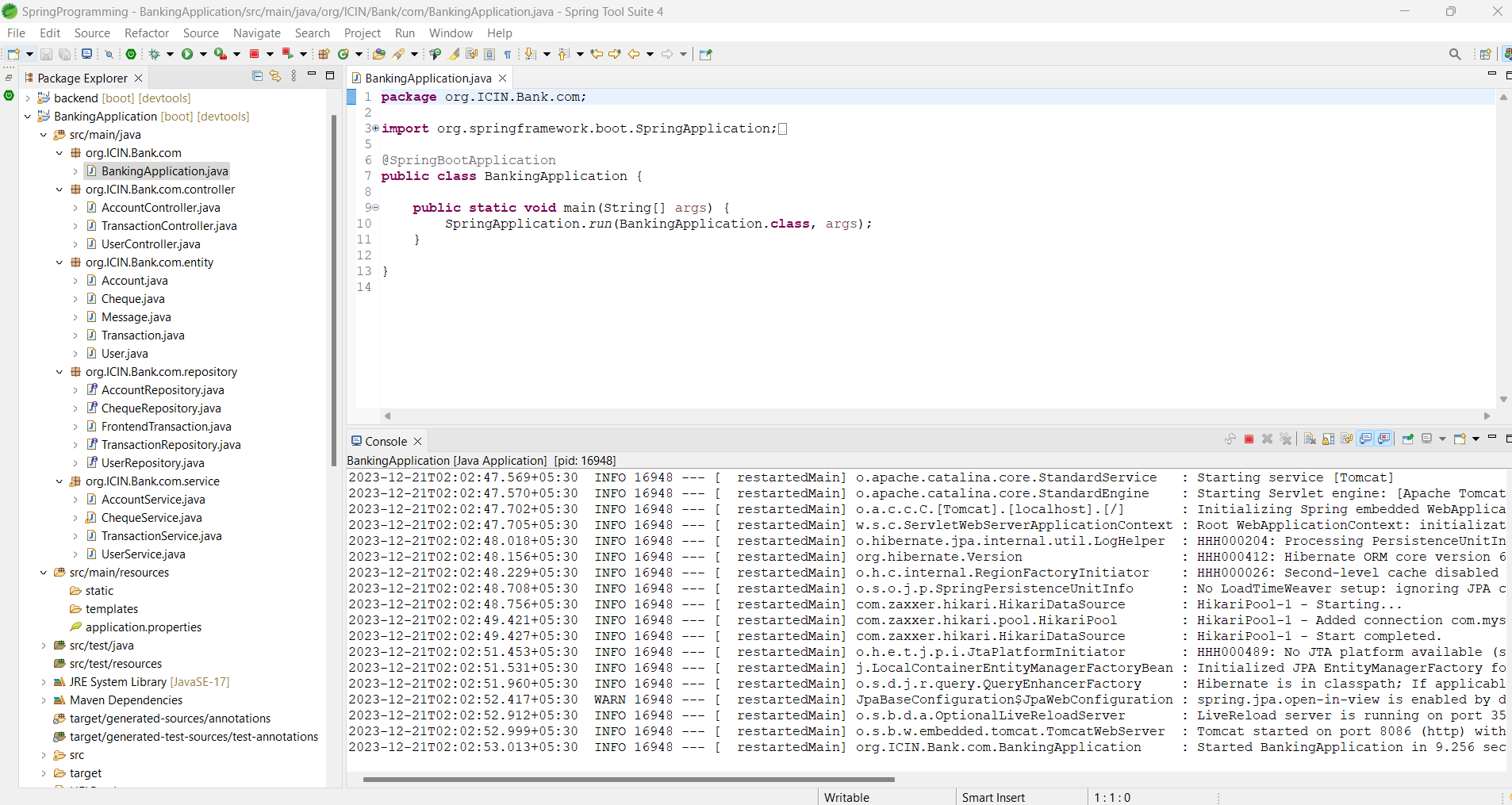
BackEnd:



**Controllers:**

**Account Controller:**

package org.ICIN.Bank.com.controller;

import java.util.List;

import org.ICIN.Bank.com.entity.Account;

import org.ICIN.Bank.com.entity.Message;

import org.ICIN.Bank.com.service.AccountService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/account")

@CrossOrigin("http://localhost:4200")

public class AccountController {

@Autowired

private AccountService accountService;

@GetMapping("/accounts")

public List<Account> getAllAccount() {

return accountService.getAllAccounts();

}

@GetMapping("/{accountNumber}")

public Account getAccountByAccountNumber (@PathVariable String accountNumber) {

return accountService.getAccount(accountNumber);

}

@GetMapping("/isNetBankingActivated/{accountNumber}")

public boolean checkIfNetBankingActivated(@PathVariable String accountNumber)

{

return accountService.isNetBankingReg(accountNumber);

}

@PostMapping("/add-account")

public Message addAccount(@RequestBody Account account) {

accountService.addAccount(account);

return new Message("Account added");

}

@GetMapping("/netbankingenable/{status}/{accountNumber}")

public Message enableNetBanking(@PathVariable int status, @PathVariable String accountNumber) {

Message msg=new Message(accountService.upadteAccountNetBankingStatus(1, accountNumber));

return msg;

}

@GetMapping("/unregisteredUsers")

public List<Account> getUnregisteredAccounts() {

return accountService.getAllUnregisterUser();

}

@GetMapping("/depositSavings/{balance}/{accountNumber}")

public Message depositSaving(@PathVariable int balance, @PathVariable String accountNumber) {

Message msg=new Message(accountService.depositSaving(balance, accountNumber));

return msg;

}

@GetMapping("/depositPrimary/{balance}/{accountNumber}")

public Message depositPrimary(@PathVariable int balance, @PathVariable String accountNumber) {

Message msg=new Message(accountService.depositPrimary(balance, accountNumber));

return msg;

}

@GetMapping("/withdrawSavings/{balance}/{accountNumber}")

public Message withdrawSaving(@PathVariable int balance, @PathVariable String accountNumber) {

Message msg=new Message(accountService.withdrawSaving(balance, accountNumber));

return msg;

}

@GetMapping("/withdrawPrimary/{balance}/{accountNumber}")

public Message withdrawPrimary(@PathVariable int balance, @PathVariable String accountNumber) {

Message msg= new Message(accountService.withdrawPrimary(balance, accountNumber));

return msg;

}

@PutMapping("/update/{accountNumber}")

public Account updateAccountDetails(@RequestBody Account account,@PathVariable String accountNumber)

{

return accountService.updateAccountDetails(accountNumber, account);

}

}

**TransactionCcontroller**:

package org.ICIN.Bank.com.controller;

import java.util.List;

import org.ICIN.Bank.com.entity.Message;

import org.ICIN.Bank.com.entity.Transaction;

import org.ICIN.Bank.com.repository.FrontendTransaction;

import org.ICIN.Bank.com.service.TransactionService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/transaction")

@CrossOrigin("http://localhost:4200")

public class TransactionController {

@Autowired

private TransactionService transactionService;

@PostMapping("/addTransactions")

public Message addTransaction(@RequestBody Transaction transaction) {

Message msg=new Message(transactionService.addTransaction(transaction));

return msg;

}

@GetMapping("/transactions/{accountNumber}")

public List<FrontendTransaction> getTransactionsByAccountNumber(@PathVariable String accountNumber) {

return transactionService.getTransactionsForAccountNumber(accountNumber);

}

@GetMapping("/admin/get-all-pending-transactions")

public List<Transaction> getPendingTransactions() {

return this.transactionService.getAllPendingTransactions();

}

@GetMapping("/admin/allow/transaction/{id}")

public Message permitTransaction(@PathVariable Long id) {

Message msg=new Message(transactionService.updateTransaction(id));

return msg;

}

@GetMapping("/transactions/{accountNumber}/{startDate}/{endDate}")

public List<FrontendTransaction> getFilteredTransactions(@PathVariable String accountNumber,

@PathVariable String startDate, @PathVariable String endDate) {

return this.transactionService.getFilteredTransactions(accountNumber, startDate, endDate);

}

}

**UserController**

**package org.ICIN.Bank.com.controller;**

**import java.util.List;**

**import org.ICIN.Bank.com.entity.Cheque;**

**import org.ICIN.Bank.com.entity.Message;**

**import org.ICIN.Bank.com.entity.User;**

**import org.ICIN.Bank.com.service.ChequeService;**

**import org.ICIN.Bank.com.service.UserService;**

**import org.springframework.beans.factory.annotation.Autowired;**

**import org.springframework.web.bind.annotation.CrossOrigin;**

**import org.springframework.web.bind.annotation.GetMapping;**

**import org.springframework.web.bind.annotation.PathVariable;**

**import org.springframework.web.bind.annotation.PostMapping;**

**import org.springframework.web.bind.annotation.RequestBody;**

**import org.springframework.web.bind.annotation.RequestMapping;**

**import org.springframework.web.bind.annotation.RestController;**

**@RestController**

**@RequestMapping("/user")**

**@CrossOrigin("http://localhost:4200")**

**public class UserController {**

**@Autowired**

**private UserService userService;**

**@Autowired**

**private ChequeService chequeService;**

**@GetMapping("/users")**

**public List<User> getAllUser() {**

**return userService.getAllUser();**

**}**

**@GetMapping("/{userId}")**

**public User getUserByUserId(@PathVariable String userId) {**

**return userService.getUserbyUserId(userId);**

**}**

**@GetMapping("/check/{userId}/{password}")**

**public Message checkUserByIdAndPassword(@PathVariable String userId, @PathVariable String password) {**

**Message msg =new Message (userService.checkUserByIdAndPassword(userId,password));**

**return msg;**

**}**

**@PostMapping("/addUser")**

**public Message addUser(@RequestBody User user) {**

**Message msg=new Message(userService.addUser(user));**

**return msg;**

**}**

**@GetMapping("/update/loginPassword/{newPassword}/{accountNumber}")**

**public Message updateLoginPassword(@PathVariable String newPassword, @PathVariable String accountNumber) {**

**Message msg=new Message(userService.updateLoginPassword(newPassword, accountNumber));**

**return msg;**

**}**

**@GetMapping("/update/transactionPassword/{newPassword}/{accountNumber}")**

**public Message updateTransactionPassword(@PathVariable String newPassword, @PathVariable String accountNumber) {**

**Message msg=new Message(userService.updateLoginPassword(newPassword, accountNumber));**

**return msg;**

**}**

**@GetMapping("/request-cheque-book/{accountNumber}/{accountType}")**

**public Message requestChequeBook(@PathVariable String accountNumber, @PathVariable String accountType) {**

**Message msg =new Message(chequeService.generateChequeBook(accountNumber, accountType));**

**return msg;**

**}**

**@GetMapping("/cheque-books/{accountNumber}")**

**public List<Cheque> getChequeBooksIssuedByAccountNumber(@PathVariable String accountNumber) {**

**return chequeService.getAllChequeBookIssued(accountNumber);**

**}**

**@GetMapping("/cheque-books-requests")**

**public List<Cheque> getAllChequeBookRequests() {**

**return chequeService.getAllPendingChequeBookRequests();**

**}**

**@GetMapping("/cheque-books-requests/accept/{chequeBookNumber}")**

**public Message acceptChequeBookRequest(@PathVariable String chequeBookNumber) {**

**Message msg=new Message(chequeService.changeChequeBookStatus(chequeBookNumber));**

**return msg;**

**}**

**@GetMapping("/block/{userId}")**

**public Message blockUser(@PathVariable String userId) {**

**Message msg=new Message(userService.blockUser(userId));**

**return msg;**

**}**

**@GetMapping("/unblock/{userId}")**

**public Message unblockUser(@PathVariable String userId) {**

**Message msg=new Message(userService.unblockUser(userId));**

**return msg;**

**}**

**@GetMapping("/blocked-users")**

**public List<User> getAllblockUser() {**

**return userService.getAllBlockedUser();**

**}**

**}**

**Entity**

**AccountEntiry**

package org.ICIN.Bank.com.controller;

import java.util.List;

import org.ICIN.Bank.com.entity.Cheque;

import org.ICIN.Bank.com.entity.Message;

import org.ICIN.Bank.com.entity.User;

import org.ICIN.Bank.com.service.ChequeService;

import org.ICIN.Bank.com.service.UserService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/user")

@CrossOrigin("http://localhost:4200")

public class UserController {

@Autowired

private UserService userService;

@Autowired

private ChequeService chequeService;

@GetMapping("/users")

public List<User> getAllUser() {

return userService.getAllUser();

}

@GetMapping("/{userId}")

public User getUserByUserId(@PathVariable String userId) {

return userService.getUserbyUserId(userId);

}

@GetMapping("/check/{userId}/{password}")

public Message checkUserByIdAndPassword(@PathVariable String userId, @PathVariable String password) {

Message msg =new Message (userService.checkUserByIdAndPassword(userId,password));

return msg;

}

@PostMapping("/addUser")

public Message addUser(@RequestBody User user) {

Message msg=new Message(userService.addUser(user));

return msg;

}

@GetMapping("/update/loginPassword/{newPassword}/{accountNumber}")

public Message updateLoginPassword(@PathVariable String newPassword, @PathVariable String accountNumber) {

Message msg=new Message(userService.updateLoginPassword(newPassword, accountNumber));

return msg;

}

@GetMapping("/update/transactionPassword/{newPassword}/{accountNumber}")

public Message updateTransactionPassword(@PathVariable String newPassword, @PathVariable String accountNumber) {

Message msg=new Message(userService.updateLoginPassword(newPassword, accountNumber));

return msg;

}

@GetMapping("/request-cheque-book/{accountNumber}/{accountType}")

public Message requestChequeBook(@PathVariable String accountNumber, @PathVariable String accountType) {

Message msg =new Message(chequeService.generateChequeBook(accountNumber, accountType));

return msg;

}

@GetMapping("/cheque-books/{accountNumber}")

public List<Cheque> getChequeBooksIssuedByAccountNumber(@PathVariable String accountNumber) {

return chequeService.getAllChequeBookIssued(accountNumber);

}

@GetMapping("/cheque-books-requests")

public List<Cheque> getAllChequeBookRequests() {

return chequeService.getAllPendingChequeBookRequests();

}

@GetMapping("/cheque-books-requests/accept/{chequeBookNumber}")

public Message acceptChequeBookRequest(@PathVariable String chequeBookNumber) {

Message msg=new Message(chequeService.changeChequeBookStatus(chequeBookNumber));

return msg;

}

@GetMapping("/block/{userId}")

public Message blockUser(@PathVariable String userId) {

Message msg=new Message(userService.blockUser(userId));

return msg;

}

@GetMapping("/unblock/{userId}")

public Message unblockUser(@PathVariable String userId) {

Message msg=new Message(userService.unblockUser(userId));

return msg;

}

@GetMapping("/blocked-users")

public List<User> getAllblockUser() {

return userService.getAllBlockedUser();

}

}

ChequeEntity

package org.ICIN.Bank.com.entity;

import java.util.Date;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class Cheque {

@Id

private String chequeBookNumber;

private String accountNumber;

private String accountType;

private Date chequeBookIssueDate;

private int chequeBookStatus;

public Cheque(String chequeBookNumber, String accountNumber, String accountType, Date chequeBookIssueDate,

int chequeBookStatus) {

super();

this.chequeBookNumber = chequeBookNumber;

this.accountNumber = accountNumber;

this.accountType = accountType;

this.chequeBookIssueDate = chequeBookIssueDate;

this.chequeBookStatus = chequeBookStatus;

}

public Cheque() {

// TODO Auto-generated constructor stub

}

public String getChequeBookNumber() {

return chequeBookNumber;

}

public void setChequeBookNumber(String chequeBookNumber) {

this.chequeBookNumber = chequeBookNumber;

}

public String getAccountNumber() {

return accountNumber;

}

public void setAccountNumber(String accountNumber) {

this.accountNumber = accountNumber;

}

public String getAccountType() {

return accountType;

}

public void setAccountType(String accountType) {

this.accountType = accountType;

}

public Date getChequeBookIssueDate() {

return chequeBookIssueDate;

}

public void setChequeBookIssueDate(Date chequeBookIssueDate) {

this.chequeBookIssueDate = chequeBookIssueDate;

}

public int getChequeBookStatus() {

return chequeBookStatus;

}

public void setChequeBookStatus(int chequeBookStatus) {

this.chequeBookStatus = chequeBookStatus;

}

}

Massege

**package** org.ICIN.Bank.com.entity;

**public** **class** Message {

**private** String message;

**public** String getMessage() {

**return** message;

}

**public** **void** setMessage(String message) {

**this**.message = message;

}

**public** Message(String message) {

**super**();

**this**.message = message;

}

**public** Message() {

**super**();

}

}

**Transaction Entity**

**package** org.ICIN.Bank.com.entity;

**public** **class** Message {

**private** String message;

**public** String getMessage() {

**return** message;

}

**public** **void** setMessage(String message) {

**this**.message = message;

}

**public** Message(String message) {

**super**();

**this**.message = message;

}

**public** Message() {

**super**();

}

}

UserEntity

package org.ICIN.Bank.com.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class User {

@Id

// @GeneratedValue(generator = "system-uuid")

// @GenericGenerator(name = "system-uuid", strategy = "uuid")

private String accountNumber;

private String userId;

private String userName;

private String password;

private int accountIsBlocked;

public User(String accountNumber, String userName, String password, int accountIsBlocked) {

super();

this.accountNumber = accountNumber;

this.userName = userName;

this.password = password;

this.accountIsBlocked = accountIsBlocked;

}

public User() {

// TODO Auto-generated constructor stub

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getAccountNumber() {

return accountNumber;

}

public void setAccountNumber(String accountNumber) {

this.accountNumber = accountNumber;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public int getAccountIsBlocked() {

return accountIsBlocked;

}

public void setAccountIsBlocked(int accountIsBlocked) {

this.accountIsBlocked = accountIsBlocked;

}

}

AccountRepo

package org.ICIN.Bank.com.repository;

import java.util.List;

import org.ICIN.Bank.com.entity.Account;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import jakarta.transaction.Transactional;

public interface AccountRepository extends JpaRepository<Account, String>{

@Query(value = "SELECT \* from account where ACCOUNT\_NUMBER like ?1", nativeQuery = true)

List<Account> getAccountByAccountNumber(String accountNumber);

@Transactional

@Modifying

@Query(value = "UPDATE account set NET\_BANKING=?1 where ACCOUNT\_NUMBER like ?2", nativeQuery = true)

void updateNetBankingStatus(int status ,String accountNumber);

@Query(value = "SELECT \* from account where NET\_BANKING like ?1", nativeQuery = true)

List<Account> getAllUnRegisteredUser(int Status);

@Transactional

@Modifying

@Query(value = "UPDATE account set BALANCE\_PRIMARY=?1 where ACCOUNT\_NUMBER like ?2", nativeQuery = true)

void updatePrimaryBalance(double balance,String accountNumber);

@Transactional

@Modifying

@Query(value = "UPDATE account set BALANCE\_SAVINGS=?1 where ACCOUNT\_NUMBER like ?2", nativeQuery = true)

void updateSavingBalance(double balance,String accountNumber);

@Transactional

@Modifying

@Query(value = "UPDATE account set BALANCE\_PRIMARY=?1 where ACCOUNT\_NUMBER like ?2", nativeQuery = true)

void depositPrimary(int balance,String accountNumber);

@Transactional

@Modifying

@Query(value = "UPDATE account set BALANCE\_SAVINGS=?1 where ACCOUNT\_NUMBER like ?2", nativeQuery = true)

void depositSaving(int balance,String accountNumber);

@Transactional

@Modifying

@Query(value = "UPDATE account set BALANCE\_PRIMARY=?1 where ACCOUNT\_NUMBER like ?2", nativeQuery = true)

void withdrawPrimary(int balance, String accountNumber);

@Transactional

@Modifying

@Query(value = "UPDATE account set BALANCE\_SAVINGS=?1 where ACCOUNT\_NUMBER like ?2", nativeQuery = true)

void withdrawSaving(int balance, String accountNumber);

}

ChequeRepo

package org.ICIN.Bank.com.repository;

import java.util.List;

import org.ICIN.Bank.com.entity.Cheque;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import jakarta.transaction.Transactional;

public interface ChequeRepository extends JpaRepository<Cheque, String>{

@Query(value = "SELECT \* from cheque where ACCOUNT\_NUMBER like ?1 order by CHEQUE\_BOOK\_ISSUE\_DATE desc", nativeQuery = true)

List<Cheque> getChequeBookIssueByAccountNumber(String accountNumber);

@Query(value = "SELECT \* from cheque where CHEQUE\_BOOK\_STATUS like ?1 order by CHEQUE\_BOOK\_ISSUE\_DATE", nativeQuery = true)

List<Cheque> getAllPendingChequeBookRequests(int status);

@Transactional

@Modifying

@Query(value = "UPDATE cheque set CHEQUE\_BOOK\_STATUS=?1 where CHEQUE\_BOOK\_NUMBER like ?2", nativeQuery = true)

void getUpdatedPendingChequeBookRequests(int i, String chequeBookNumber);

}

FrontendTransaction

**package** org.ICIN.Bank.com.repository;

**import** org.springframework.stereotype.Repository;

@Repository

**public** **class** FrontendTransaction {

**private** Long id;

**private** String fromAccountNumber;

**private** String toAccountNumber;

**private** String fromUserName;

**private** String toUserName;

**private** String fromAccountType;

**private** String toAccountType;

**private** **double** transferAmount;

**private** String transferMessage;

**private** String transferType;

**private** String transferDate;

**private** **int** transferStatus;

**public** FrontendTransaction() {

// **TODO** Auto-generated constructor stub

}

**public** FrontendTransaction(Long id, String fromAccountNumber, String toAccountNumber, String fromUserName,

String toUserName, String fromAccountType, String toAccountType, **double** transferAmount,

String transferMessage, String transferType, String transferDate, **int** transferStatus) {

**super**();

**this**.id = id;

**this**.fromAccountNumber = fromAccountNumber;

**this**.toAccountNumber = toAccountNumber;

**this**.fromUserName = fromUserName;

**this**.toUserName = toUserName;

**this**.fromAccountType = fromAccountType;

**this**.toAccountType = toAccountType;

**this**.transferAmount = transferAmount;

**this**.transferMessage = transferMessage;

**this**.transferType = transferType;

**this**.transferDate = transferDate;

**this**.transferStatus = transferStatus;

}

**public** Long getId() {

**return** id;

}

**public** **void** setId(Long id) {

**this**.id = id;

}

**public** String getFromAccountNumber() {

**return** fromAccountNumber;

}

**public** **void** setFromAccountNumber(String fromAccountNumber) {

**this**.fromAccountNumber = fromAccountNumber;

}

**public** String getToAccountNumber() {

**return** toAccountNumber;

}

**public** **void** setToAccountNumber(String toAccountNumber) {

**this**.toAccountNumber = toAccountNumber;

}

**public** String getFromUserName() {

**return** fromUserName;

}

**public** **void** setFromUserName(String fromUserName) {

**this**.fromUserName = fromUserName;

}

**public** String getToUserName() {

**return** toUserName;

}

**public** **void** setToUserName(String toUserName) {

**this**.toUserName = toUserName;

}

**public** String getFromAccountType() {

**return** fromAccountType;

}

**public** **void** setFromAccountType(String fromAccountType) {

**this**.fromAccountType = fromAccountType;

}

**public** String getToAccountType() {

**return** toAccountType;

}

**public** **void** setToAccountType(String toAccountType) {

**this**.toAccountType = toAccountType;

}

**public** **double** getTransferAmount() {

**return** transferAmount;

}

**public** **void** setTransferAmount(**double** transferAmount) {

**this**.transferAmount = transferAmount;

}

**public** String getTransferMessage() {

**return** transferMessage;

}

**public** **void** setTransferMessage(String transferMessage) {

**this**.transferMessage = transferMessage;

}

**public** String getTransferType() {

**return** transferType;

}

**public** **void** setTransferType(String transferType) {

**this**.transferType = transferType;

}

**public** String getTransferDate() {

**return** transferDate;

}

**public** **void** setTransferDate(String transferDate) {

**this**.transferDate = transferDate;

}

**public** **int** getTransferStatus() {

**return** transferStatus;

}

**public** **void** setTransferStatus(**int** transferStatus) {

**this**.transferStatus = transferStatus;

}

}

Transaction

package org.ICIN.Bank.com.repository;

import java.util.List;

import org.ICIN.Bank.com.entity.Transaction;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import jakarta.transaction.Transactional;

@Repository

public interface TransactionRepository extends JpaRepository<Transaction, Long> {

@Query(value = "SELECT \* from transaction where FROM\_ACCOUNT\_NUMBER like ?1 OR TO\_ACCOUNT\_NUMBER like ?1 order by ID desc", nativeQuery = true)

List<Transaction> getTransactionForAccountNumber(String accountNumber);

@Query(value = "SELECT \* from transaction where TRANSFER\_STATUS like ?1 order by TRANSFER\_DATE", nativeQuery = true)

List<Transaction> getPendingTransactions(int status);

@Query(value = "SELECT \* from transaction " + "where (FROM\_ACCOUNT\_NUMBER like ?1 OR TO\_ACCOUNT\_NUMBER like ?1) "

+ "and " + "(TRANSFER\_DATE between CAST(?2 as DATE) and CAST(?3 as DATE)) "

+ "order by TRANSFER\_DATE desc", nativeQuery = true)

List<Transaction> getFilteredTransactions(String accountNumber, String startDate, String endDate);

@Transactional

@Modifying

@Query(value = "UPDATE transaction set TRANSFER\_STATUS=?1 where ID like ?2", nativeQuery = true)

void updateTransaction(int status, Long id);

}

UserRepo

package org.ICIN.Bank.com.repository;

import java.util.List;

import org.ICIN.Bank.com.entity.User;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import jakarta.transaction.Transactional;

public interface UserRepository extends JpaRepository<User, String> {

@Query(value = "SELECT \* from user where USER\_ID like ?1", nativeQuery = true)

List<User> getUserByUserId(String userId);

@Query(value = "SELECT \* from user where ACCOUNT\_IS\_BLOCKED like ?1", nativeQuery = true)

List<User> getAllBlockedUser(int status);

@Transactional

@Modifying

@Query(value = "UPDATE user set PASSWORD=?1 where ACCOUNT\_NUMBER like ?2", nativeQuery = true)

void updateUserPassword(String newPassword, String accountNumber);

@Transactional

@Modifying

@Query(value = "UPDATE user set ACCOUNT\_IS\_BLOCKED=?1 where USER\_ID like ?2", nativeQuery = true)

void toggleBlockUser(int status, String userId);

}

AccountService

package org.ICIN.Bank.com.service;

import java.util.List;

import org.ICIN.Bank.com.entity.Account;

import org.ICIN.Bank.com.repository.AccountRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import jakarta.transaction.Transactional;

@Service

public class AccountService {

@Autowired

private AccountRepository accountRepository;

public List<Account> getAllAccounts() {

return accountRepository.findAll();

}

public Account getAccount(String accountNumber) {

List<Account> account = accountRepository.getAccountByAccountNumber(accountNumber);

if (account.size() == 0) {

return null;

}

return account.get(0);

}

public boolean isNetBankingReg(String accountNumber) {

List<Account> account = this.accountRepository.findAll();

for (Account accounts : account) {

if (accounts.getAccountNumber().equals(accountNumber)) {

if (accounts.getNetBanking() == 0) {

return false;

}

return false;

}

}

return false;

}

public Account addAccount(Account account) {

// List<Account> account1 = accountRepository.findAll();

// for (Account temp : account1) {

// if (temp.getAccountNumber().equals(account.getAccountNumber())) {

// return account;

// } else {

// this.accountRepository.save(account);

// }

// }

accountRepository.save(account);

return account;

}

public Account getAccountById(String accId) {

return accountRepository.findById(accId).get();

}

@Transactional

public Account updateAccountDetails(String accountNumber, Account account) {

Account acc = new Account();

acc.setAccountCIFNumber(account.getAccountCIFNumber());

acc.setAccountNumber(account.getAccountNumber());

acc.setUserName(account.getUserName());

acc.setBalancePrimary(account.getBalancePrimary());

acc.setBalanceSavings(account.getBalanceSavings());

acc.setBranch(account.getBranch());

acc.setNetBanking(account.getNetBanking());

return accountRepository.save(acc);

}

@Transactional

public String upadteAccountNetBankingStatus(int status, String accountNumber) {

accountRepository.updateNetBankingStatus(1, accountNumber);

return "Account is now net banking enable";

}

public List<Account> getAllUnregisterUser() {

return accountRepository.getAllUnRegisteredUser(1);

}

public String depositPrimary(int balance, String accountNumber) {

this.accountRepository.depositPrimary(balance, accountNumber);

return "Deposit to primary account is successful";

}

public String depositSaving(int balance, String accountNumber) {

this.accountRepository.depositSaving(balance, accountNumber);

return "Deposit to savings account is successful";

}

public String withdrawPrimary(int balance, String accountNumber) {

accountRepository.withdrawPrimary(balance, accountNumber);

return "Withdrawal from primary account is successful";

}

public String withdrawSaving(int balance, String accountNumber) {

accountRepository.withdrawSaving(balance, accountNumber);

return "Withdrawal from savings account is successful";

}

}

Cheque Service

package org.ICIN.Bank.com.service;

import java.util.Date;

import java.util.List;

import org.ICIN.Bank.com.entity.Cheque;

import org.ICIN.Bank.com.repository.ChequeRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import jakarta.transaction.Transactional;

@Service

public class ChequeService {

@Autowired

private ChequeRepository chequeRepository;

@Transactional

public String generateChequeBook(String accountNumber, String accountType)

{

// if (isSomeChequeBookRequested(accountNumber, accountType).equals("No pending request"))

// {

int count = getCheckBookCount(accountNumber) + 1;

Cheque cheque = new Cheque();

cheque.setAccountNumber(accountNumber);

cheque.setChequeBookNumber(accountType + "" + count);

// cheque.setChequeBookNumber(chequeBook);

cheque.setAccountType(accountType);

cheque.setChequeBookIssueDate(new Date());

cheque.setChequeBookStatus(0);

this.chequeRepository.save(cheque);

return "Requested for a cheque book for account number - " + accountNumber;

// }

// return "There is already a pending request for cheque book from " + accountType + " account - " + accountNumber;

}

private int getCheckBookCount(String accountNumber) {

int count = 0;

List<Cheque> cheques = this.chequeRepository.findAll();

for (Cheque temp : cheques) {

if (temp.getAccountNumber().equals(accountNumber)) {

count++;

}

}

return count;

}

private Object isSomeChequeBookRequested(String accountNumber, String accountType) {

List<Cheque> list = this.chequeRepository.findAll();

for (Cheque temp : list) {

if (temp.getAccountNumber().equals(accountNumber) && temp.getAccountType().equals(accountType)) {

if (temp.getChequeBookStatus() == 0) {

return "Already Requested!";

}

}

}

return "No Pending Request";

}

public String changeChequeBookStatus(String chequeBookNumber) {

Cheque cheque = this.chequeRepository.getOne(chequeBookNumber);

this.chequeRepository.getUpdatedPendingChequeBookRequests(1, chequeBookNumber);

return "Cheque Book Request Accepted for account number - " + cheque.getAccountNumber();

}

public List<Cheque> getAllChequeBookIssued(String accountNumber) {

return chequeRepository.getChequeBookIssueByAccountNumber(accountNumber);

}

public List<Cheque> getAllPendingChequeBookRequests() {

return chequeRepository.getAllPendingChequeBookRequests(1);

}

}

Transaction Service

*package org.ICIN.Bank.com.service;*

*import java.util.ArrayList;*

*import java.util.Date;*

*import java.util.List;*

*import org.ICIN.Bank.com.entity.Account;*

*import org.ICIN.Bank.com.entity.Transaction;*

*import org.ICIN.Bank.com.repository.AccountRepository;*

*import org.ICIN.Bank.com.repository.FrontendTransaction;*

*import org.ICIN.Bank.com.repository.TransactionRepository;*

*import org.springframework.beans.factory.annotation.Autowired;*

*import org.springframework.stereotype.Service;*

*@Service*

*public class TransactionService {*

*@Autowired*

*private TransactionRepository transactionRepository;*

*@Autowired*

*private AccountRepository accountRepository;*

*public String addTransaction(Transaction transaction) {*

*List<Transaction> transactions = transactionRepository.getTransactionForAccountNumber(transaction.getFromAccountNumber());*

*double amountPrimary = 0;*

*double amountSaving = 0;*

*for (Transaction temp : transactions)*

*{*

*if(temp.getTransferStatus()==1)*

*{*

*if(temp.getFromAccountType().equals("primary"))*

*{*

*amountPrimary+=temp.getTransferAmount();*

*}else {*

*amountSaving+=temp.getTransferAmount();*

*}*

*}*

*}*

*@SuppressWarnings("deprecation")*

*Account accountSender=accountRepository.getOne(transaction.getFromAccountNumber());*

*@SuppressWarnings("deprecation")*

*Account accountReceiver=accountRepository.getOne(transaction.getToAccountNumber());*

*boolean check=false;*

*List<Account>acc=accountRepository.findAll();*

*for(Account temp:acc)*

*{*

*if(temp.getAccountNumber().equals(transaction.getToAccountNumber()))*

*{*

*check=true;*

*}*

*}*

*if(!check)*

*{*

*return "Transfer bank account does not exists!";*

*}*

*@SuppressWarnings("deprecation")*

*Account myAccount = accountRepository.getOne(transaction.getFromAccountNumber());*

*if (transaction.getFromAccountType().equals("Primary")) {*

*if (myAccount.getBalancePrimary() - amountPrimary < transaction.getTransferAmount()) {*

*return "You already have some pending transactions!\nYour primary account would not have that much balance if these transactions are permitted!";*

*}*

*} else {*

*if (myAccount.getBalanceSavings() - amountSaving < transaction.getTransferAmount()) {*

*return "You already have some pending transactions!\nYour savings account would not have that much balance if these transactions are permitted!";*

*}*

*}*

*// Transaction finalTransaction = new Transaction(transaction.getFromAccountNumber(),*

*// transaction.getToAccountNumber(), accountSender.getUserName(),*

*// accountReceiver.getUserName(), transaction.getFromAccountType(),*

*// transaction.getToAccountType(), transaction.getTransferAmount(), transaction.getTransferMessage(),*

*// new Date(), transaction.getTransferStatus());*

*Transaction finalTransaction=new Transaction(transaction.getFromAccountNumber(),transaction.getToAccountNumber(),accountSender.getUserName(),*

*accountReceiver.getAccountNumber(),transaction.getFromAccountType(),transaction.getToAccountType(),transaction.getTransferAmount(),*

*transaction.getTransferMessage(),new Date(), transaction.getTransferStatus());*

*this.transactionRepository.save(finalTransaction);*

*return "Transfer initiated.\nCheck the status in the transactions tab!";*

*}*

*@SuppressWarnings("deprecation")*

*public String updateTransaction(Long id) {*

*Transaction transaction = transactionRepository.getOne(id);*

*Account receiverAccount = this.accountRepository.getOne(transaction.getToAccountNumber());*

*Account senderAccount = this.accountRepository.getOne(transaction.getFromAccountNumber());*

*if(transaction.getFromAccountType().equals("Primary")) {*

*if(transaction.getToAccountType().equals("Primary")) {*

*this.accountRepository.updatePrimaryBalance(*

*receiverAccount.getBalancePrimary()+transaction.getTransferAmount(),*

*receiverAccount.getAccountNumber()*

*);*

*} else {*

*this.accountRepository.updateSavingBalance(*

*receiverAccount.getBalanceSavings()+transaction.getTransferAmount(),*

*receiverAccount.getAccountNumber()*

*);*

*}*

*this.accountRepository.updatePrimaryBalance(*

*senderAccount.getBalancePrimary()-transaction.getTransferAmount(),*

*senderAccount.getAccountNumber()*

*);*

*} else {*

*if(transaction.getToAccountType().equals("Primary")) {*

*this.accountRepository.updatePrimaryBalance(*

*receiverAccount.getBalancePrimary()+transaction.getTransferAmount(),*

*receiverAccount.getAccountNumber()*

*);*

*} else {*

*this.accountRepository.updateSavingBalance(*

*receiverAccount.getBalanceSavings()+transaction.getTransferAmount(),*

*receiverAccount.getAccountNumber()*

*);*

*}*

*this.accountRepository.updateSavingBalance(*

*senderAccount.getBalanceSavings()-transaction.getTransferAmount(),*

*senderAccount.getAccountNumber()*

*);*

*}*

*this.transactionRepository.updateTransaction(1, id);*

*return "Transaction Permitted!";*

*}*

*public List<FrontendTransaction> getTransactionsForAccountNumber(String accountNumber) {*

*List<Transaction> mainList =transactionRepository.getTransactionForAccountNumber(accountNumber);*

*List<FrontendTransaction> toReturnList = new ArrayList<FrontendTransaction>();*

*for (Transaction temp : mainList) {*

*FrontendTransaction tempTransaction = new FrontendTransaction();*

*tempTransaction.setId(temp.getId());*

*tempTransaction.setFromAccountNumber(temp.getFromAccountNumber());*

*tempTransaction.setToAccountNumber(temp.getToAccountNumber());*

*tempTransaction.setFromUserName(temp.getFromUserName());*

*tempTransaction.setToUserName(temp.getToUserName());*

*tempTransaction.setFromAccountType(temp.getFromAccountType());*

*tempTransaction.setToAccountType(temp.getToAccountType());*

*tempTransaction.setTransferAmount(temp.getTransferAmount());*

*tempTransaction.setTransferMessage(temp.getTransferMessage());*

*tempTransaction.setTransferDate(temp.getTransferDate()+"");*

*tempTransaction.setTransferStatus(temp.getTransferStatus());*

*if (temp.getFromAccountNumber().equals(accountNumber)) {*

*tempTransaction.setTransferType("Debited");*

*} else {*

*tempTransaction.setTransferType("Credited");*

*}*

*toReturnList.add(tempTransaction);*

*}*

*return toReturnList;*

*}*

*public List<Transaction> getAllPendingTransactions() {*

*return this.transactionRepository.getPendingTransactions(0);*

*}*

*public List<FrontendTransaction> getFilteredTransactions(String accountNumber, String startDate, String endDate) {*

*List<Transaction> mainList = this.transactionRepository*

*.getFilteredTransactions(accountNumber, startDate, endDate);*

*System.out.println(mainList);*

*List<FrontendTransaction> toReturnList = new ArrayList<FrontendTransaction>();*

*for (Transaction temp : mainList) {*

*FrontendTransaction tempTransaction = new FrontendTransaction();*

*tempTransaction.setId(temp.getId());*

*tempTransaction.setFromAccountNumber(temp.getFromAccountNumber());*

*tempTransaction.setToAccountNumber(temp.getToAccountNumber());*

*tempTransaction.setFromUserName(temp.getFromUserName());*

*tempTransaction.setToUserName(temp.getToUserName());*

*tempTransaction.setFromAccountType(temp.getFromAccountType());*

*tempTransaction.setToAccountType(temp.getToAccountType());*

*tempTransaction.setTransferAmount(temp.getTransferAmount());*

*tempTransaction.setTransferMessage(temp.getTransferMessage());*

*tempTransaction.setTransferDate(temp.getTransferDate()+"");*

*tempTransaction.setTransferStatus(temp.getTransferStatus());*

*if (temp.getFromAccountNumber().equals(accountNumber)) {*

*tempTransaction.setTransferType("Debited");*

*} else {*

*tempTransaction.setTransferType("Credited");*

*}*

*toReturnList.add(tempTransaction);*

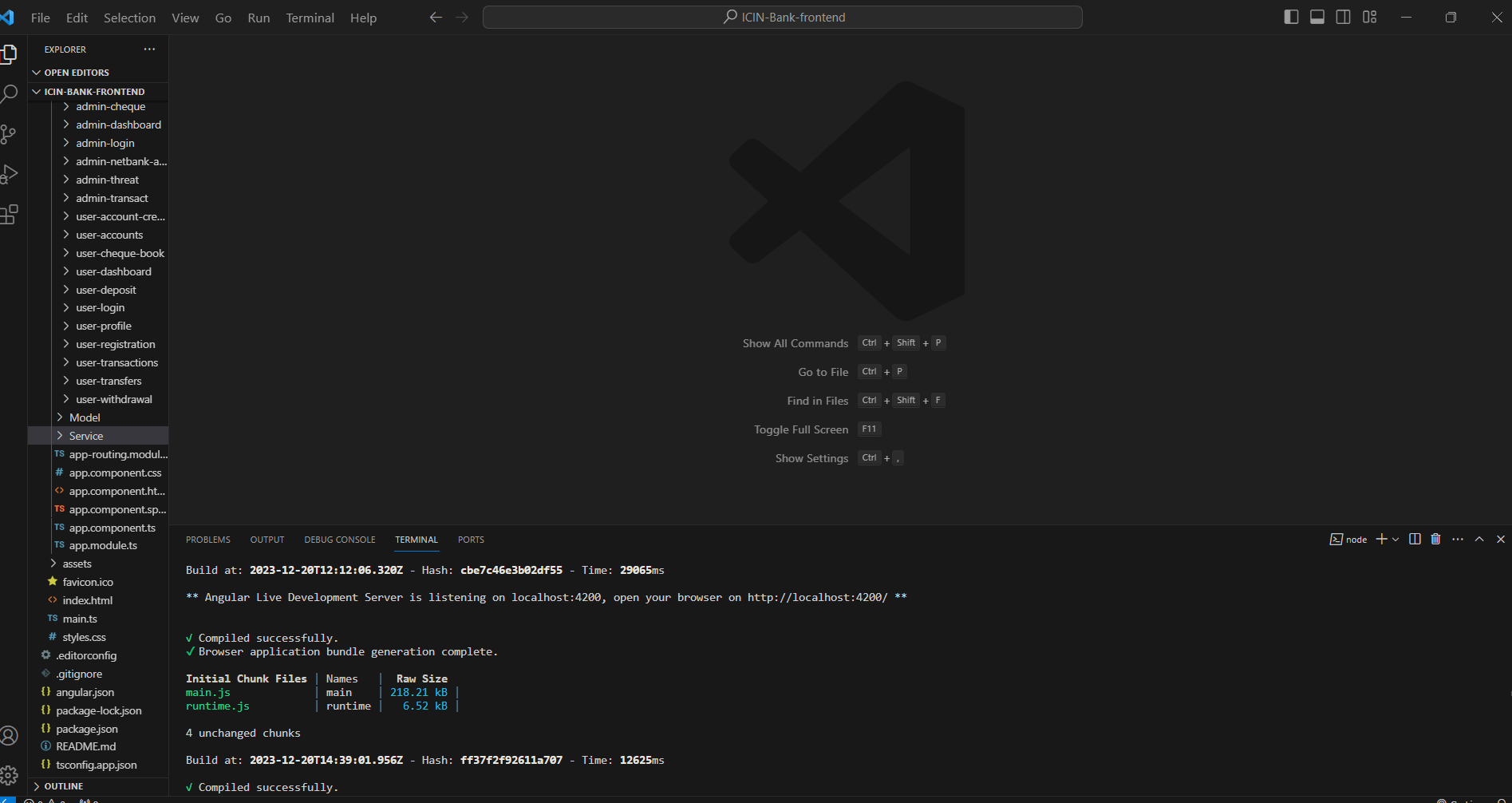
*}*

*return toReturnList;*

*}*

*}*

*FrontEnd*

**