**JAVA SWING BASED – DIGITALIMAGINARY – SQL CONNECTIVITY USING JDBC**

*A Report*

*Submitted in partial fulfillment of the Requirements*

*for the COURSE*

**DATABASE MANAGEMENT SYSTEMS**

**By**

### S.BAGAWAN REDDY

### <1602-21-737-012>

**Under the guidance of Ms B. Leelavathy**



### Department of Information Technology Vasavi College of Engineering (Autonomous) (Affiliated to Osmania University) Ibrahimbagh, Hyderabad-31

**2022-2023**

BONAFIDE CERTIFICATE

This is to certify that this project report titled ***‘DIGITAL IMAGINARY’***

is a project work of **S. Bagawan Reddy** bearing roll no. 1602-21-737-012 who carried out this project under my supervision in the IV semester for the academic year 2022- 2023.

Signature Signature

External Examiner Internal Examiner

# ABSTRACT

The E-learning project aims to provide a web-based platform for students to access course materials, take mock quizzes, track their progress, and to access question bank. The system includes a user table to manage user information. The system also includes a Progress table to track the user's progress in the course. It also includes the list of students who solved questions from question bank. This project provides a comprehensive solution for online learning, which is especially relevant in the current context where remote learning is becoming more popular. It allows users to learn at their own pace and provides tools to track their progress and improve their learning outcomes.

# Requirement Analysis

## List of Tables:

* Students
* Notes
* Mock\_Quiz
* Questions
* Scores

## List of Attributes with their Domain Types:

**1.Students Table:**

|  |  |  |
| --- | --- | --- |
| Attribute | Domain | CONSTARINT |
| Student\_id | NUMBER | Primary\_Key |
| Student\_name | VARCHAR | Not Null |
| Class | VARCHAR |  |
| Password | VARCHAR | Not Null |

**2.Notes Table:**

|  |  |  |
| --- | --- | --- |
| Attribute | Domain | CONSTRAINT |
| Chapter\_no | NUMBER | Primary\_Key |
| Chapter\_name | VARCHAR | Not Null |

|  |  |  |
| --- | --- | --- |
| Attribute | Domain | Constraint |
| Quiz\_id | NUMBER | Primary\_Key |
| Duration | TIME |  |

**3.Mock\_Quiz:**

**4.Questions:**

|  |  |  |
| --- | --- | --- |
| Attribute | Domain | CONSTRAINT |
| Question\_no | NUMBER | Primary\_Key |
| Question | VARCHAR | Not Null |

**5.Scores:**

|  |  |  |
| --- | --- | --- |
| Attribute | Domain | CONSTARINT |
| Student\_id | NUMBER | Foreign\_Key |
| Quiz\_id | NUMBER | Foreign\_Key |
| Marks | NUMBER | Not Null |

# AIM AND PRIORITY OF THE PROJECT

Digital Imaginary is an E-Learning platform where student can login with their roll number and password to view the required documents and can ask questions to the faculty. There is another feature to write quiz and they can watch their respective scores for each attempt.

# ARCHITECTURE AND TECHNOLOGY

### Software used:

Java, Oracle 11g Database, Java SE version 14, Run SQL.

### Java SWING:

**Java SWING** is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) - an API for providing a graphical user interface (GUI) for Java programs.

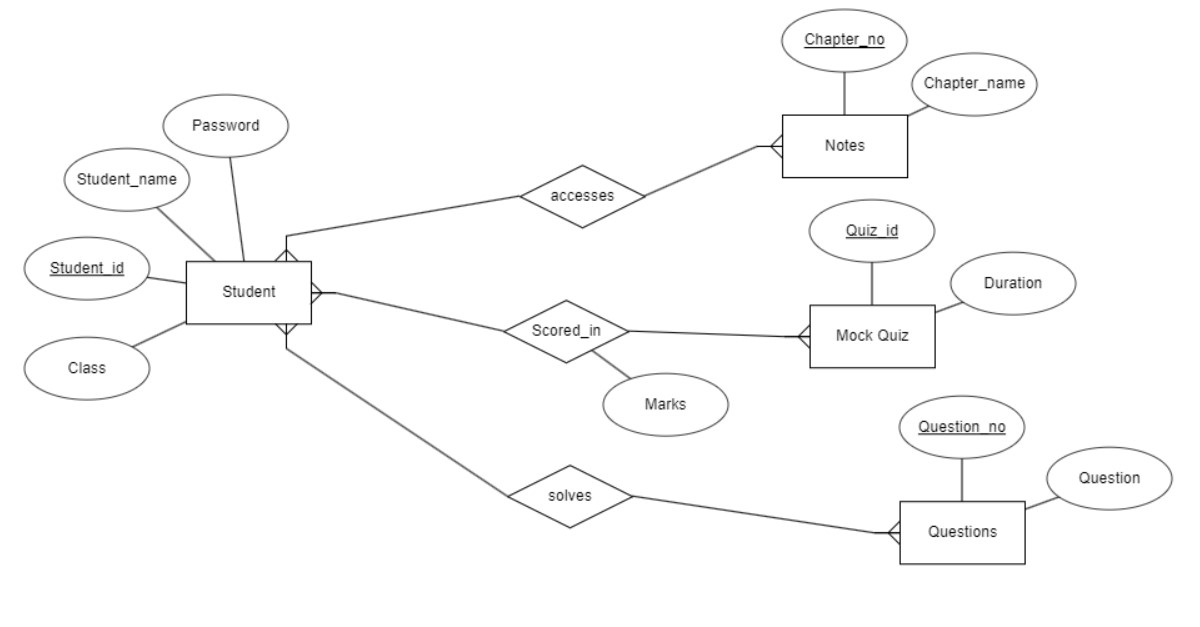
Swing was developed to provide a more sophisticated set of GUI components than the earlier AWT. Swing provides a look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

### SQL:

Structure Query Language(SQL) is a database query language used for storing and managing data in **Relational** DBMS. SQL was the first commercial language introduced for E.F Codd's Relational model of database. Today almost all RDBMS (MySql, Oracle, Infomix, Sybase, MS Access) use **SQL** as the standard database query language. SQL is used to perform all types of data operations in RDBMS.

**DESIGN**

### Entity Relationship Diagram



1.Creating table for students with constraints :

**QUERY:** create table students(

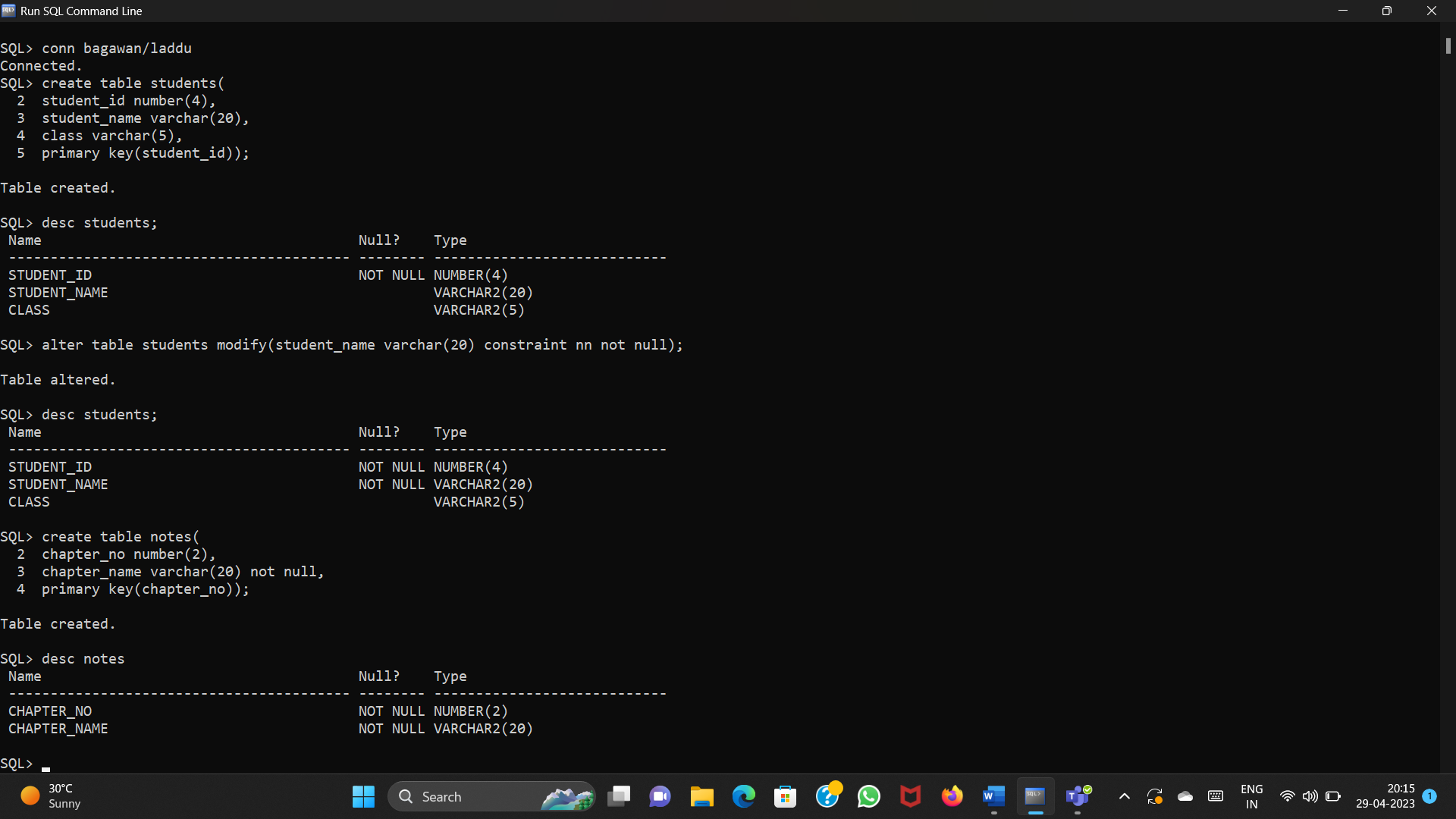
2 student\_id number(4),

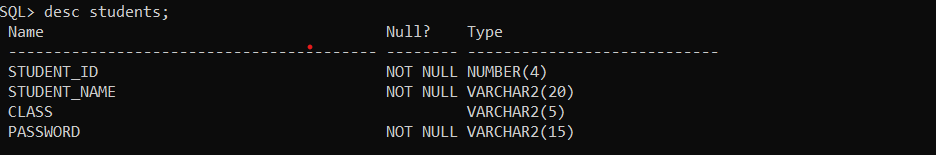
3 student\_name varchar(20),

4 class varchar(5),

5 password varchar(15),

6 primary key(student\_id));





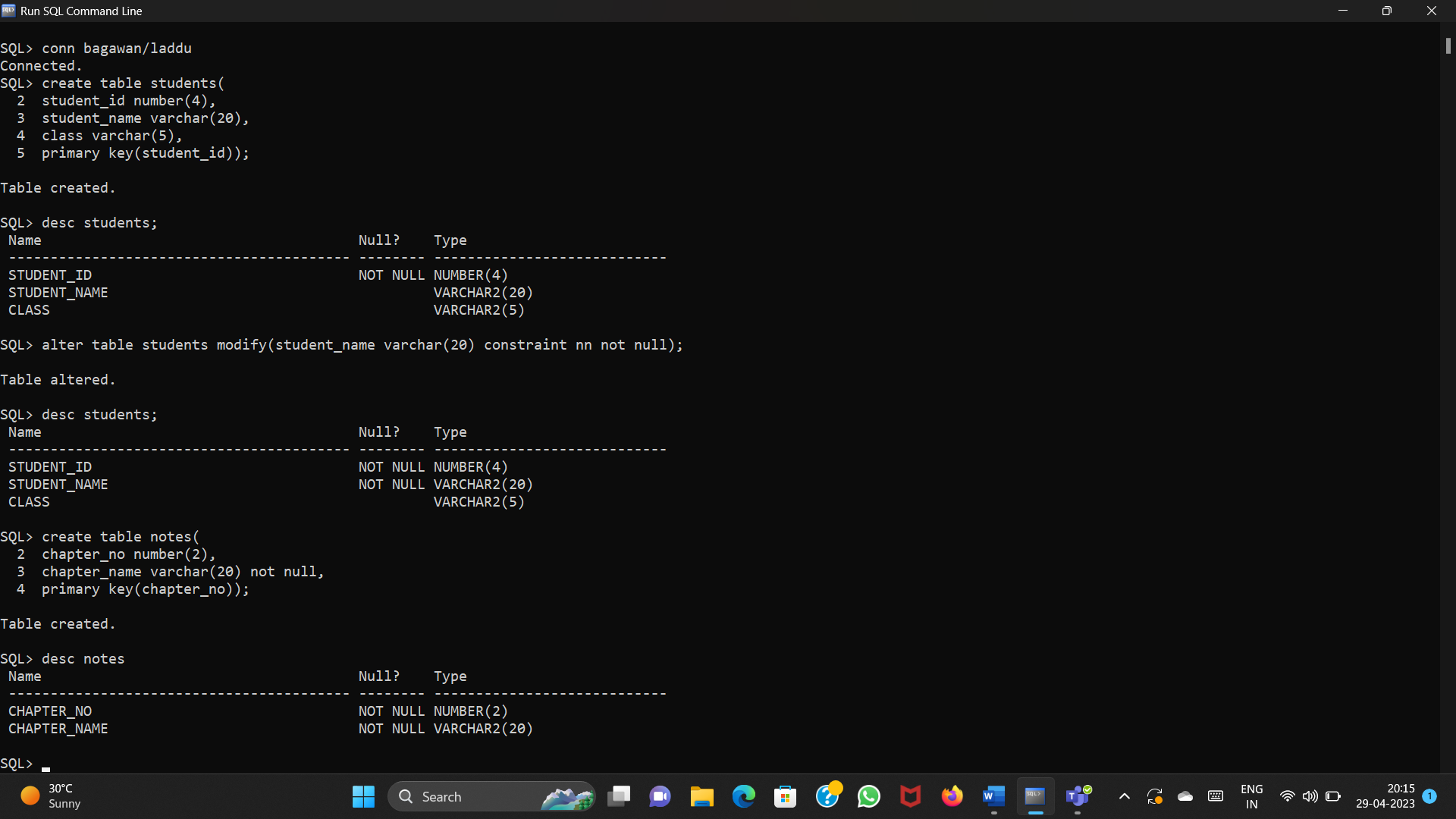
2. Creating notes table:

**QUERY:** create table notes(

2 chapter\_no number(2),

3 chapter\_name varchar(20) not null,

4 primary key(chapter\_no));

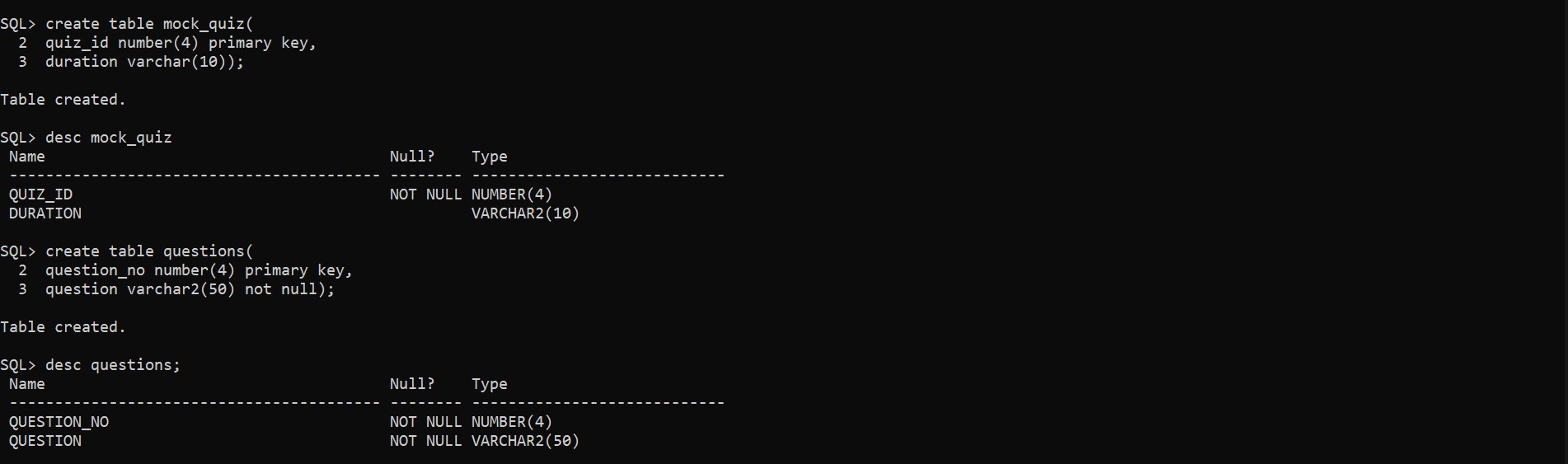


3.Creating mock\_quiz table:

**QUERY:** create table mock\_quiz(

2 quiz\_id number(4) primary key,

3 duration varchar(10));

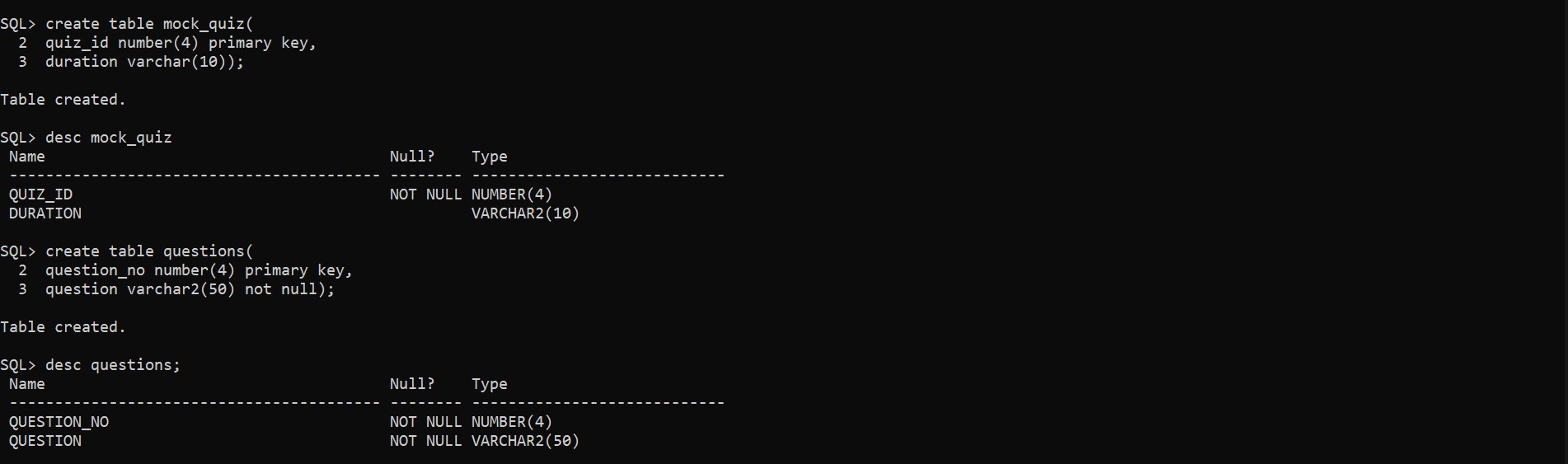


4.Creating Questions table:

**QUERY:** create table questions(

2 question\_no number(4) primary key,

3 question varchar2(50) not null);



5.Creating scores table:

**QUERY:** create table scores(

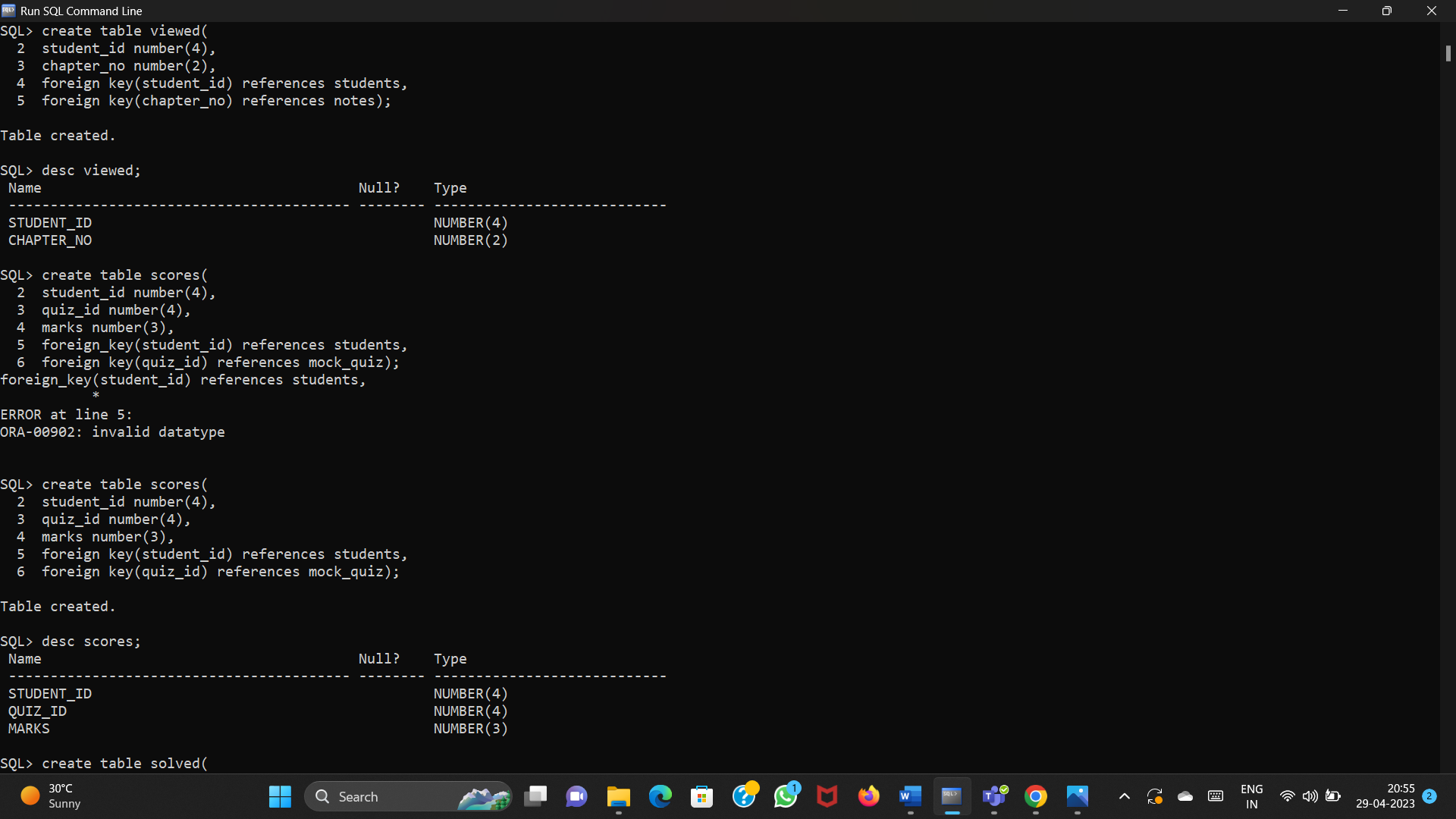
2 student\_id number(4),

3 quiz\_id number(4),

4 marks number(3),

5 foreign key(student\_id) references students,

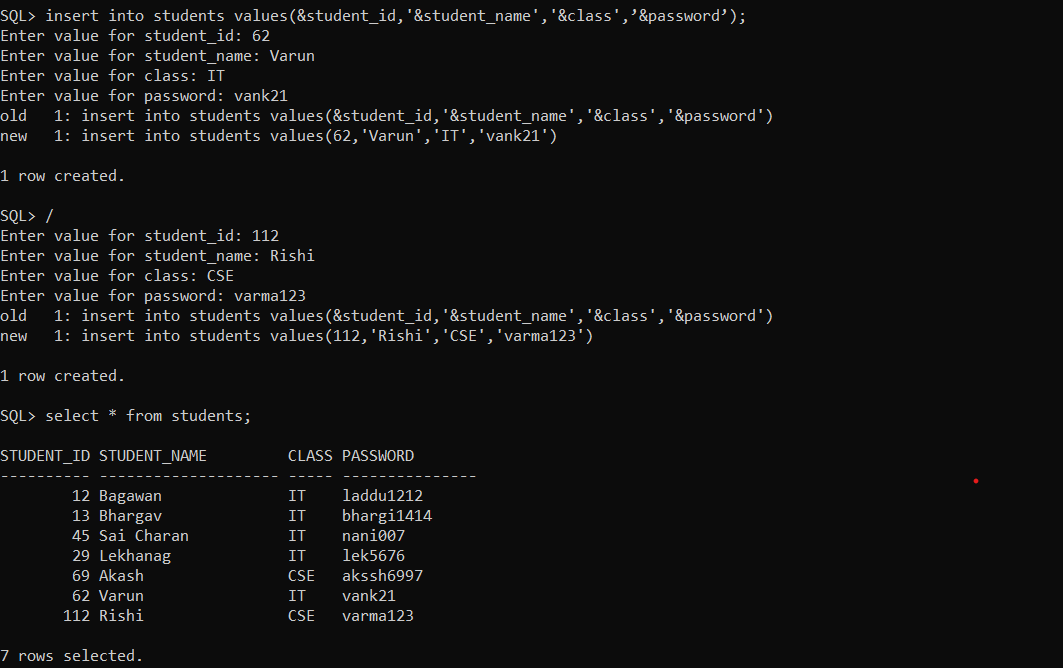
6 foreign key(quiz\_id) references mock\_quiz);



**DML COMMANDS:**

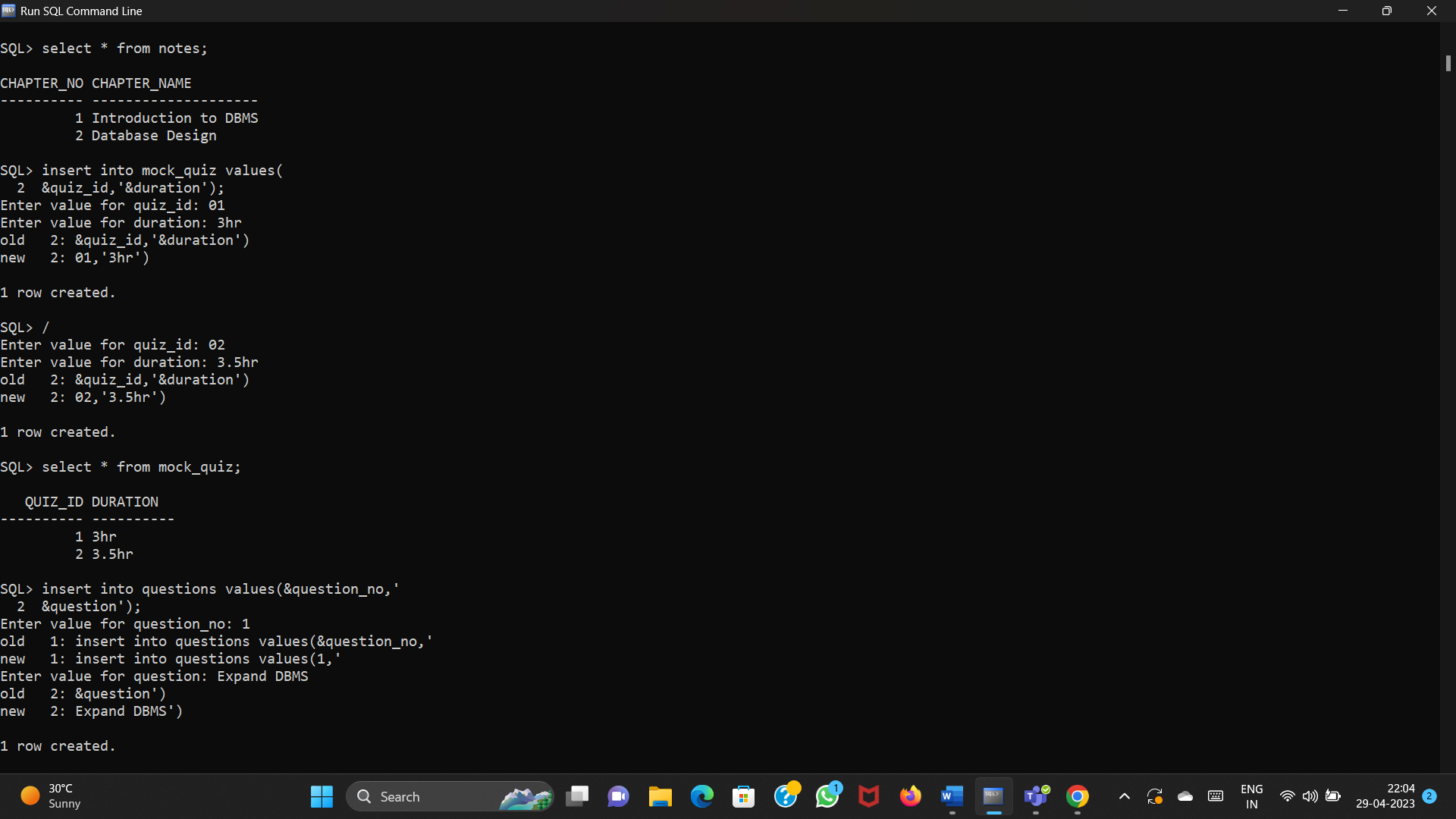
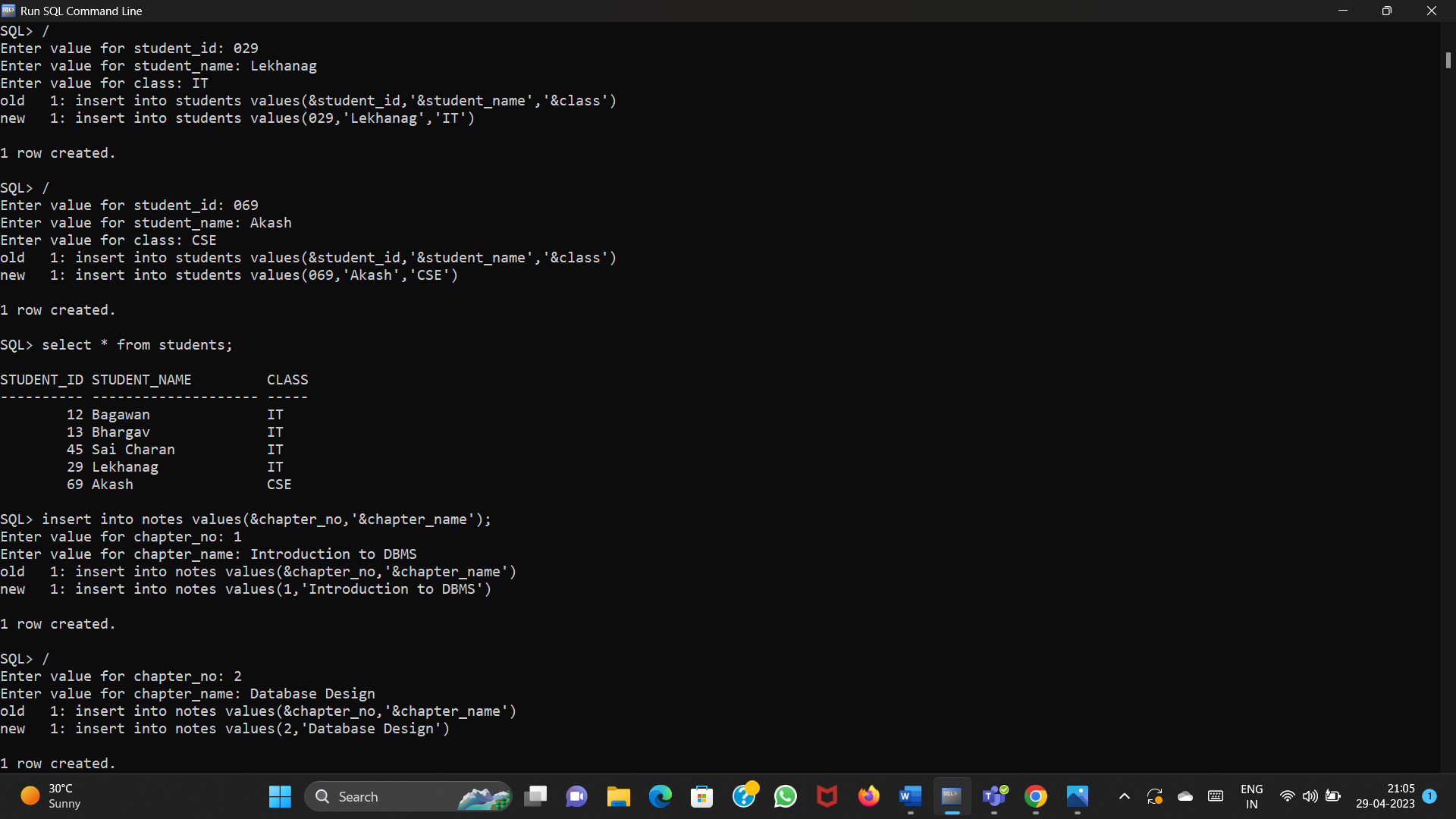
1.Insert values into students:

**QUERY:** insert into students values(&student\_id,'&student\_name','&class',’&password’);



2.Insert values into notes:

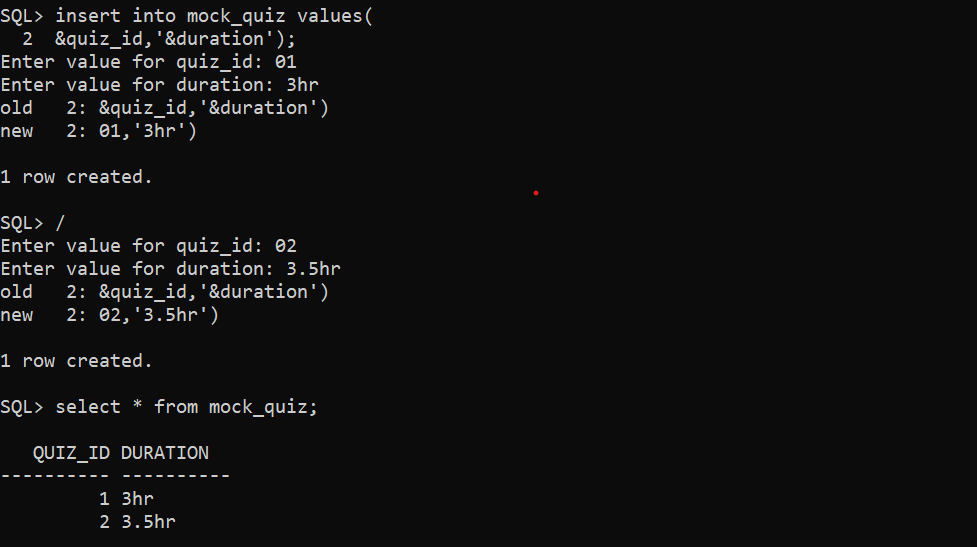
**QUERY:** insert into notes values(&chapter\_no,'&chapter\_name');



3.Insert values into mock\_quiz:

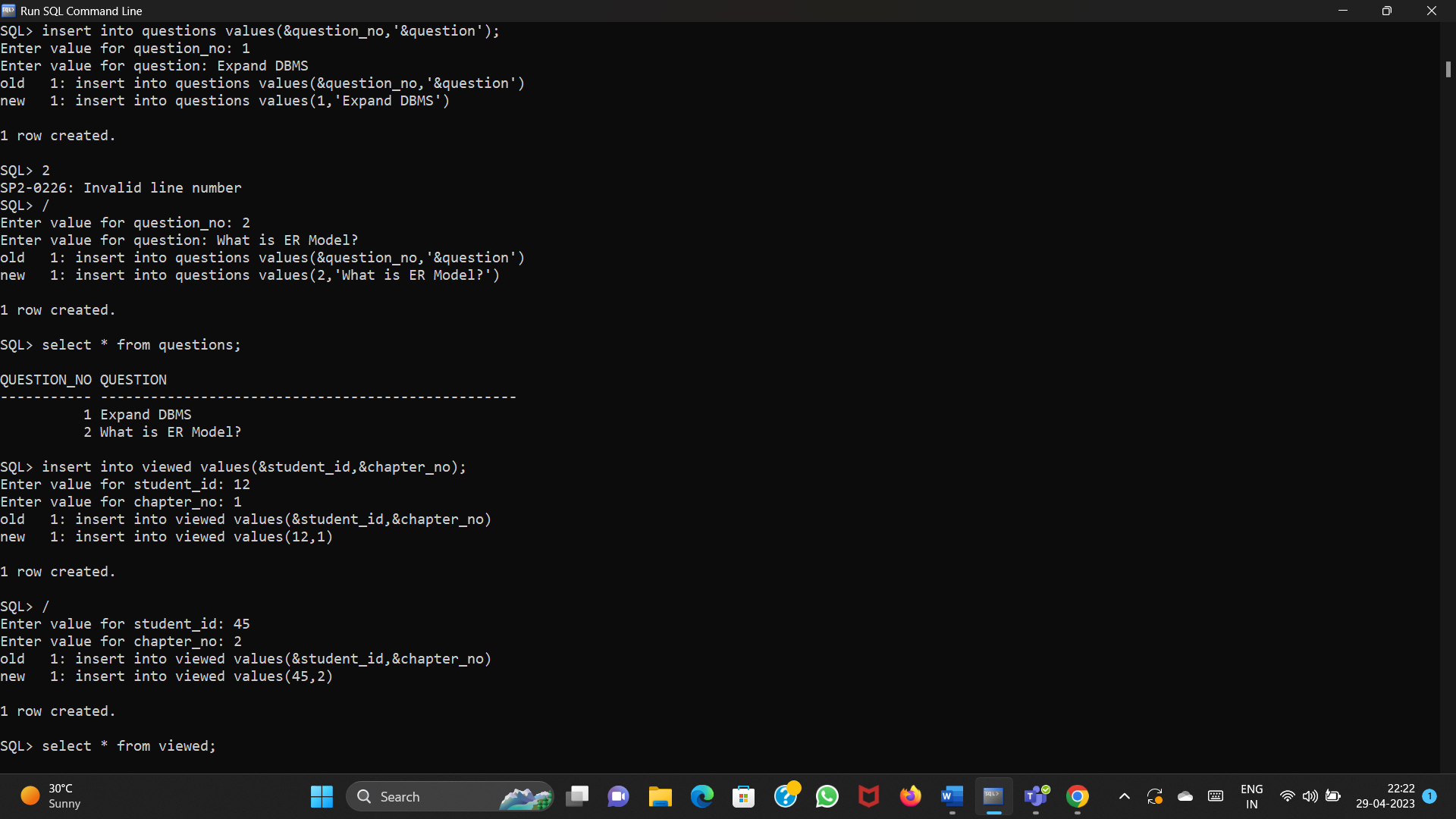
**QUERY:** insert into mock\_quiz values(

2 &quiz\_id,'&duration');



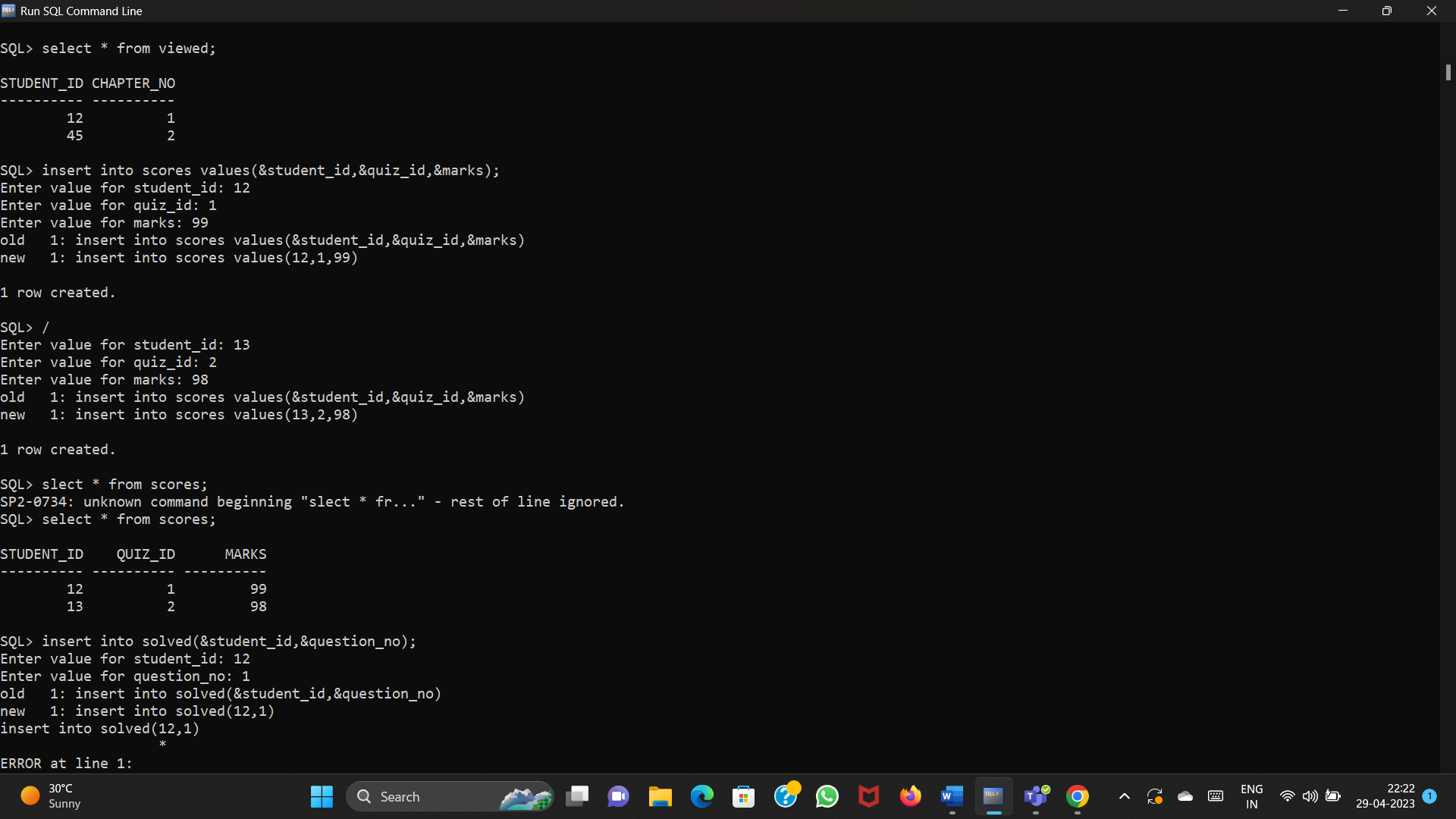
4.Insert values into questions:

**QUERY:** insert into questions values(&question\_no,'&question');



5.Insert values into scores:

**QUERY:** insert into scores values(&student\_id,&quiz\_id,&marks);



# IMPLEMENTATION

### JAVA-SQL Connectivity using JDBC:

**Java Database Connectivity (JDBC)** is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database and is oriented towards relational databases.

The connection to the database can be performed using Java programming (JDBC API) as:

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

int a=stmt.executeUpdate("insert into students values("+playerId+",'"+playerName+"','"+position+"','"+password+"')");

if(a>0)

JOptionPane.showMessageDialog(null, "Sign Up Successful", "Success", JOptionPane.INFORMATION\_MESSAGE);

else

JOptionPane.showMessageDialog(null, "Enter Valid Details", "Error", JOptionPane.ERROR\_MESSAGE);

// Close the connection

con.close();

} catch (ClassNotFoundException c) {

System.err.println("Failed to load JDBC driver: " + c.getMessage());

} catch (SQLException c) {

System.err.println("Failed to connect to the database: " + c.getMessage());

}

}

**Front-end Programs (User Interfaces) Home Page:**

**1.Login and Signup Page:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

public class Digital\_Imaginary extends JFrame {

private JTextField signUpPlayerIdField;

private JTextField signUpPlayerNameField;

private JTextField signUpPositionField;

private JPasswordField signUpPasswordField;

private JTextField loginPlayerIdField;

private JPasswordField loginPasswordField;

// Database to store player information (In-memory storage for simplicity)

//private Map<String, Player> playerDatabase;

public Digital\_Imaginary() {

// Initialize the database

//playerDatabase = new HashMap<>();

setTitle("Sign Up/Login");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(400, 300);

setLocationRelativeTo(null);

// Create panels for sign up and login components

JPanel signUpPanel = createSignUpPanel();

JPanel loginPanel = createLoginPanel();

// Create tabbed pane to switch between sign up and login

JTabbedPane tabbedPane = new JTabbedPane();

tabbedPane.addTab("Sign Up", signUpPanel);

tabbedPane.addTab("Login", loginPanel);

// Add tabbed pane to the frame

getContentPane().add(tabbedPane);

}

private JPanel createSignUpPanel() {

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(6, 2));

// Sign Up components

JLabel signUpPlayerIdLabel = new JLabel("Student ID:");

signUpPlayerIdField = new JTextField(20);

JLabel signUpPlayerNameLabel = new JLabel("Student Name:");

signUpPlayerNameField = new JTextField(20);

JLabel signUpPositionLabel = new JLabel("Class:");

signUpPositionField = new JTextField(20);

JLabel signUpPasswordLabel = new JLabel("Password:");

signUpPasswordField = new JPasswordField(20);

JButton signUpButton = new JButton("Sign Up");

// Sign Up button action listener

signUpButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String playerId = signUpPlayerIdField.getText();

String playerName = signUpPlayerNameField.getText();

String position = signUpPositionField.getText();

String password = new String(signUpPasswordField.getPassword());

if (playerId.isEmpty() || playerName.isEmpty() || position.isEmpty() || password.isEmpty()) {

JOptionPane.showMessageDialog(null, "Please fill in all the fields", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

else

{

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

int a=stmt.executeUpdate("insert into students values("+playerId+",'"+playerName+"','"+position+"','"+password+"')");

if(a>0)

JOptionPane.showMessageDialog(null, "Sign Up Successful", "Success", JOptionPane.INFORMATION\_MESSAGE);

else

JOptionPane.showMessageDialog(null, "Enter Valid Details", "Error", JOptionPane.ERROR\_MESSAGE);

// Close the connection

con.close();

} catch (ClassNotFoundException c) {

System.err.println("Failed to load JDBC driver: " + c.getMessage());

} catch (SQLException c) {

System.err.println("Failed to connect to the database: " + c.getMessage());

}

}

//JOptionPane.showMessageDialog(null, "Sign In Successful", "Success", JOptionPane.INFORMATION\_MESSAGE);

// Clear the sign up fields

signUpPlayerIdField.setText("");

signUpPlayerNameField.setText("");

signUpPositionField.setText("");

//signUpContactInfoField.setText("");

signUpPasswordField.setText("");

//JOptionPane.showMessageDialog(null, "Sign up successful!");

}

});

// Add components to the sign up panel

panel.add(signUpPlayerIdLabel);

panel.add(signUpPlayerIdField);

panel.add(signUpPlayerNameLabel);

panel.add(signUpPlayerNameField);

panel.add(signUpPositionLabel);

panel.add(signUpPositionField);

panel.add(signUpPasswordLabel);

panel.add(signUpPasswordField);

panel.add(new JLabel()); // Empty label for spacing

panel.add(signUpButton);

return panel;

}

private JPanel createLoginPanel() {

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(3, 2));

// Login components

JLabel loginPlayerIdLabel = new JLabel("Student ID:");

loginPlayerIdField = new JTextField(20);

JLabel loginPasswordLabel = new JLabel("Password:");

loginPasswordField = new JPasswordField(20);

JButton loginButton = new JButton("Login");

// Login button action listener

loginButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String playerId = loginPlayerIdField.getText();

String password = new String(loginPasswordField.getPassword());

// Retrieve player object from the database

if (playerId.isEmpty() || password.isEmpty()) {

JOptionPane.showMessageDialog(null, "Please enter ID and password", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

else {

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

ResultSet rs=stmt.executeQuery("select password from students where student\_id="+playerId+"and password='"+password+"'");

if(rs.next())

{

JOptionPane.showMessageDialog(null, "Login Successful", "Success", JOptionPane.INFORMATION\_MESSAGE);

setVisible(false);

new Home(Integer.parseInt(playerId));

}

else

JOptionPane.showMessageDialog(null, "Enter Valid Details", "Error", JOptionPane.ERROR\_MESSAGE);

// Close the connection

con.close();

} catch (ClassNotFoundException c) {

System.err.println("Failed to load JDBC driver: " + c.getMessage());

} catch (SQLException c) {

System.err.println("Failed to connect to the database: " + c.getMessage());

}

}

// Clear the login fields

loginPlayerIdField.setText("");

loginPasswordField.setText("");

}

});

// Add components to the login panel

panel.add(loginPlayerIdLabel);

panel.add(loginPlayerIdField);

panel.add(loginPasswordLabel);

panel.add(loginPasswordField);

panel.add(new JLabel()); // Empty label for spacing

panel.add(loginButton);

return panel;

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

public void run() {

new Digital\_Imaginary().setVisible(true);

}

});

}

}

/\*

class Player {

private String playerId;

private String playerName;

private String position;

private String contactInfo;

private String password;

public Player(String playerId, String playerName, String position, String contactInfo, String password) {

this.playerId = playerId;

this.playerName = playerName;

this.position = position;

this.contactInfo = contactInfo;

this.password = password;

}

public String getPlayerId() {

return playerId;

}

public String getPlayerName() {

return playerName;

}

public String getPosition() {

return position;

}

public String getContactInfo() {

return contactInfo;

}

public String getPassword() {

return password;

}

\*/

**2.Home Page:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.List;

import java.sql.\*;

public class Home {

private JFrame frame;

private JPanel insertQuestionPanel;

private JTextField questionNumberTextField;

private JTextField questionTextField;

private JPanel insertNotesPanel;

private JTextField chapterNumberTextField;

private JTextField chapterNameTextField;

private JTextArea scoresTextArea;

public int id;

public Home(int x) {

this.id=x;

initialize();

}

private void initialize() {

frame = new JFrame("Digital\_Imaginary");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JMenuBar menuBar = new JMenuBar();

JMenu notesMenu = new JMenu("Notes");

JMenuItem viewNotesItem = new JMenuItem("View Notes");

JMenuItem insertNotesItem = new JMenuItem("Insert Notes");

notesMenu.add(viewNotesItem);

notesMenu.add(insertNotesItem);

JMenu questionsMenu = new JMenu("Questions");

JMenuItem viewQuestionsItem = new JMenuItem("View Questions");

JMenuItem insertQuestionItem = new JMenuItem("Insert Question");

JMenuItem updateQuestionItem = new JMenuItem("Update Question");

JMenuItem deleteQuestionItem = new JMenuItem("Delete Question");

questionsMenu.add(viewQuestionsItem);

questionsMenu.add(insertQuestionItem);

questionsMenu.add(updateQuestionItem);

questionsMenu.add(deleteQuestionItem);

JMenu quizMenu = new JMenu("Quiz");

JMenuItem startQuizItem = new JMenuItem("Quiz1");

quizMenu.add(startQuizItem);

JMenuItem startQuizItem1 = new JMenuItem("Quiz2");

quizMenu.add(startQuizItem1);

JMenu scoresMenu = new JMenu("Scores");

JMenuItem viewScoresItem = new JMenuItem("View Scores");

scoresMenu.add(viewScoresItem);

menuBar.add(notesMenu);

menuBar.add(questionsMenu);

menuBar.add(quizMenu);

menuBar.add(scoresMenu);

startQuizItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Show all the questions

new QuizApplication(id);

}

});

startQuizItem1.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Show all the questions

new QuizApplication1(id);

}

});

viewQuestionsItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Show all the questions

showQuestions();

}

});

insertQuestionItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Create and show the insert question panel

createInsertQuestionPanel();

}

});

updateQuestionItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Create and show the update question panel

createUpdateQuestionPanel();

}

});

deleteQuestionItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Create and show the delete question panel

createDeleteQuestionPanel();

}

});

insertNotesItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Create and show the insert notes panel

createInsertNotesPanel();

}

});

viewNotesItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Create and show the insert notes panel

showNotes();

}

});

viewScoresItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

// Show the student scores

showStudentScores();

}

});

ImageIcon img=new ImageIcon(getClass().getResource("p1.png"));

JLabel x=new JLabel(img);

frame.add(x);

frame.setJMenuBar(menuBar);

frame.setSize(600, 500); // Increased frame size

frame.setVisible(true);

}

private void showNotes() {

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

ResultSet rs=stmt.executeQuery("select \* from notes");

StringBuilder sb = new StringBuilder();

while(rs.next())

{

int x=rs.getInt(1);

String m=rs.getString(2);

sb.append(x).append(": ").append(m).append("\n");

}

JTextArea textArea = new JTextArea(sb.toString());

textArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(textArea);

scrollPane.setPreferredSize(new Dimension(400, 300));

JOptionPane.showMessageDialog(frame, scrollPane, "Notes", JOptionPane.PLAIN\_MESSAGE);

con.close();

System.out.println("Connection closed successfully.");

} catch (ClassNotFoundException e) {

System.err.println("Failed to load JDBC driver: " + e.getMessage());

} catch (SQLException e) {

System.err.println("Failed to connect to the database: " + e.getMessage());

}

}

private void showQuestions() {

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

ResultSet rs=stmt.executeQuery("select \* from questions");

StringBuilder sb = new StringBuilder();

while(rs.next())

{

int x=rs.getInt(1);

String m=rs.getString(2);

sb.append(x).append(": ").append(m).append("\n");

}

JTextArea textArea = new JTextArea(sb.toString());

textArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(textArea);

scrollPane.setPreferredSize(new Dimension(400, 300));

JOptionPane.showMessageDialog(frame, scrollPane, "Questions", JOptionPane.PLAIN\_MESSAGE);

con.close();

} catch (ClassNotFoundException e) {

System.err.println("Failed to load JDBC driver: " + e.getMessage());

} catch (SQLException e) {

System.err.println("Failed to connect to the database: " + e.getMessage());

}

}

private void createInsertQuestionPanel() {

insertQuestionPanel = new JPanel();

insertQuestionPanel.setLayout(new GridLayout(2, 2));

JLabel questionNumberLabel = new JLabel("Question Number:");

questionNumberTextField = new JTextField();

JLabel questionLabel = new JLabel("Question:");

questionTextField = new JTextField();

insertQuestionPanel.add(questionNumberLabel);

insertQuestionPanel.add(questionNumberTextField);

insertQuestionPanel.add(questionLabel);

insertQuestionPanel.add(questionTextField);

int option = JOptionPane.showConfirmDialog(frame, insertQuestionPanel, "Insert Question", JOptionPane.OK\_CANCEL\_OPTION);

if (option == JOptionPane.OK\_OPTION) {

// Retrieve the entered question number and question text

String questionNumber = questionNumberTextField.getText();

String questionText = questionTextField.getText();

if (questionNumber.isEmpty() || questionText.isEmpty() ) {

JOptionPane.showMessageDialog(null, "Please fill in all the fields", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

int a=stmt.executeUpdate("insert into questions values("+questionNumber+",'"+questionText+"')");

if(a>0)

JOptionPane.showMessageDialog(null, "Inserted successfully", "Error", JOptionPane.INFORMATION\_MESSAGE);

else

JOptionPane.showMessageDialog(null, "Enter valid Inputs", "Error", JOptionPane.ERROR\_MESSAGE);

// Close the connection

con.close();

} catch (ClassNotFoundException s) {

System.err.println("Failed to load JDBC driver: " + s.getMessage());

} catch (SQLException s) {

System.err.println("Failed to connect to the database: " + s.getMessage());

}

// Display a confirmation message

JOptionPane.showMessageDialog(frame, "Question Inserted:\nQuestion Number: " + questionNumber + "\nQuestion: " + questionText);

}

}

private void createUpdateQuestionPanel() {

insertQuestionPanel = new JPanel();

insertQuestionPanel.setLayout(new GridLayout(2, 2));

JLabel questionNumberLabel = new JLabel("Question Number:");

questionNumberTextField = new JTextField();

JLabel questionLabel = new JLabel("Question:");

questionTextField = new JTextField();

insertQuestionPanel.add(questionNumberLabel);

insertQuestionPanel.add(questionNumberTextField);

insertQuestionPanel.add(questionLabel);

insertQuestionPanel.add(questionTextField);

int option = JOptionPane.showConfirmDialog(frame, insertQuestionPanel, "Update Question", JOptionPane.OK\_CANCEL\_OPTION);

if (option == JOptionPane.OK\_OPTION) {

// Retrieve the entered question number and question text

String questionNumber = questionNumberTextField.getText();

String questionText = questionTextField.getText();

if (questionNumber.isEmpty() || questionText.isEmpty() ) {

JOptionPane.showMessageDialog(null, "Please fill in all the fields", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

int a = stmt.executeUpdate("UPDATE questions SET question='" + questionText + "'WHERE question\_no=" + questionNumber);

if(a>0)

JOptionPane.showMessageDialog(null, "Updated successfully", "Error", JOptionPane.INFORMATION\_MESSAGE);

else

JOptionPane.showMessageDialog(null, "Enter valid question number", "Error", JOptionPane.ERROR\_MESSAGE);

// Close the connection

con.close();

} catch (ClassNotFoundException s) {

System.err.println("Failed to load JDBC driver: " + s.getMessage());

} catch (SQLException s) {

System.err.println("Failed to connect to the database: " + s.getMessage());

}

// Display a confirmation message

JOptionPane.showMessageDialog(frame, "Question Updated:\nQuestion Number: " + questionNumber + "\nQuestion: " + questionText);

}

}

private void createDeleteQuestionPanel() {

insertQuestionPanel = new JPanel();

insertQuestionPanel.setLayout(new GridLayout(1, 2));

JLabel questionNumberLabel = new JLabel("Question Number:");

questionNumberTextField = new JTextField();

insertQuestionPanel.add(questionNumberLabel);

insertQuestionPanel.add(questionNumberTextField);

int option = JOptionPane.showConfirmDialog(frame, insertQuestionPanel, "Delete Question", JOptionPane.OK\_CANCEL\_OPTION);

if (option == JOptionPane.OK\_OPTION) {

// Retrieve the entered question number

String questionNumber = questionNumberTextField.getText();

if (questionNumber.isEmpty() ) {

JOptionPane.showMessageDialog(null, "Please fill in all the fields", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

int a=stmt.executeUpdate("delete from questions where question\_no="+questionNumber);

if(a>0)

JOptionPane.showMessageDialog(null, "Deleted successfully", "Error", JOptionPane.INFORMATION\_MESSAGE);

else

JOptionPane.showMessageDialog(null, "Enter valid question number", "Error", JOptionPane.ERROR\_MESSAGE);

// Close the connection

con.close();

} catch (ClassNotFoundException s) {

System.err.println("Failed to load JDBC driver: " + s.getMessage());

} catch (SQLException s) {

System.err.println("Failed to connect to the database: " + s.getMessage());

}

// Display a confirmation message

JOptionPane.showMessageDialog(frame, "Question Deleted\nQuestion Number: " + questionNumber);

}

}

private void createInsertNotesPanel() {

insertNotesPanel = new JPanel();

insertNotesPanel.setLayout(new GridLayout(2, 2));

JLabel chapterNumberLabel = new JLabel("Chapter Number:");

chapterNumberTextField = new JTextField();

JLabel chapterNameLabel = new JLabel("Chapter Name:");

chapterNameTextField = new JTextField();

insertNotesPanel.add(chapterNumberLabel);

insertNotesPanel.add(chapterNumberTextField);

insertNotesPanel.add(chapterNameLabel);

insertNotesPanel.add(chapterNameTextField);

int option = JOptionPane.showConfirmDialog(frame, insertNotesPanel, "Insert Notes", JOptionPane.OK\_CANCEL\_OPTION);

if (option == JOptionPane.OK\_OPTION) {

// Retrieve the entered chapter number and chapter name

String chapterNumber = chapterNumberTextField.getText();

String chapterName = chapterNameTextField.getText();

if (chapterNumber.isEmpty() || chapterName.isEmpty() ) {

JOptionPane.showMessageDialog(null, "Please fill in all the fields", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

int a = stmt.executeUpdate("INSERT INTO notes VALUES (" + chapterNumber + ", '" + chapterName + "')");

if(a>0)

JOptionPane.showMessageDialog(null, "Inserted successfully", "Error", JOptionPane.INFORMATION\_MESSAGE);

else

JOptionPane.showMessageDialog(null, "Enter valid chapter number and chapter name", "Error", JOptionPane.ERROR\_MESSAGE);

} catch (ClassNotFoundException e) {

System.err.println("Failed to load JDBC driver: " + e.getMessage());

} catch (SQLException e) {

System.err.println("Failed to connect to the database: " + e.getMessage());

}

JOptionPane.showMessageDialog(frame, "Notes Inserted:\nChapter Number: " + chapterNumber + "\nChapter Name: " + chapterName);

}

}

private void showStudentScores() {

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

ResultSet rs=stmt.executeQuery("select quiz\_no,marks,time from scores where student\_id="+id);

StringBuilder sb = new StringBuilder();

scoresTextArea = new JTextArea();

scoresTextArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(scoresTextArea);

scrollPane.setPreferredSize(new Dimension(400, 300));

while(rs.next())

{

int x=rs.getInt(1);

int y=rs.getInt(2);

sb.append("Quiz-").append(x).append(": ").append(y).append(" ").append(rs.getString(3)).append("\n");

}

scoresTextArea.setText(sb.toString());

JOptionPane.showMessageDialog(frame, scoresTextArea, "Student Scores", JOptionPane.PLAIN\_MESSAGE);

con.close();

//System.out.println("Connection closed successfully.");

} catch (ClassNotFoundException s) {

System.err.println("Failed to load JDBC driver: " + s.getMessage());

} catch (SQLException s) {

System.err.println("Failed to connect to the database: " + s.getMessage());

}

}

}

**2.Quiz Application1 Page:**

import javax.swing.\*;

import javax.swing.border.EmptyBorder;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.Statement;

import java.text.SimpleDateFormat;

import java.util.Date;

public class QuizApplication extends JFrame {

private JLabel[] questionLabels;

private JRadioButton[][] options;

private JButton submitButton;

private int totalQuestions;

private int score;

int id;

// Quiz questions and answers

private String[] questions = {

"Question 1: What is the full form of DDL?",

"Question 2: What is DBMS?",

"Question 3: Which type of data can be stored in the database?",

"Question 4: In which of the following formats is data stored in the database management system?",

"Question 5: Which of the following is known as a set of entities of the same type that share the same properties or attributes?"

};

private String[][] optionsList = {

{"Data Definition Language", "Data Derivation Language", "Dynamic Data Language", "Detailed Data Language"},

{"DBMS is a collection of queries", "DBMS is a high-level language", "DBMS is a programming language", "DBMS stores, modifies, retrieves data"},

{"Image-oriented data", "Text files containing data", "Data in the form of audio or video", "All of the above"},

{"Image", "Text", "Table", "Graph"},

{"Relation set", "Entity set", "Entity Relation model", "Tuples"}

};

private int[] answers = {0, 3, 3,2,1}; // Index of the correct answer for each question

public QuizApplication(int id) {

this.id=id;

setTitle("Quiz Application");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new BorderLayout());

setResizable(false);

JPanel contentPanel = new JPanel();

contentPanel.setLayout(new BoxLayout(contentPanel, BoxLayout.Y\_AXIS));

contentPanel.setBorder(new EmptyBorder(10, 10, 10, 10));

questionLabels = new JLabel[questions.length];

options = new JRadioButton[questions.length][4];

ButtonGroup[] groups = new ButtonGroup[questions.length];

setSize(900, 600);

setVisible(true);

for (int i = 0; i < questions.length; i++) {

JPanel questionPanel = new JPanel(new BorderLayout());

questionPanel.setBorder(new EmptyBorder(10, 0, 10, 0));

questionLabels[i] = new JLabel();

questionPanel.add(questionLabels[i], BorderLayout.NORTH);

JPanel optionsPanel = new JPanel(new GridLayout(0, 1));

groups[i] = new ButtonGroup();

for (int j = 0; j < 4; j++) {

options[i][j] = new JRadioButton();

optionsPanel.add(options[i][j]);

groups[i].add(options[i][j]);

}

questionPanel.add(optionsPanel, BorderLayout.CENTER);

contentPanel.add(questionPanel);

}

JScrollPane scrollPane = new JScrollPane(contentPanel);

scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL\_SCROLLBAR\_ALWAYS);

scrollPane.setPreferredSize(new Dimension(600, 600));

submitButton = new JButton("Submit");

submitButton.setAlignmentX(Component.CENTER\_ALIGNMENT);

submitButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

checkAnswers();

setVisible(false);

showResult();

}

});

contentPanel.add(Box.createRigidArea(new Dimension(0, 10)));

contentPanel.add(submitButton);

totalQuestions = questions.length;

score = 0;

showQuestions();

add(scrollPane, BorderLayout.CENTER);

pack();

setLocationRelativeTo(null); // Center the frame on the screen

}

private void showQuestions() {

for (int i = 0; i < totalQuestions; i++) {

questionLabels[i].setText(questions[i]);

for (int j = 0; j < 4; j++) {

options[i][j].setText(optionsList[i][j]);

options[i][j].setSelected(false);

}

}

}

private void checkAnswers() {

for (int i = 0; i < totalQuestions; i++) {

for (int j = 0; j < 4; j++) {

if (options[i][j].isSelected() && j == answers[i]) {

score++;

break;

}

}

}

}

private void showResult() {

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");

Date date = new Date();

int a=stmt.executeUpdate("insert into scores values("+id+","+score+",1,'"+formatter.format(date)+"')");

if(a>0)

JOptionPane.showMessageDialog(null, "Attempted Quiz Successfully", "Success", JOptionPane.INFORMATION\_MESSAGE);

else

JOptionPane.showMessageDialog(null, "Fail", "Error", JOptionPane.ERROR\_MESSAGE);

// Close the connection

con.close();

//System.out.println("Connection closed successfully.");

} catch (ClassNotFoundException s) {

System.err.println("Failed to load JDBC driver: " + s.getMessage());

} catch (SQLException s) {

System.err.println("Failed to connect to the database: " + s.getMessage());

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

@Override

public void run() {

QuizApplication quizApplication = new QuizApplication(1);

quizApplication.setSize(800, 600); // Increase the size of the frame

quizApplication.setVisible(true);

}

});

}

}

**2.Quiz Application2 Page:**

import javax.swing.\*;

import javax.swing.border.EmptyBorder;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.Statement;

import java.text.SimpleDateFormat;

import java.util.Date;

public class QuizApplication1 extends JFrame {

private JLabel[] questionLabels;

private JRadioButton[][] options;

private JButton submitButton;

private int totalQuestions;

private int score;

int id;

// Quiz questions and answers

private String[] questions = {

"Question 1: What does an RDBMS consist of?",

"Question 2: The ability to query data, as well as insert, delete, and alter tuples, is offered by \_\_\_\_\_\_\_\_\_\_\_\_",

"Question 3: Which of the following command is correct to delete the values in the relation teaches?",

"Question 4: Which one of the following given statements possibly contains the error?",

"Question 5: What do you mean by one to many relationships?"

};

private String[][] optionsList = {

{"Collection of Records", "Collection of Keys", "Collection of Tables", "Collection of Fields"},

{"TCL (Transaction Control Language)", "DCL (Data Control Language)", "DDL (Data Definition Langauge)", "DML (Data Manipulation Langauge)"},

{"Delete from teaches;", "Delete from teaches where Id =’Null’;", "Remove table teaches;", "Drop table teaches;"},

{"select \* from emp where empid = 10003;", "select empid from emp where empid = 10006;", "select empid from emp;", "select empid where empid = 1009 and Lastname = 'GELLER';"},

{"One class may have many teachers", "One teacher can have many classes", "Many classes may have many teachers", "Many teachers may have many classes"}

};

private int[] answers = {2,3,0,3,1}; // Index of the correct answer for each question

public QuizApplication1(int id) {

this.id=id;

setTitle("Quiz Application");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new BorderLayout());

setResizable(false);

JPanel contentPanel = new JPanel();

contentPanel.setLayout(new BoxLayout(contentPanel, BoxLayout.Y\_AXIS));

contentPanel.setBorder(new EmptyBorder(10, 10, 10, 10));

questionLabels = new JLabel[questions.length];

options = new JRadioButton[questions.length][4];

ButtonGroup[] groups = new ButtonGroup[questions.length];

setSize(900, 600);

setVisible(true);

for (int i = 0; i < questions.length; i++) {

JPanel questionPanel = new JPanel(new BorderLayout());

questionPanel.setBorder(new EmptyBorder(10, 0, 10, 0));

questionLabels[i] = new JLabel();

questionPanel.add(questionLabels[i], BorderLayout.NORTH);

JPanel optionsPanel = new JPanel(new GridLayout(0, 1));

groups[i] = new ButtonGroup();

for (int j = 0; j < 4; j++) {

options[i][j] = new JRadioButton();

optionsPanel.add(options[i][j]);

groups[i].add(options[i][j]);

}

questionPanel.add(optionsPanel, BorderLayout.CENTER);

contentPanel.add(questionPanel);

}

JScrollPane scrollPane = new JScrollPane(contentPanel);

scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL\_SCROLLBAR\_ALWAYS);

scrollPane.setPreferredSize(new Dimension(600, 600));

submitButton = new JButton("Submit");

submitButton.setAlignmentX(Component.CENTER\_ALIGNMENT);

submitButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

checkAnswers();

setVisible(false);

showResult();

}

});

contentPanel.add(Box.createRigidArea(new Dimension(0, 10)));

contentPanel.add(submitButton);

totalQuestions = questions.length;

score = 0;

showQuestions();

add(scrollPane, BorderLayout.CENTER);

pack();

setLocationRelativeTo(null); // Center the frame on the screen

}

private void showQuestions() {

for (int i = 0; i < totalQuestions; i++) {

questionLabels[i].setText(questions[i]);

for (int j = 0; j < 4; j++) {

options[i][j].setText(optionsList[i][j]);

options[i][j].setSelected(false);

}

}

}

private void checkAnswers() {

for (int i = 0; i < totalQuestions; i++) {

for (int j = 0; j < 4; j++) {

if (options[i][j].isSelected() && j == answers[i]) {

score++;

break;

}

}

}

}

private void showResult() {

String driverClassName = "oracle.jdbc.driver.OracleDriver";

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String username = "bagawan";

String pass = "laddu";

try {

// Load the JDBC driver

Class.forName(driverClassName);

// Establish a connection to the database

Connection con = DriverManager.getConnection(url, username, pass);

// Perform database operations using the connection

Statement stmt=con.createStatement();

SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");

Date date = new Date();

int a=stmt.executeUpdate("insert into scores values("+id+","+score+",2,'"+formatter.format(date)+"')");

if(a>0)

JOptionPane.showMessageDialog(null, "Attempted Quiz Successfully", "Success", JOptionPane.INFORMATION\_MESSAGE);

else

JOptionPane.showMessageDialog(null, "Fail", "Error", JOptionPane.ERROR\_MESSAGE);

// Close the connection

con.close();

//System.out.println("Connection closed successfully.");

} catch (ClassNotFoundException s) {

System.err.println("Failed to load JDBC driver: " + s.getMessage());

} catch (SQLException s) {

System.err.println("Failed to connect to the database: " + s.getMessage());

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

@Override

public void run() {

QuizApplication1 quizApplication = new QuizApplication1(1);

quizApplication.setSize(800, 600); // Increase the size of the frame

quizApplication.setVisible(true);

}

});

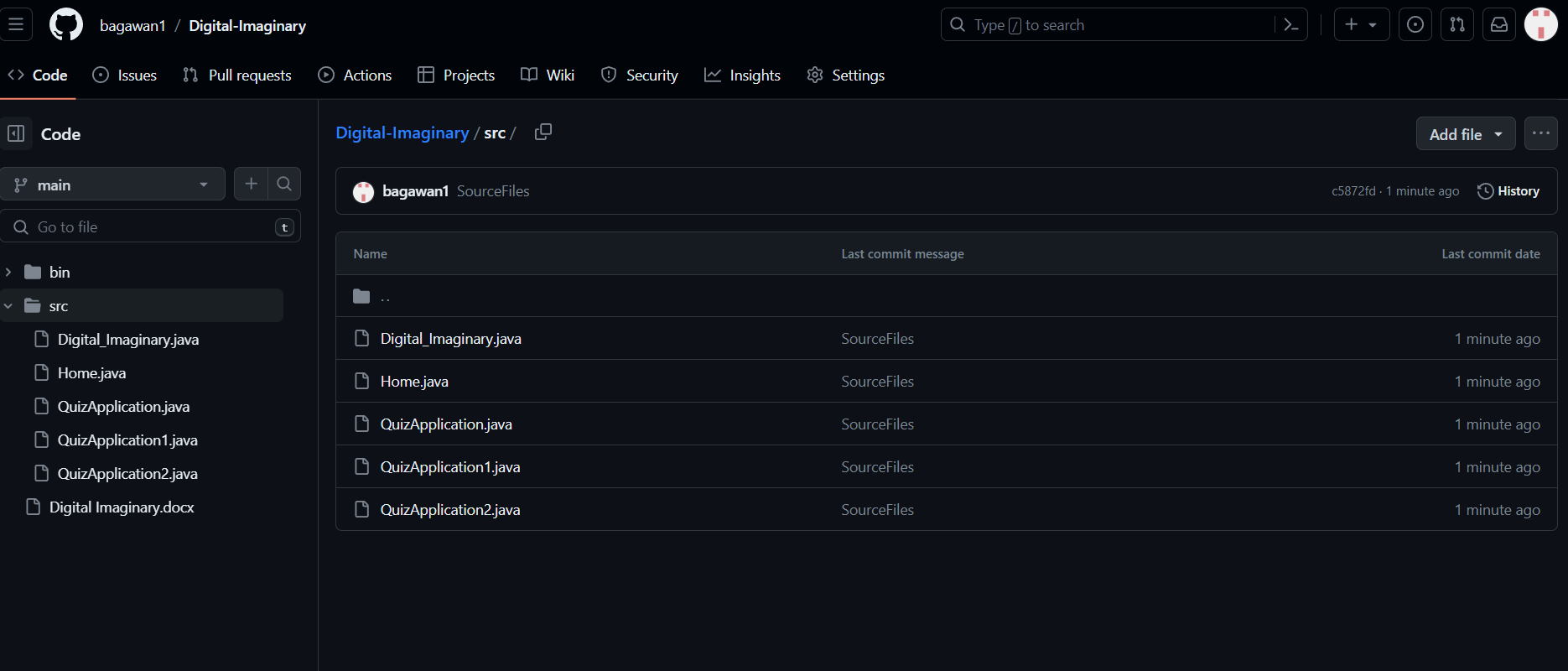
}

}

**GitHub Links and Folder Structure**

**LINK:** [**https://github.com/bagawan1/Digital-Imaginary/**](https://github.com/bagawan1/Digital-Imaginary/)

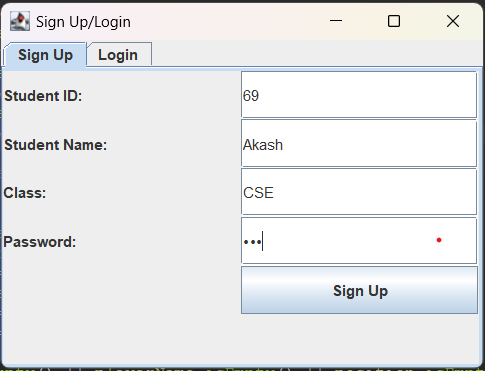
**FOLDER STRUCTURE:**

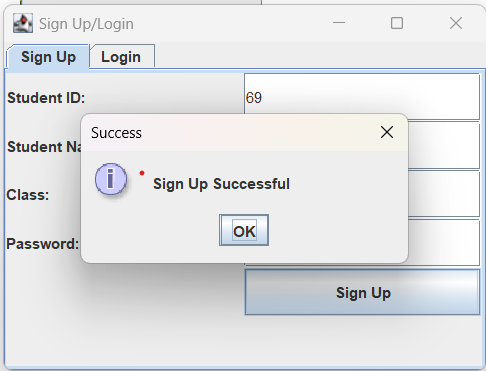
****

# TESTING

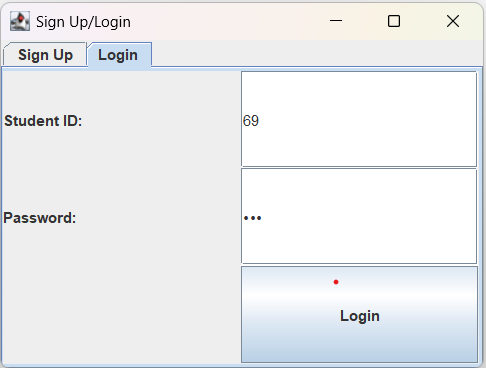
**DIGITAL IMAGINARY PAGE:**

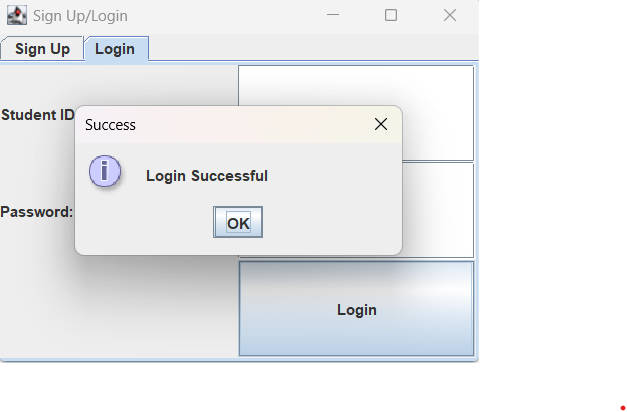
**Signup:**



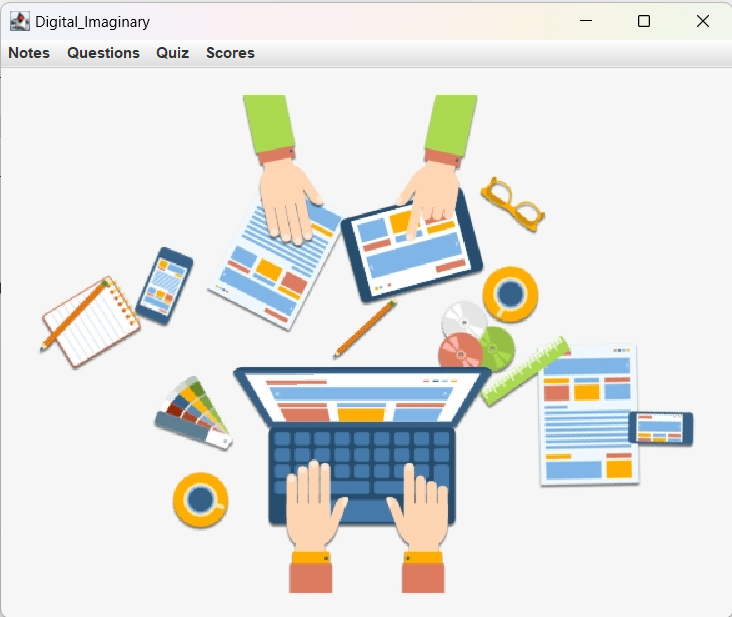


**Login:**

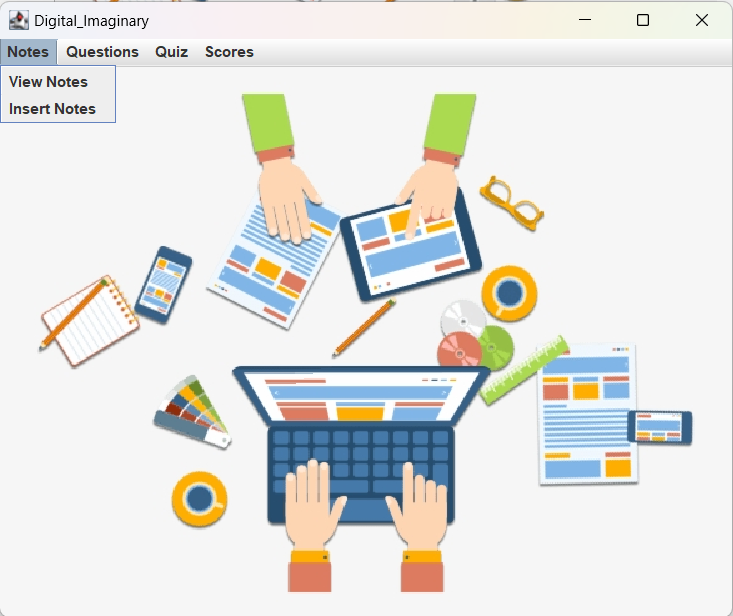
****

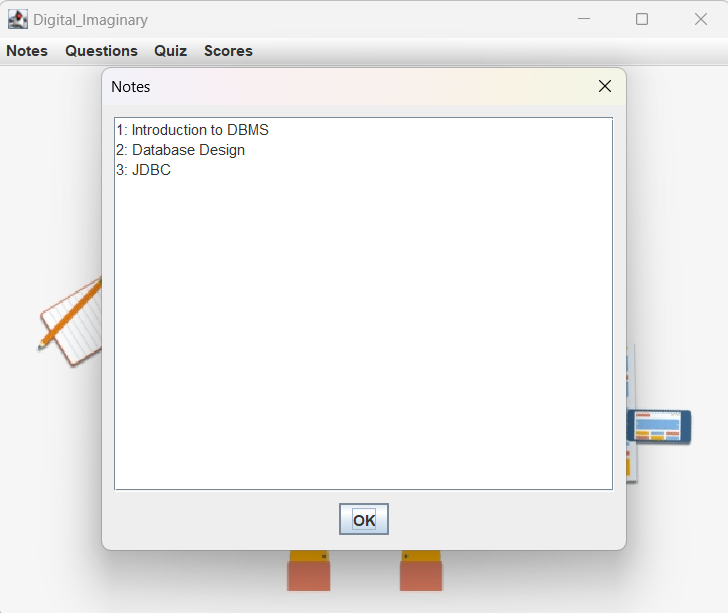
****

**Home:**

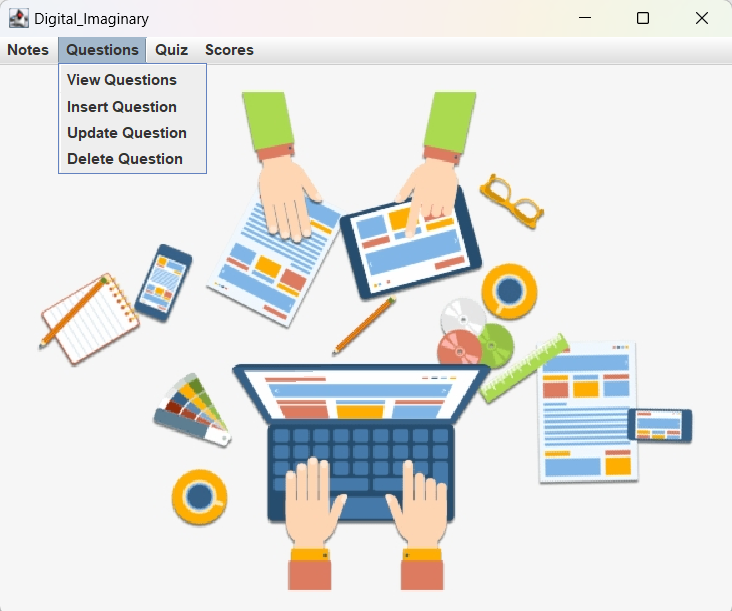


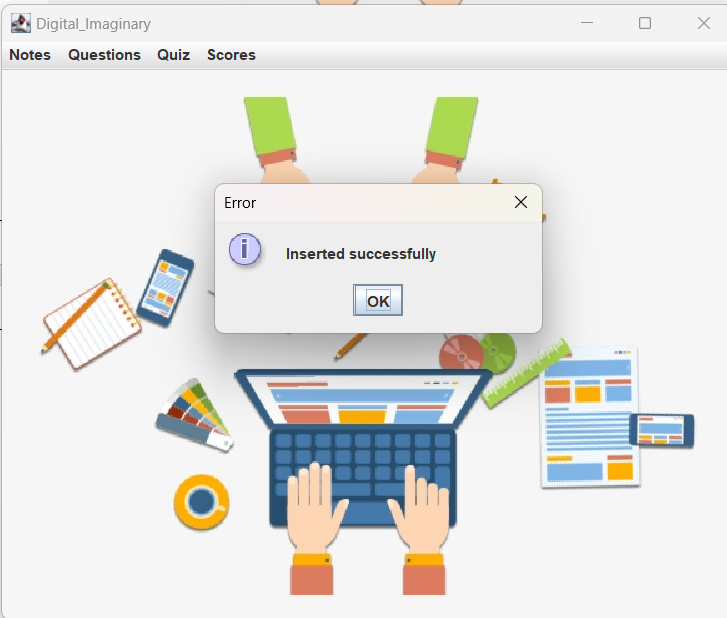
**Notes:**

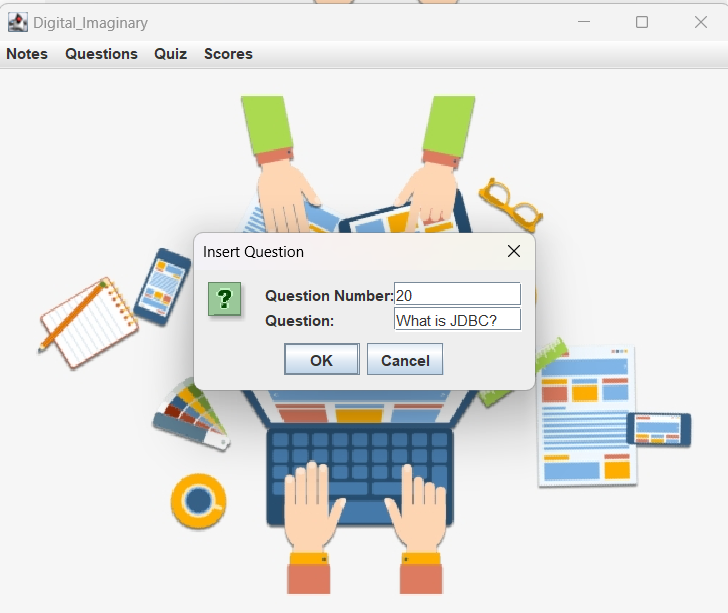


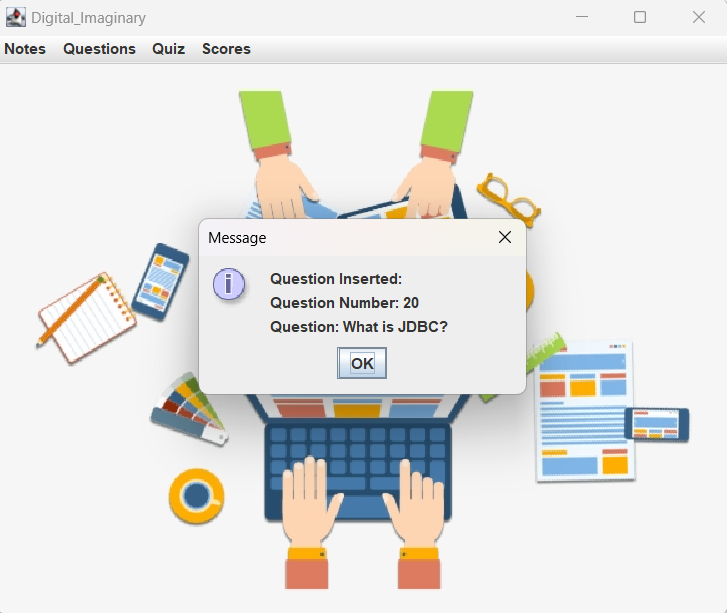
****

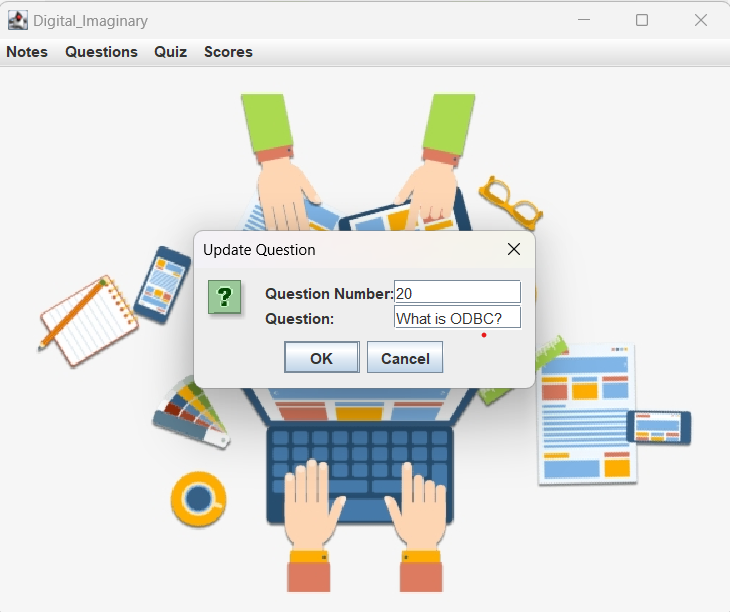
**Questions:**

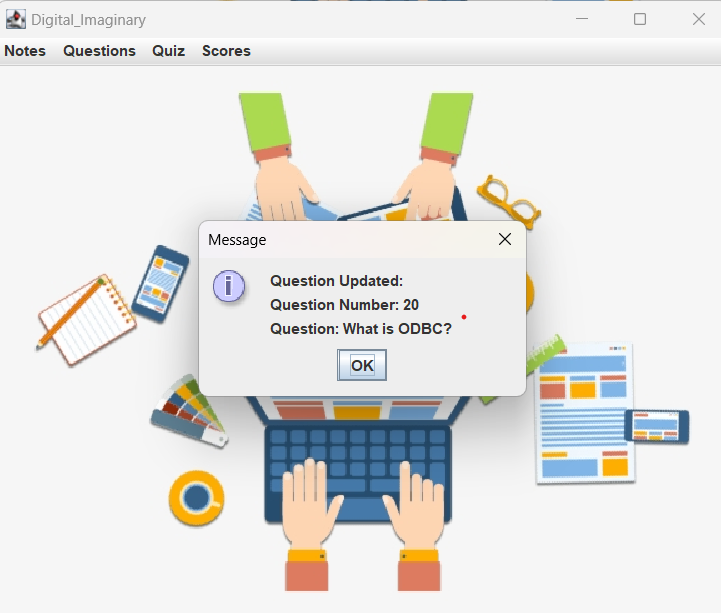
****

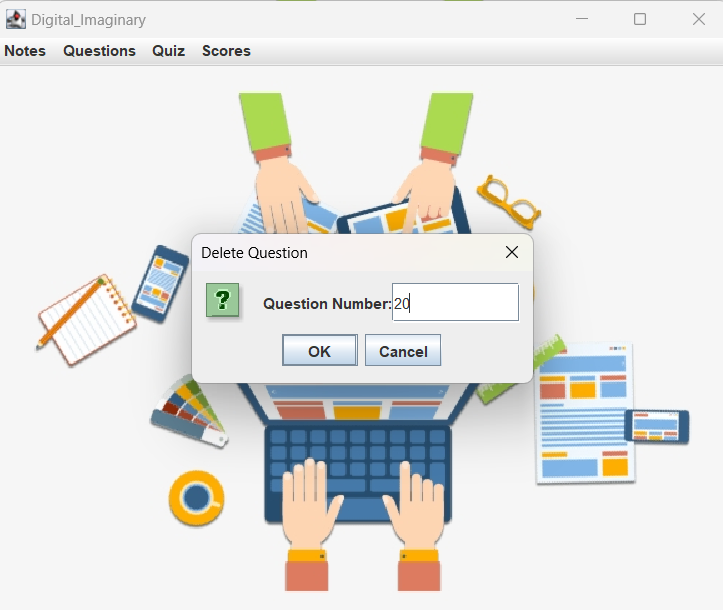
****

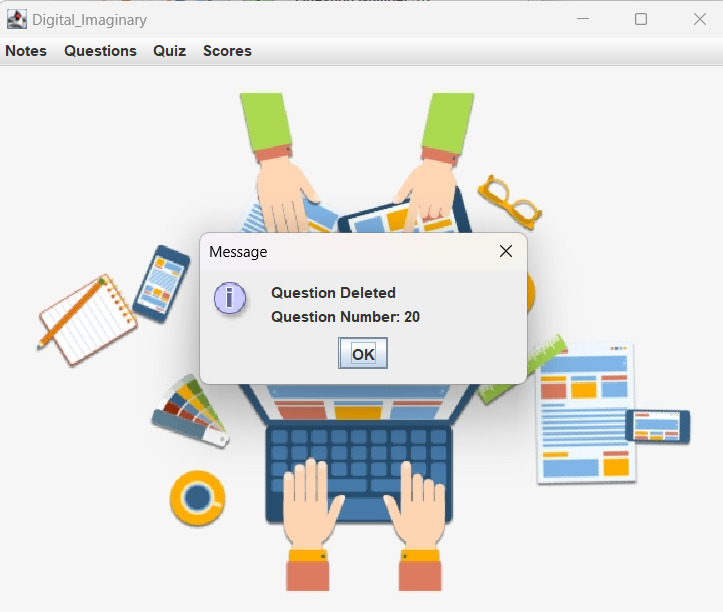
****



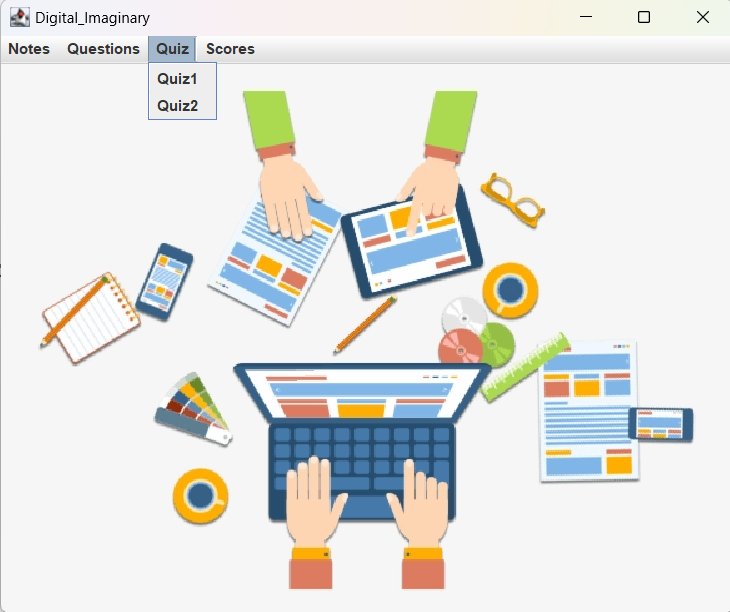


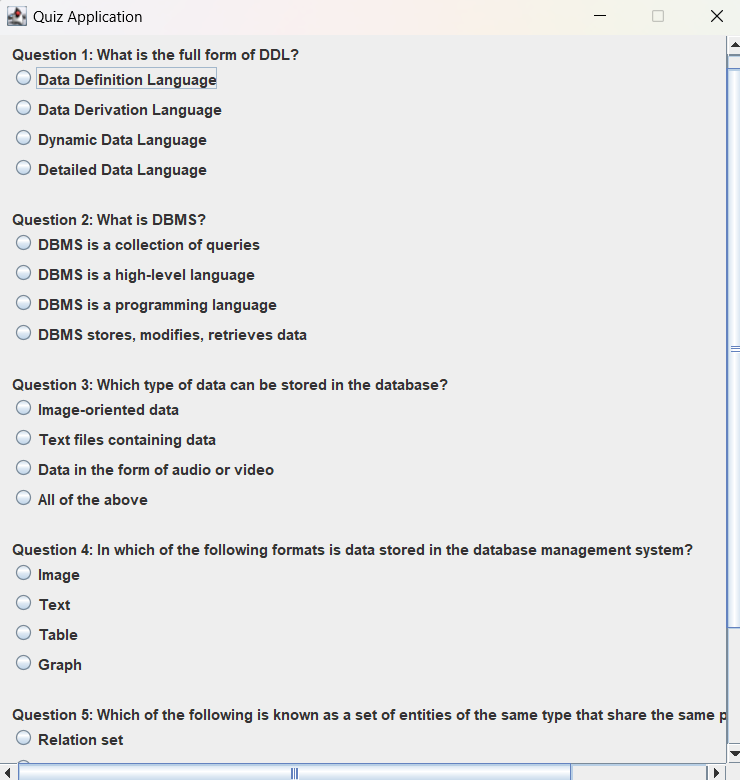
****

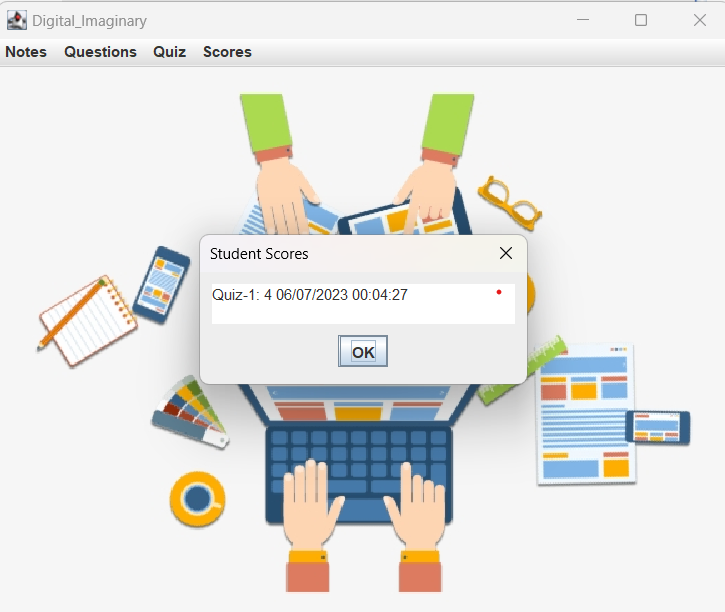
****

****

**Quiz:**







**RESULTS**

I have completed ***‘DIGITAL IMAGINARY’*** project successfully.

## DISCUSSION AND FUTURE WORK

This project contains the basic interaction of giving information by students for suggesting the correct career choice. It has a very basic user interface.

Future scope would be to make the UI more appealing by using graphics. more feature would be to allow student-users to upload their resumes and official One documents required so that we can suggest more accurate career choices. We can also think of including a feedback system to allow the users to leave their valuable feedback after using this app. Making this feedback to be publicly viewable, would attract many more users to use this app.

## REFERENCES

* https://docs.oracle.com/javase/7/docs/api/
* [https://www.javatpoint.com/java-swing](http://www.javatpoint.com/java-swing)
* https://stackoverflow.com/