c) Pseudocode

i) GUI subsystem

String loginName;

String loginPassword;

String SQL\_loginName;

String SQL\_loginPassword;

Boolean agreementAcceptance;

Void main{

connectToDatabase();

loginPage();

System.exit(0);

}

void loginPage(){

launch loginPage GUI

wait for input for the username and password

get username and password

Check if user name is valid using verifyLogin()

if yes, launch the user agreement

if no, display error message and keep waiting for input

launch acceptance agreement

when user agrees, launch mainMenu()

}

void mainMenu(){

launch mainMenu GUI

if user clicks “show balance”, display balance

if user clicks “transfer money”, transfer money to stated account

if user clicks “show transaction history”, transaction history is displayed

if user clicks “show interest earnings”. Interest earnings are showed

if user clicks “log out”, user is logged out and the logoutpage appears

}

void logoutPage(){

launch logoutPage

display log out message

start a timer for 10 seconds

after timer expires, close application

}

ii) Session Manager

Connection SQLconnection;

Statement statement;

ResultSet resultSet;

String hashPassword(loginPassword){

Hash password according to algorithm

}

void connectToDatabase(SQL\_loginName, SQL\_loginPassword){

set up connection with connectionURL, SQL\_loginName, and SQL\_loginPassword;

}

Boolean verifyLogin(loginName, loginPassword){

Create connection

Execute statement

If record is found, return true

If record is not found, return false

}

String getBalance(loginName){

Create connection

Execute statement

Return balance

}

String getTransactionHistory(loginName){

Create connection

Execute statement

Format string

Return transaction history

}

String transferMoney(loginName, transferringAccountType, receivingAccountType, transferAmount){

Create connection

Get balance of both accounts

Account for the transfer

Execute statement updating database with new values

Return confirmation of transfer

}

String calculateInterest(loginName, accountType){

Get balance of account

Calculate interest based on account type and balance using Interest Calculator

Return the string with the information

}

iii) Interest Calculator

void interestCalculator(){

get all active accounts from database

get all balances for savings

loop through all savings accounts and calculate interest using calculateInterest()

get all balances for checking

loop through all checking accounts and calculate interest using calculateInterest()

}

double calculateInterest(loginName, accountType, balance){

get transaction history for last 30 days

find average balance of the last 30 days

calculate interest based on the average balance of the last 30 days

return interest amount

}