**AccountService.java**

package com.example.service;

import javax.ejb.Local;

@Local

public interface AccountService {

boolean deposit(int accountId, double amount);

boolean withdraw(int accountId, double amount);

}

**AccountServiceImpl.java**

package com.example.service;

import com.example.entity.Account;

import javax.ejb.Stateless;

import javax.persistence.EntityManager;

import javax.persistence.PersistenceContext;

@Stateless

public class AccountServiceImpl implements AccountService {

@PersistenceContext(unitName = "BankPU")

private EntityManager em;

@Override

public boolean deposit(int accountId, double amount) {

Account account = em.find(Account.class, accountId);

if (account != null) {

account.setBalance(account.getBalance() + amount);

em.merge(account);

return true;

}

return false;

}

@Override

public boolean withdraw(int accountId, double amount) {

Account account = em.find(Account.class, accountId);

if (account != null && account.getBalance() >= amount) {

account.setBalance(account.getBalance() - amount);

em.merge(account);

return true;

}

return false;

}

}

**Account.java**

package com.example.entity;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

@Entity

public class Account {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private double balance;

// Getters and Setters

public int getId() { return id; }

public void setId(int id) { this.id = id; }

public double getBalance() { return balance; }

public void setBalance(double balance) { this.balance = balance; }

}

**AccountController.java**

package com.example.controller;

import com.example.service.AccountService;

import javax.ejb.EJB;

import javax.ws.rs.\*;

import javax.ws.rs.core.MediaType;

import javax.ws.rs.core.Response;

@Path("/account")

public class AccountController {

@EJB

private AccountService accountService;

@POST

@Path("/deposit/{id}/{amount}")

@Produces(MediaType.APPLICATION\_JSON)

public Response deposit(@PathParam("id") int accountId, @PathParam("amount") double amount) {

if (accountService.deposit(accountId, amount)) {

return Response.ok().entity("Deposit successful").build();

}

return Response.status(Response.Status.BAD\_REQUEST).entity("Deposit failed").build();

}

@POST

@Path("/withdraw/{id}/{amount}")

@Produces(MediaType.APPLICATION\_JSON)

public Response withdraw(@PathParam("id") int accountId, @PathParam("amount") double amount) {

if (accountService.withdraw(accountId, amount)) {

return Response.ok().entity("Withdrawal successful").build();

}

return Response.status(Response.Status.BAD\_REQUEST).entity("Insufficient funds").build();

}

}

