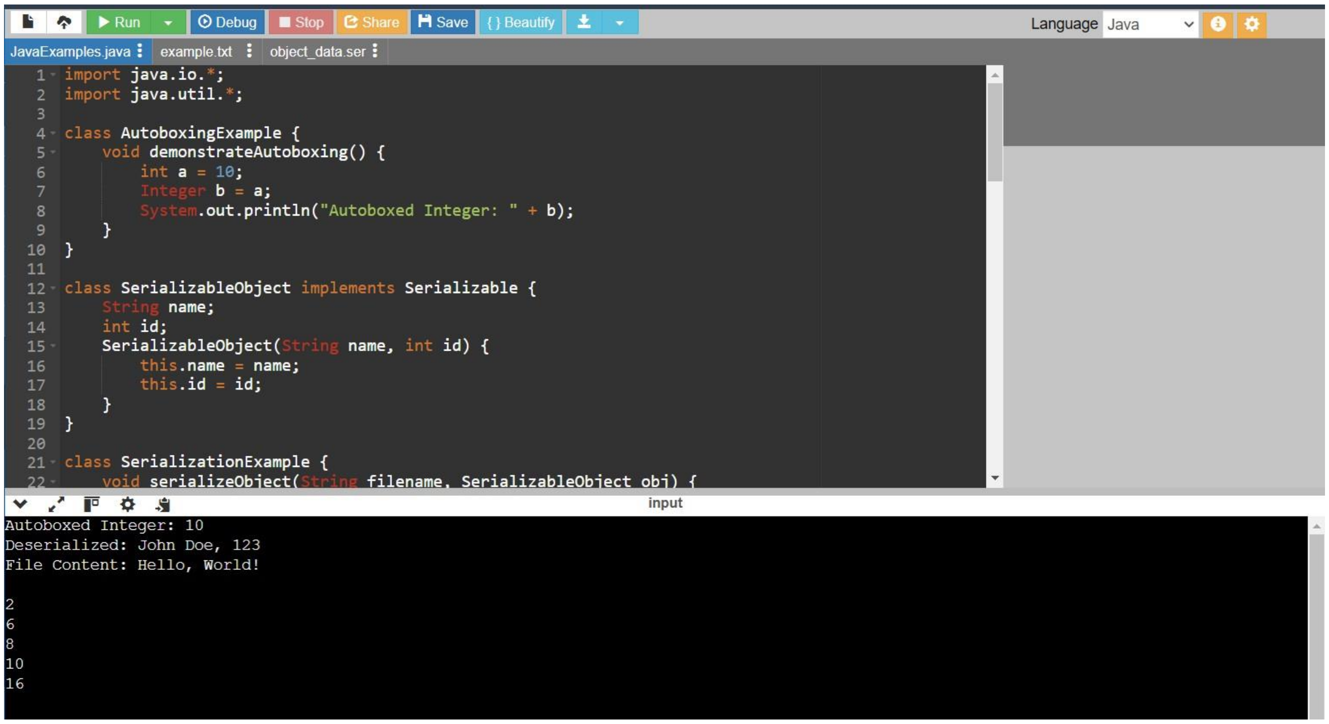
   - **Code**

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner; import
org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean; import
org.springframework.data.jpa.repository.JpaRepository; import
org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import javax.persistence.*;
```

```
@SpringBootApplication public class
BankingApp {     public static void
main(String[] args) {
        SpringApplication.run(BankingApp.class, args);
    }
    @Bean
    CommandLineRunner run(AccountService service) {
        return args -> {
service.createAccount("Alice", 1000);
service.createAccount("Bob", 500);
service.transferMoney("Alice", "Bob", 200);
        };
}
    @Entity     static class
Account {         @Id
@GeneratedValue
private Long id;
private String owner;
        private double balance;

        public Account() {}        public Account(String
owner, double balance) {             this.owner =
owner;             this.balance = balance;
        }

        // Getters & Setters         public String getOwner() { return owner;
}        public void setOwner(String owner) { this.owner = owner; }
public double getBalance() { return balance; }        public void
setBalance(double balance) { this.balance = balance; }
    }

    interface AccountRepo extends JpaRepository<Account, Long> {
        Account findByOwner(String owner);
    }
```

```
  @Service     static class
AccountService {
      @Autowired
      private AccountRepo repo;

      public void createAccount(String owner, double balance) {
repo.save(new Account(owner, balance));
      }
      @Transactional
      public void transferMoney(String from, String to, double amount) {
         Account src = repo.findByOwner(from);          Account dest =
repo.findByOwner(to);          if (src.getBalance() < amount) throw new
RuntimeException("Insufficient funds");          src.setBalance(src.getBalance() -
amount);          dest.setBalance(dest.getBalance() + amount);          repo.save(src);
repo.save(dest);
      }
   }
}
```

4. Output:



```java
import java.io.*;
import java.util.*;

class AutoboxingExample {
    void demonstrateAutoboxing() {
        int a = 10;
        Integer b = a;
        System.out.println("Autoboxed Integer: " + b);
    }
}

class SerializableObject implements Serializable {
    String name;
    int id;
    SerializableObject(String name, int id) {
        this.name = name;
        this.id = id;
    }
}

class SerializationExample {
    void serializeObject(String filename, SerializableObject obj) {
```

```
Autoboxed Integer: 10
Deserialized: John Doe, 123
File Content: Hello, World!

2
6
8
10
16
```

5. Learning Outcome:

By developing a Spring-based banking application integrated with Hibernate, learners will gain hands-on experience in managing database transactions in a real-world scenario. They will understand how to use Spring's declarative transaction management to ensure data consistency and integrity during operations such as fund transfers. This project will enhance their skills in using Spring Framework, Hibernate ORM, and relational database design while reinforcing best practices for transactional operations in enterprise applications.