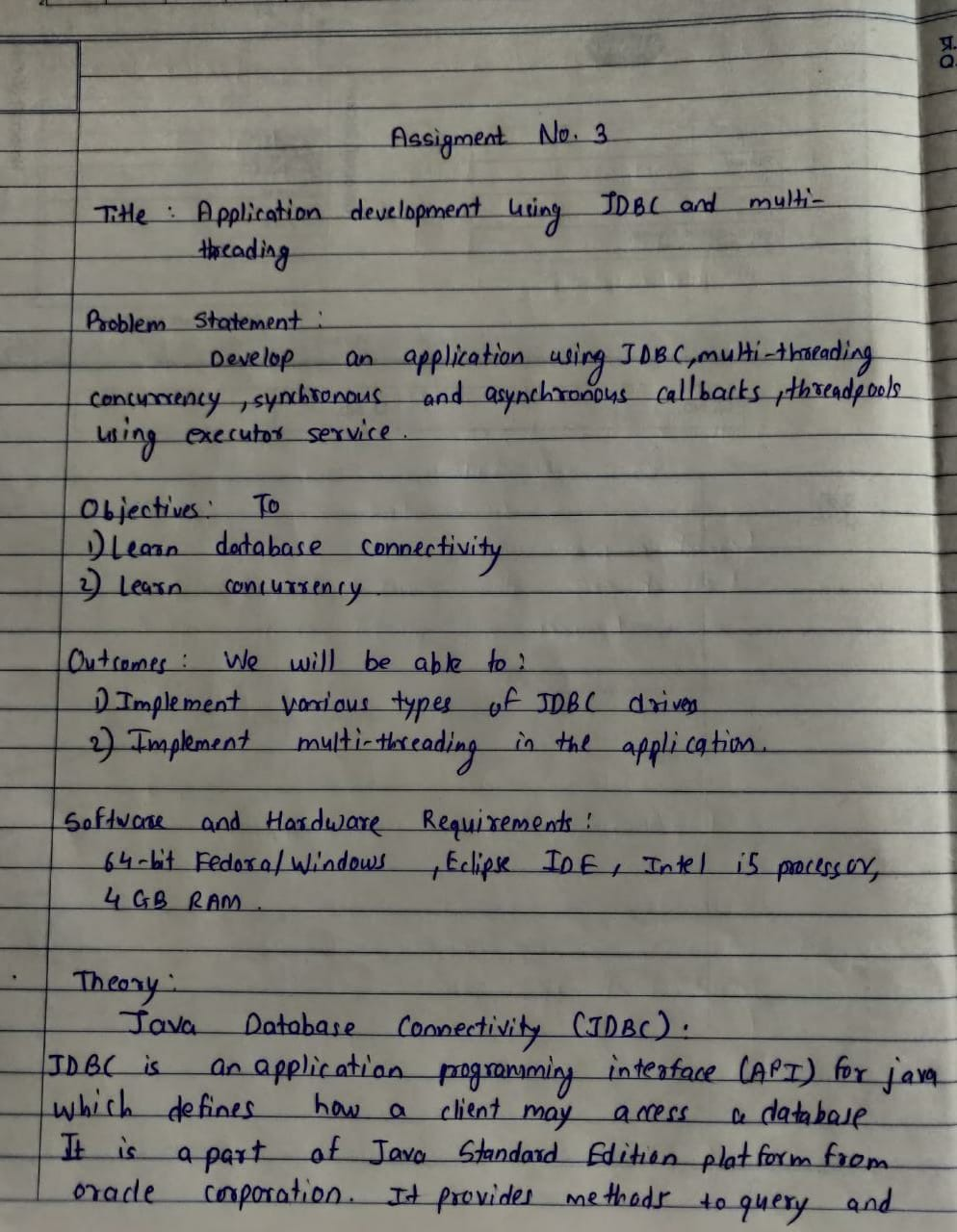
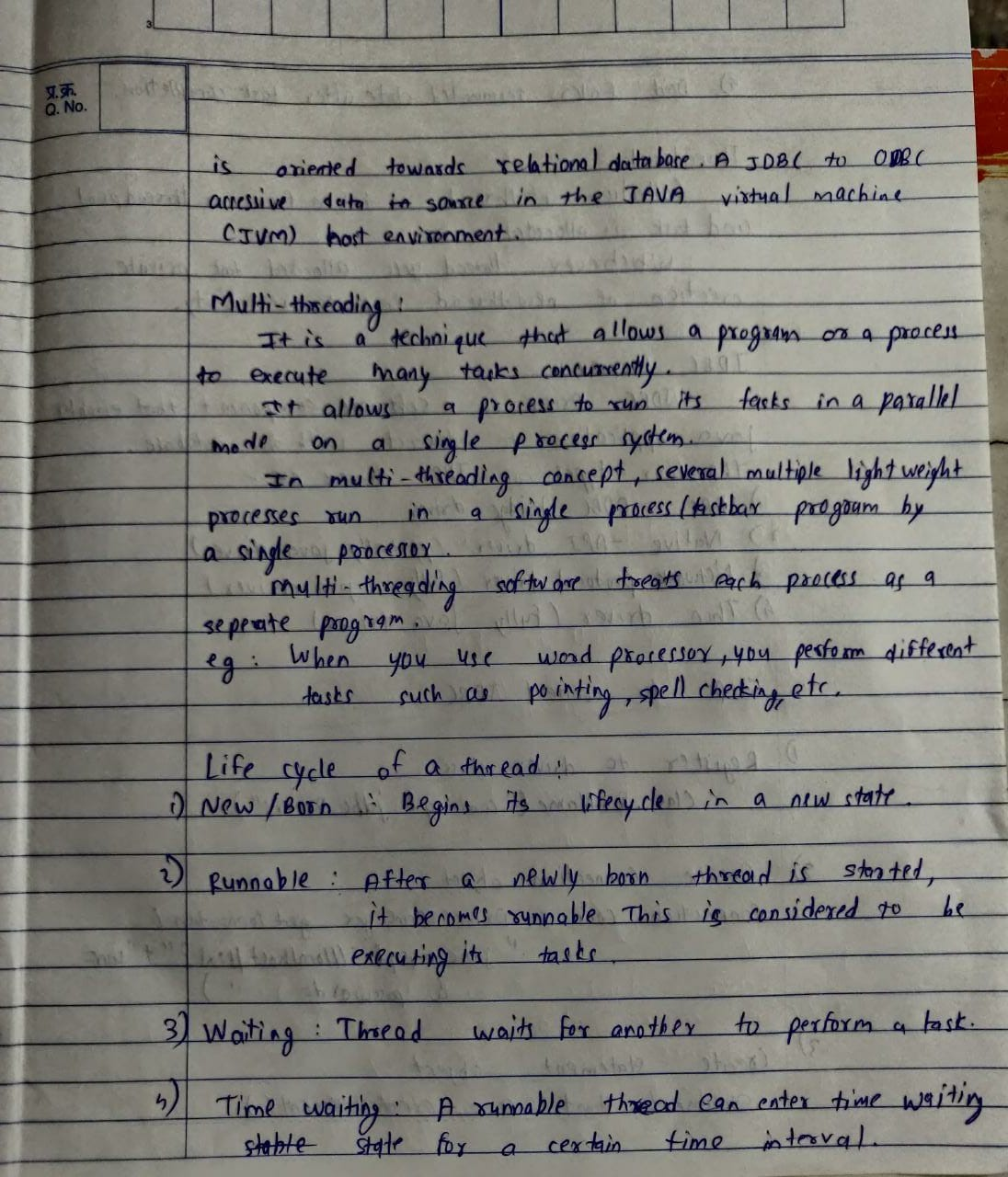
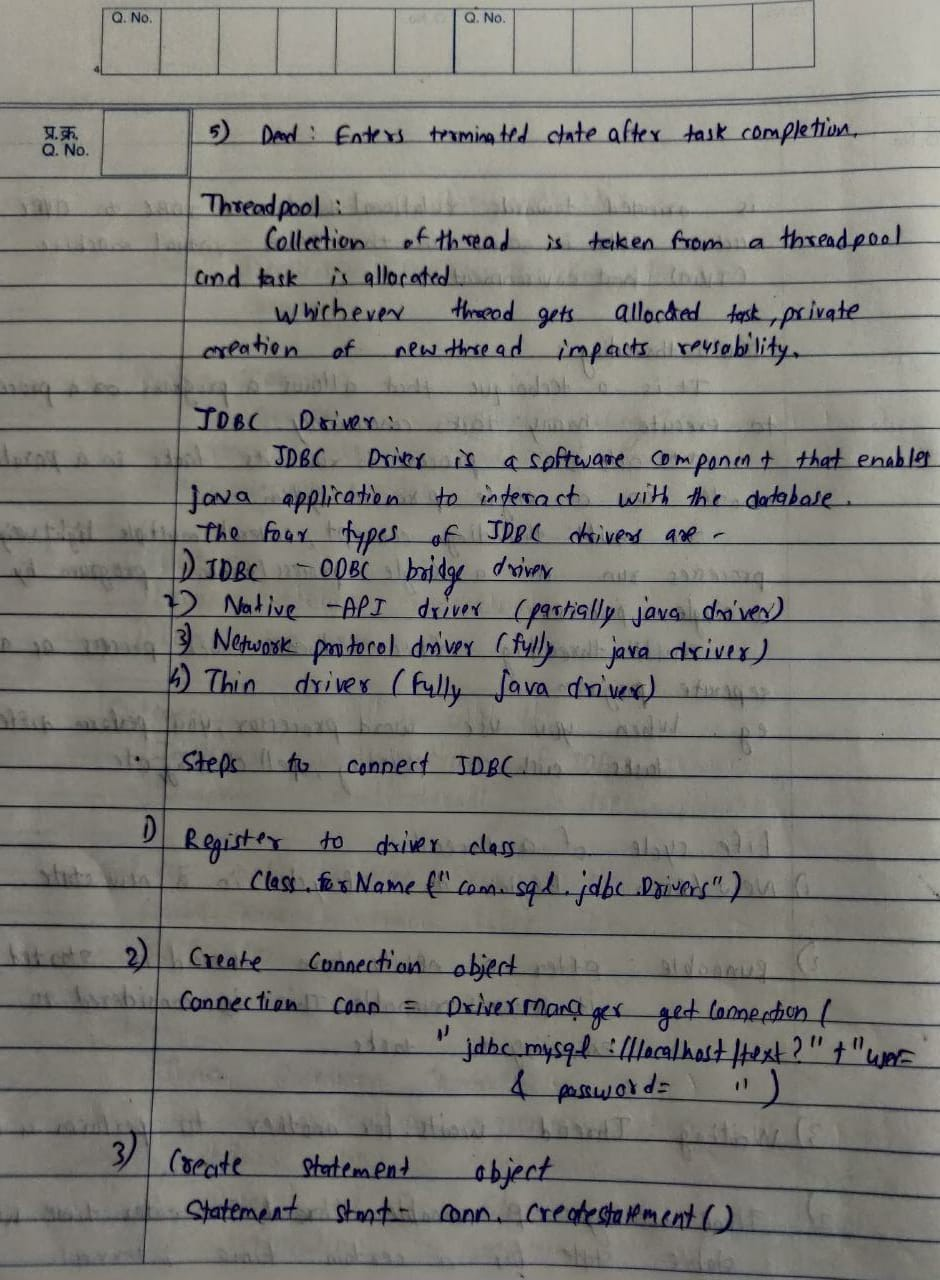
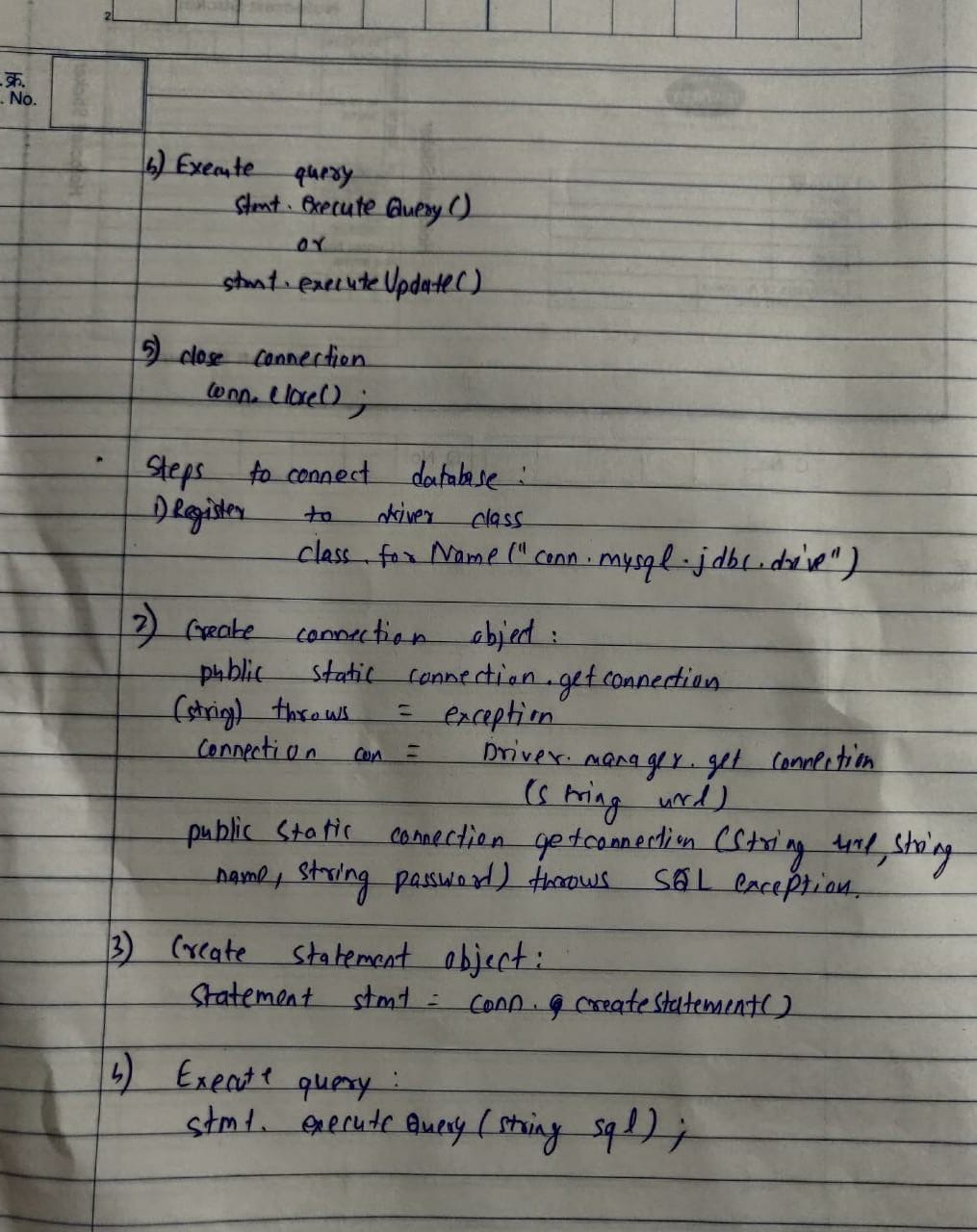
***Write-Up***









***Sample Code:***

***// Server.java***

package multithread;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

import java.sql.\*;

import java.util.concurrent.ExecutorService;

import java.util.concurrent.Executors;

public class Server {

private static final int PORT = 9169;

private static final ExecutorService pool = Executors.newFixedThreadPool(5);

public static void main(String[] args) throws IOException, SQLException, ClassNotFoundException {

String name, pass, url;

Connection con;

Class.forName("com.mysql.cj.jdbc.Driver");

url = "jdbc:mysql://localhost:3306/moviedb";

name = "root";

pass = "mysql";

con = DriverManager.getConnection(url, name, pass);

ServerSocket listener = new ServerSocket(PORT);

while(true){

System.out.println("Waiting For Client");

Socket client = listener.accept();

System.out.println("Client Accepted");

ClientHandler clientThread = new ClientHandler(client,con);

pool.execute(clientThread);

}

}

}

***//ClientHandler.java***

package multithread;

import java.sql.\*;

import java.io.\*;

import java.net.Socket;

import java.util.ArrayList;

import java.util.Random;

public class ClientHandler implements Runnable {

private Socket client;

private BufferedReader in;

private PrintWriter out;

Connection con;

ArrayList<String> usernames = new ArrayList<>();

Statement stmt;

ArrayList<String> movieswatched = new ArrayList<>();

ArrayList<String> moviesnotwatched = new ArrayList<>();

boolean contain,passeq;

public ClientHandler(Socket clientSocket, Connection con) throws IOException, SQLException {

this.client = clientSocket;

this.con = con;

stmt = con.createStatement();

in = new BufferedReader(new InputStreamReader(client.getInputStream()));

out = new PrintWriter(client.getOutputStream(), true);

}

@Override

public void run() {

try {

// Getting the usernames

ResultSet rs = stmt.executeQuery("SELECT username FROM users");

while (rs.next()) {

usernames.add(rs.getString(1));

}

while (true) {

Statement stmt = con.createStatement();

String request = in.readLine();

// Checking for socket close case

if (request.equals("911")) {

try {

in.close();

out.close();

client.close();

} catch (IOException e) {

e.printStackTrace();

}

break;

}

// Log In request

if (request.equals("1")) {

String username = in.readLine();

String password = in.readLine();

// Check if username exists

contain = usernames.contains(username);

if (contain) {

// Username existed. check for password

out.println("1");

rs = stmt.executeQuery("SELECT password FROM users WHERE username=\"" + username + "\";");

rs.next();

passeq = password.equals(rs.getString(1));

if (!passeq) {

// Password didn't match

out.println("0");

} else {

// Password matched

out.println("1");

while (true) {

String clientresp;

clientresp = in.readLine();

// Sign out case

if (clientresp.equals("3")) break;

// Rewatch Case

else if (clientresp.equals("2")) {

rs = stmt.executeQuery("SELECT MovieName FROM movietable WHERE " + username + "=\"Y\";");

while (rs.next()) {

movieswatched.add(rs.getString(1));

}

String randomMovie;

String clientres;

while (!movieswatched.isEmpty()) {

int index = new Random().nextInt(movieswatched.size());

randomMovie = movieswatched.get(index);

out.println(randomMovie);

clientres = in.readLine();

if (clientres.equals("1")) break;

movieswatched.remove(index);

}

if (movieswatched.isEmpty()) out.println("404");

}

// New Recommendation Case

else {

rs = stmt.executeQuery("SELECT MovieName FROM movietable WHERE " + username + "=\"N\";");

while (rs.next()) {

moviesnotwatched.add(rs.getString(1));

}

String randomMovie = "";

String clientres;

int flag = 0;

while (!moviesnotwatched.isEmpty()) {

int index = new Random().nextInt(moviesnotwatched.size());

randomMovie = moviesnotwatched.get(index);

out.println(randomMovie);

clientres = in.readLine();

if (clientres.equals("1")) {

flag = 1;

break;

}

moviesnotwatched.remove(index);

}

if (moviesnotwatched.isEmpty()) out.println("404");

if (flag == 1) {

stmt.executeUpdate("UPDATE movietable set " + username + "=\"Y\" where MovieName=\"" + randomMovie + "\"");

}

}

}

}

}

// Username didn't match

else {

out.println("0");

}

}

// Sign Up Case

if (request.equals("2")) {

String usernamesignup = in.readLine();

String passwordsignup = in.readLine();

if (usernames.contains(usernamesignup)) {

// Username Already Exists

out.println("0");

} else {

usernames.add(usernamesignup);

stmt.executeUpdate("ALTER TABLE movietable ADD " + usernamesignup + " varchar(1);");

stmt.executeUpdate("UPDATE movietable SET " + usernamesignup + "=\"N\"");

stmt.executeUpdate("INSERT INTO users values(\"" + usernamesignup + "\",\"" + passwordsignup + "\");");

out.println("1");

}

}

}

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

in.close();

out.close();

client.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

***//Client.java***

package multithread;

import java.io.\*;

import java.net.Socket;

public class Client {

public static void main(String[] args) throws IOException {

Socket socket = new Socket("127.0.0.1", 9169);

BufferedReader keyboard = new BufferedReader(new InputStreamReader(System.in));

BufferedReader reader = new BufferedReader(new InputStreamReader(socket.getInputStream()));

PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

String choice;

do {

System.out.println("-----WELCOME-----\n1.Login\t\t2.Sign Up\t\t911.Close the Socket");

choice = keyboard.readLine();

label:

switch (choice) {

case "1":

out.println(1);

System.out.print("Enter username: ");

out.println(keyboard.readLine());

System.out.print("Enter password: ");

out.println(keyboard.readLine());

String usernameresponse = reader.readLine();

if (usernameresponse.equals("0"))

System.out.println("Account with this username doesn't exist!");

else {

String passwordresponse = reader.readLine();

if (passwordresponse.equals("0"))

System.out.println("Incorrect Password");

else {

System.out.println("Logged In Successfully");

String loginchoice;

while (true) {

System.out.println("\n1. A new movie recommendation.\t\t2. A rewatch recommendation\t\t3.Sign out and exit");

loginchoice = keyboard.readLine();

switch (loginchoice) {

case "3":

out.println("3");

choice = "911";

break label;

case "2":

out.println("2");

String rewatch;

String rewatchchoice;

while (true) {

rewatch = reader.readLine();

if (rewatch.equals("404")) {

System.out.println("We're out of movies :(");

break;

}

System.out.print("\nThe rewatch recommendation is: ");

System.out.println(rewatch);

System.out.print("Would you like a different recommendation?(y/n) ");

rewatchchoice = keyboard.readLine();

if (rewatchchoice.equals("y")) out.println("0");

else {

out.println("1");

break;

}

}

break;

case "1":

out.println("1");

String recomm;

String recommchoice;

while (true) {

recomm = reader.readLine();

if (recomm.equals("404")) {

System.out.println("We're out of movies :(");

break;

}

System.out.print("\nThe movie recommendation is: ");

System.out.println(recomm);

System.out.print("Would you like a different recommendation?(y/n) ");

recommchoice = keyboard.readLine();

if (recommchoice.equals("y")) out.println("0");

else {

out.println("1");

System.out.println("We hope you enjoy the movie!!!!");

break;

}

}

break;

}

}

}

}

break;

case "2":

out.println(2);

System.out.print("Enter username: ");

out.println(keyboard.readLine());

System.out.print("Enter password: ");

out.println(keyboard.readLine());

String serverresp = reader.readLine();

if (serverresp.equals("0")) {

System.out.println("Username Already Exists!");

} else {

System.out.println("Account Created Successfully!!");

}

break;

case "911":

out.println(911);

keyboard.close();

reader.close();

out.close();

socket.close();

break;

default:

System.out.println("Enter Valid Input");

}

} while (!choice.equals("911"));

out.println(911);

keyboard.close();

reader.close();

out.close();

socket.close();

}

}

***Outputs***

