ONLINE EXAMINATION IN JAVA

JAVA MINI PROJECT

SUBMITTED BY: PADALA RAMU

Registration no: 12106704

Roll No: RK21RB**A10**

GitHub-ID: **ramu2245**

METIKALA DHEERAJ

Registration no:12106961

Roll No: RK21RB**A08**

GitHub-ID: **metikaladheeraj**

BUDATI NAVAGOPAL

Registration no: 12114580

RK21RB**A24**

GitHub-ID: **Navagopalreddy**

In partial fulfilment for the requirements of the award of the degree of

"B. Tech (Computer Science Engineering)

PHAGWARA, PUNJAB

SUBMITTED TO: Dr. Bhimasen Moharana sir

INTRODUCTION

An online exam project is an application designed to facilitate the administration of online exams. With the increasing popularity of online learning, conducting exams online has become a necessity. This Java project provides an automated system to conduct exams online, making it easier for educational institutions to assess their students' knowledge and skills.

The project allows students to log in and take the exam online. It also enables teachers or administrators to create and manage exams, set time limits, and assign marks. The application has features such as multiple-choice questions, true or false questions, and descriptive answers.

This project aims to provide an easy-to-use, efficient, and secure platform for online exams. It ensures the privacy and integrity of the exam data by encrypting it and storing it in a database. It also generates reports on student performance, which can help teachers and administrators evaluate and analyse their students' strengths and weaknesses.

The main objectives of this online exam project in Java are to reduce the time and resources required to conduct exams, minimize the chances of cheating or fraud, and provide an efficient and reliable assessment system

He online exam project in Java also offers flexibility to both students and teachers. Students can take the exam from anywhere and at any time, as long as they have access to the internet. Teachers can also create and manage exams from anywhere, eliminating the need for physical exam papers, which can be time-consuming and expensive.

Modules:

- User Management Module: This module would be responsible for managing user accounts.
- Exam Management Module: This module would be responsible for creating, updating, and managing exams. It would include functionality for creating different types of questions.
- Results Module: This module would be responsible for recording and displaying the results of exams taken by users.

Objectives:

The objectives for an online exam project are:

- Automate the process of conducting exams: The project should provide an automated platform for students to take exams online. It should include features such as creating and assigning marks.
- Ensure exam security and integrity: The project should ensure that the exam data is secure and cannot be tampered with. It should implement measures such as encrypting the data and storing it in a database.
- Provide a user-friendly interface: The project should have a user-friendly interface for students. It should be easy to navigate and use, with clear instructions and prompts.
- Ensure compatibility with multiple devices: The project should be compatible with different devices such as desktops, laptops, tablets, and smartphones. It should also work on different operating systems such as Windows, Linux, and macOS.
- Provide flexibility: The project should provide flexibility to both students and teachers. Students should be able to take exams from anywhere and at any time, while teachers should be able to create and manage exams from anywhere.
- Improve efficiency and reduce costs: The project should reduce the time and resources required to conduct exams. It should eliminate the need for physical exam papers and reduce the cost of printing and distributing them.

Observations:

- The project combines GUI programming with event-driven programming to create a simple online examination system.
- The project uses Swing and awt components to create the GUI interface for the application.
- The project includes a LoginGUI class that allows users to log in or sign up for an account.
- The LoginGUI class includes event handlers for the sign-in and sign-up buttons that check the user's credentials and create a new instance of the OnlineTest class.
- The OnlineTest class implements the ActionListener interface to handle user events.
- The OnlineTest class includes a set of radio buttons for each question, a Next button to move to the next question, a Bookmark button to save the current question, and a Result button to display the user's final score.
- The OnlineTest class uses a set of arrays to store the questions, options, and correct answers.
- The project is organized into separate classes for the GUI and the application logic, which makes it easier to maintain and modify.
- However, the project could be improved by adding more error handling and input validation to prevent users from entering invalid input or crashing the program.
- The project could also be extended by adding more features such as a timer, scoring system, or support for multiple users.

Software requirements:

- Java Development Kit (JDK) - This is necessary for developing the Java code of the project.
- Integrated Development Environment (IDE) - An IDE such as Eclipse, NetBeans, or IntelliJ IDEA can be used for writing, debugging, and testing the code.
- JavaScript - These technologies are used to create the user interface of the online exam system.

Hardware Requirements:

- Server: A high-performance server with multiple cores and high-speed RAM is required to handle the heavy load of concurrent users.
- Storage: Sufficient storage space is required to store the exam questions, answers, user data, and other relevant information.
- Clients: End-users will require desktop computers, laptops, tablets, or smartphones with a reliable internet connection to access the online exam system

Procedure:

This program is a simple multiple-choice online test application in Java. It uses Swing and Awt GUI components to create the user interface, and allows the user to answer a series of questions and navigate between them using the "Next" and "Bookmark" buttons.

It creates a JFrame window, which contains a JLabel to display the current question, a ButtonGroup to group the answer options, and an array of JRadioButtons to display the answer options. It also contains two buttons: "Next" to move to the next question, and "Bookmark" to bookmark the current question and move to the next question.

When the user clicks the "Next" button, the program checks whether the user has selected the correct answer for the current question, updates the score, and moves to the next question. When the user clicks the "Bookmark" button, the program creates a new JButton labeled with the bookmark number, adds it to the window, and updates the array of bookmarks. When the user clicks a bookmark button, the program moves to the bookmarked question and disables the bookmark button.

The program contains a set() method that updates the question and answer options based on the current question number, and a check() method that checks whether the user has selected the correct answer for the current question.

Overall, the program provides a simple example of how to create a multiple-choice online test application in Java using Swing. However, there are several areas for improvement, such as adding a timer, shuffling the questions and answer options, and storing the questions and answers in an external file or database.

PROGRAMS

Login page

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.io.File;
import java.io.IOException;

public class LoginGUI extends JFrame {
    private JLabel userLabel, passLabel, confirmLabel;
    private JTextField userField;
    private JPasswordField passField, confirmField;
    private JButton signUpButton, signInButton;
    private LoginData savedData;
   private boolean loggedIn;

    public LoginGUI() {
        super("Login Page");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400, 200);
        setLocationRelativeTo(null);

        userLabel = new JLabel("Username: ");
        userField = new JTextField(20);
        passLabel = new JLabel("Password: ");
        passField = new JPasswordField(20);
        confirmLabel = new JLabel("Confirm Password: ");
        confirmField = new JPasswordField(20);
        signUpButton = new JButton("Sign Up");
        signInButton = new JButton("Sign In");

        JPanel panel = new JPanel(new GridLayout(4, 2));
        panel.add(userLabel);
        panel.add(userField);
        panel.add(passLabel);
        panel.add(passField);
        panel.add(confirmLabel);
        panel.add(confirmField);
        panel.add(signUpButton);
        panel.add(signInButton);

        getContentPane().add(panel);

        signUpButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String username = userField.getText();
                String password = new String(passField.getPassword());
```

```
                String confirm = new String(confirmField.getPassword());
                if (password.equals(confirm)) {
                    // Save the data to a new class
                    savedData = new LoginData(username, password);
                    // Clear the fields
                    userField.setText("");
                    passField.setText("");
                    confirmField.setText("");
                    JOptionPane.showMessageDialog(LoginGUI.this, "Sign up
successful!");
                } else {
                    JOptionPane.showMessageDialog(LoginGUI.this, "Passwords do
not match!", "Error", JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        signInButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String username = userField.getText();
                String password = new String(passField.getPassword());
                if (savedData != null &&
savedData.getUsername().equals(username) &&
savedData.getPassword().equals(password)) {
                    // Clear the fields
                    userField.setText("");
                    passField.setText("");
                    confirmField.setText("");
                    JOptionPane.showMessageDialog(LoginGUI.this, "Login
successful!");
                    // Set the loggedIn flag to true
                    loggedIn = true;
                    // Create a new instance of the OnlineTest class and
display it
                    OnlineTest onlineTest = new OnlineTest("Online Test");
                    onlineTest.setVisible(true);
                    onlineTest.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                    onlineTest.setSize(800, 600);
                    onlineTest.setLocationRelativeTo(null);
                } else {
                    JOptionPane.showMessageDialog(LoginGUI.this, "Incorrect
username or password", "Error", JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        setVisible(true);
    }
```

```
    public boolean isLoggedIn() {
        return loggedIn;
    }

    public static void main(String[] args) {
        new LoginGUI();
    }
}

class LoginData {
    private String username;
    private String password;

    public LoginData(String username, String password) {
        this.username = username;
        this.password = password;
        // Save the data to a file or database
    }

    public String getUsername() {
        return username;
    }

    public String getPassword() {
        return password;
    }
}
```

Online test page

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class OnlineTest extends JFrame implements ActionListener
{
    JLabel l;
    JRadioButton jb[]=new JRadioButton[5];
    JButton b1,b2;
    ButtonGroup bg;
    int count=0,current=0,x=1,y=1,now=0;
    int m[]=new int[10];
    OnlineTest(String s)
    {
        super(s);
        l=new JLabel();
```

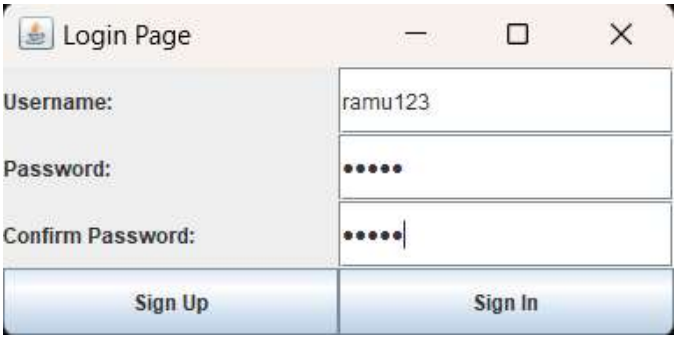
```
        add(l);
        bg=new ButtonGroup();
        for(int i=0;i<5;i++)
        {
            jb[i]=new JRadioButton();
            add(jb[i]);
            bg.add(jb[i]);
        }
        b1=new JButton("Next");
        b2=new JButton("Bookmark");
        b1.addActionListener(this);
        b2.addActionListener(this);
        add(b1);add(b2);
        set();
        l.setBounds(30,40,450,20);
        jb[0].setBounds(50,80,100,20);
        jb[1].setBounds(50,110,100,20);
        jb[2].setBounds(50,140,100,20);
        jb[3].setBounds(50,170,100,20);
        b1.setBounds(100,240,100,30);
        b2.setBounds(270,240,100,30);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(null);
        setLocation(250,100);
        setVisible(true);
        setSize(600,350);
    }
    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==b1)
        {
            if(check())
                count=count+1;
            current++;
            set();
            if(current==9)
            {
                b1.setEnabled(false);
                b2.setText("Result");
            }
        }
        if(e.getActionCommand().equals("Bookmark"))
        {
            JButton bk=new JButton("Bookmark"+x);
            bk.setBounds(480,20+30*x,100,30);
            add(bk);
            bk.addActionListener(this);
            m[x]=current;
```

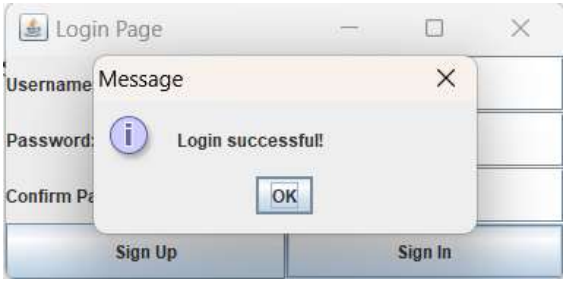
```
            x++;
            current++;
            set();
            if(current==9)
                b2.setText("Result");
            setVisible(false);
            setVisible(true);
        }
        for(int i=0,y=1;i<x;i++,y++)
        {
        if(e.getActionCommand().equals("Bookmark"+y))
        {
            if(check())
                count=count+1;
            now=current;
            current=m[y];
            set();
            ((JButton)e.getSource()).setEnabled(false);
            current=now;
        }
        }

        if(e.getActionCommand().equals("Result"))
        {
            if(check())
                count=count+1;
            current++;
            //System.out.println("correct ans="+count);
            JOptionPane.showMessageDialog(this,"correct ans="+count);
            System.exit(0);
        }
    }
    void set()
    {
        jb[4].setSelected(true);
        if(current==0)
        {
            l.setText("Que1: Which one among these is not a datatype");
            jb[0].setText("int");jb[1].setText("Float");jb[2].setText("boolean
");jb[3].setText("char");
        }
        if(current==1)
        {
            l.setText("Que2: Which class is available to all the class
automatically");
            jb[0].setText("Swing");jb[1].setText("Applet");jb[2].setText("Obje
ct");jb[3].setText("ActionEvent");
        }
```

```
        if(current==2)
        {
            l.setText("Que3: Which package is directly available to our class
without importing it");
            jb[0].setText("swing");jb[1].setText("applet");jb[2].setText("net"
);jb[3].setText("lang");
        }
        if(current==3)
        {
            l.setText("Que4: String class is defined in which package");
            jb[0].setText("lang");jb[1].setText("Swing");jb[2].setText("Applet
");jb[3].setText("awt");
        }
        if(current==4)
        {
            l.setText("Que5: Which institute is best for java coaching");
            jb[0].setText("Utek");jb[1].setText("Aptech");jb[2].setText("SSS
IT");jb[3].setText("jtek");
        }
        if(current==5)
        {
            l.setText("Que6: Which one among these is not a keyword");
            jb[0].setText("class");jb[1].setText("int");jb[2].setText("get");j
b[3].setText("if");
        }
        if(current==6)
        {
            l.setText("Que7: Which one among these is not a class ");
            jb[0].setText("Swing");jb[1].setText("Actionperformed");jb[2].setT
ext("ActionEvent");jb[3].setText("Button");
        }
        if(current==7)
        {
            l.setText("Que8: which one among these is not a function of Object
class");
            jb[0].setText("toString");jb[1].setText("finalize");jb[2].setText(
"equals");jb[3].setText("getDocumentBase");
        }
        if(current==8)
        {
            l.setText("Que9: which function is not present in Applet class");
            jb[0].setText("init");jb[1].setText("main");jb[2].setText("start")
;jb[3].setText("destroy");
        }
        if(current==9)
        {
            l.setText("Que10: Which one among these is not a valid
component");
```

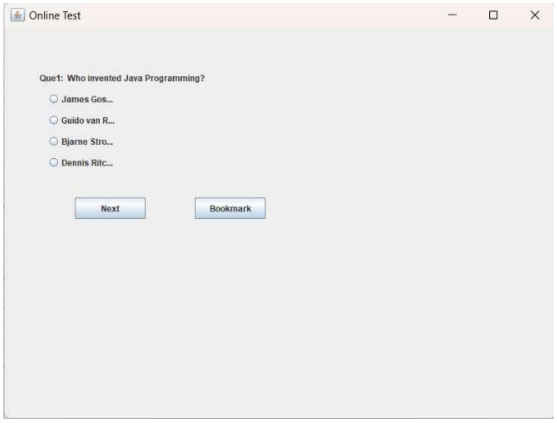
```
            jb[0].setText("JButton");jb[1].setText("JList");jb[2].setText("JBu
ttonGroup");jb[3].setText("JTextArea");
        }
        l.setBounds(30,40,450,20);
        for(int i=0,j=0;i<=90;i+=30,j++)
            jb[j].setBounds(50,80+i,200,20);
    }
    boolean check()
    {
        if(current==0)
            return(jb[1].isSelected());
        if(current==1)
            return(jb[2].isSelected());
        if(current==2)
            return(jb[3].isSelected());
        if(current==3)
            return(jb[0].isSelected());
        if(current==4)
            return(jb[2].isSelected());
        if(current==5)
            return(jb[2].isSelected());
        if(current==6)
            return(jb[1].isSelected());
        if(current==7)
            return(jb[3].isSelected());
        if(current==8)
            return(jb[1].isSelected());
        if(current==9)
            return(jb[2].isSelected());
        return false;
    }
    public static void main(String s[])
    {
        new OnlineTest("Online Test Of Java");
    }


}
```

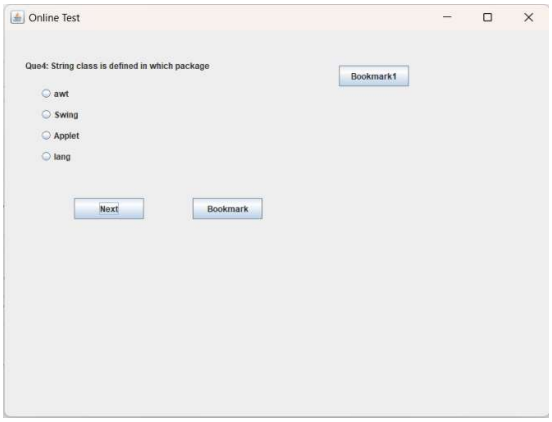
Screenshots of programs:



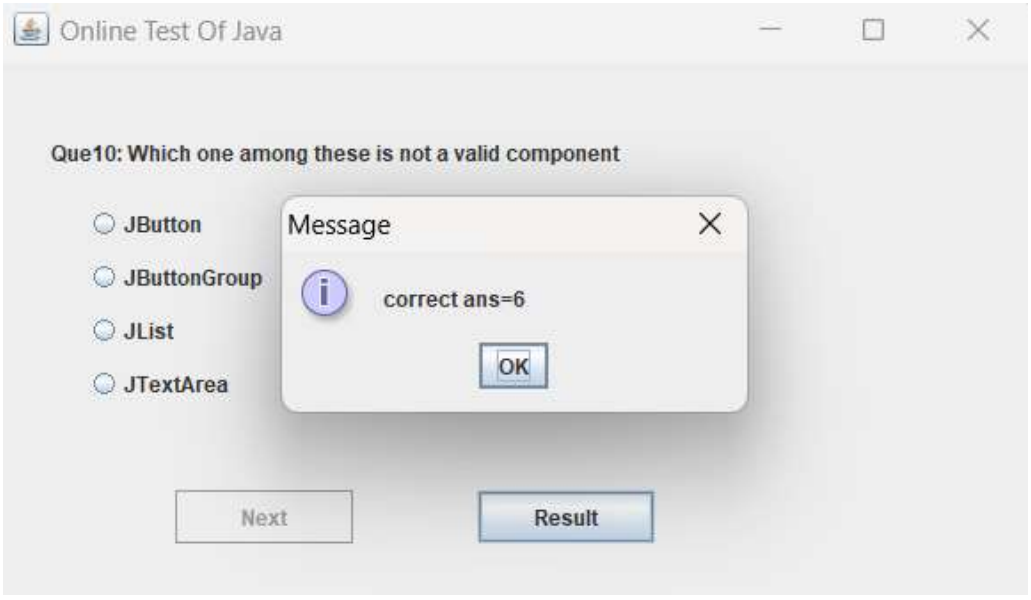


Here we can how an user is going to sign up and login to the exam.





Here we can seee how an user is going to attempt the test



Here we can see the result of an user.

Conclusion:

An online exam project in Java is a feasible and useful solution for conducting exams remotely. It allows for easy test creation, administration, and grading. The project's software requirements include a Java development environment, a database management system, and various libraries and frameworks. The hardware requirements include a computer with sufficient memory and processing power to handle the load of the application, and a stable internet connection. Overall, the project has the potential to improve the efficiency and effectiveness of exam administration, making it a valuable tool for educational institutions and businesses.

One potential challenge of developing an online exam project in Java is ensuring the security and integrity of the exam content and results. To address this issue, the project may need to incorporate features such as encryption and secure authentication methods to prevent unauthorized access or tampering of exam materials. Additionally, the project may need to consider strategies for detecting and preventing cheating, such as randomizing question orders or using remote proctoring software.

Overall, an online exam project in Java has the potential to provide a flexible and scalable solution for delivering exams to a wide audience. By carefully considering the software and hardware requirements, as well as the project's design and implementation, developers can create a robust and reliable platform for administering exams online. With the ability to support a variety of question types, provide real-time feedback, and automate grading, an online exam project in Java can streamline the exam process and make it more accessible and convenient for both students and instructors.

THANK YOU