KARPAGAM
COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
Accredited by NAAC with 'A' grade
Autonomous | Affiliated to Anna University
(An ISO 9001:2015 and ISO 14001:2015 Certified Institution)

AUTOMATIC QUIZ GENERATION

Presented By

21L355 - SUDHEESH V.G

Guided By

Dr. S. KANNIMUTHU

PROFESSOR / CSE

KARPAGAM
COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
Accredited by NAAC with 'A' grade
Autonomous | Affiliated to Anna University
(An ISO 9001:2015 and ISO 14001:2015 Certified Institution)

AGENDA

- INTRODUCTION
- PROBLEM STATEMENT
- ISSUES IN EXISTING SYSTEM
- PROPOSED SYSTEM
- TOOLS / TECHNOLOGIES / LANGUAGES USED
- ALGORITHM DESIGN
- MATHEMATICAL ANALYSIS
- IMPLEMENTATION
- SCREENSHOTS
- DEMONSTRATION
- CONCLUSION
- REFERENCES

INTRODUCTION

I am happy to introduce our cutting-edge automatic quiz generation system.To improve performance and grab lots of information in a less amount of time is pretty tough, we need to reach lots of information on sites and yet we forgot them in seconds. Whether you're a student looking to test your understanding or an educator seeking efficient assessment tools,our platform is tailored for you. They can give as many as tests as they want. Let the quiz generation revolution begin!"

PROBLEM STATEMENT

Developing an automated quiz generator that produces quizzes with diverse difficulty levels is valuable for education, training and assessment purposes. Your task is to design and develop an automated quiz generator that utilizes algorithms for question selection and arrangement to create quizzes with varying levels if difficulty.

ISSUES IN EXISTING SYSTEM

- Data Quality and Availability
- Model Complexity
- Heterogeneity
- Node Mobility
- Temporal Dynamics
- Model Validation
- Sensitivity to Parameters
- Computational Intensity
- Emerging Variants

PROPOSED SYSTEM

- Realistic Contact Network Construction
- Dynamic Network Models
- Data Integration and Calibration
- Agent-Based Modeling
- Sensitivity Analysis
- High-Performance Computing
- Scenario Planning and Risk Assessment
- Education and Training
- Adaptability and Real-Time Updates

TOOLS / TECHNOLOGIES / LANGUAGES USED

- Java is suitable for building scalable, high-performance simulations. It is often used for quiz automations and simulations where computational efficiency is crucial.

- The language used in the proposed system for building the Document Search Engine is Java. It's a versatile and powerful programming language that can be used for efficient data processing and indexing, making it well-suited for the task of creating a quiz automation. Additionally, Java provides good control over memory management, which is essential for optimizing performance.

ALGORITHM DESIGN

The goal is to create a system that can generate diverse and meaningfulquestions based on a given set of topics or content.

Question Types:

- Determine the types of questions you want in your quiz (e.g., multiple choice).
- Define the criteria for generating each type of question.

Quality Control:

- Implement checks to ensure the generated questions are accurate and meaningful.
- Evaluate the relevance of each question to the learning objectives.

MATHEMATICAL ANALYSIS

Analyzing quiz automation from a mathematical perspective involves considering various aspects such as question generation, scoring, difficulty levels, and statistical analysis of results. Here are some mathematical concepts and considerations related to quiz automation:

Question Generation:

- If the quiz includes randomly generated questions, you can use probability theoryto analyze the likelihood of certain types of questions appearing.

Scalability:

- If the quiz system needs to handle a large number of participants or questions, analyze the scalability of the algorithms used for question generation, scoring, and participant analysis.

Reliability:

- Use mathematical methods to assess the reliability of the quiz, ensuringthat it consistently measures what it intends to measure.

Data Visualization:

- Visualize quiz data using graphs and charts to make patterns and trendsmore apparent.

IMPLEMENTATION

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

class Question {
    private String questionText;
    private List<String> options;
    private String correctOption;

    public Question(String questionText, List<String> options, String correctOption) {
        this.questionText = questionText;
        this.options = options;
        this.correctOption = correctOption;
    }

    public String getQuestionText() {
        return questionText;
    }
```

```
public List<String> getOptions() {
        return options;
    }

    public String getCorrectOption() {
        return correctOption;
    }
}
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
class QuizGenerator {
    private List<Question> questions;
    public QuizGenerator() {
        questions = new ArrayList<>();
        questions.add(new Question("What is the capital of France?", List.of("Berlin",
"Paris", "Madrid", "Rome"), "Paris"));
        questions.add(new Question("Which planet is known as the Red Planet?",
List.of("Mars", "Earth", "Venus", "Jupiter"), "Mars"));
   }
```

```
public List<Question> generateQuiz(int numberOfQuestions) {
        Collections.shuffle(questions);
        return questions.subList(0, numberOfQuestions);
    }
}

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
class QuizApp extends JFrame {
    private QuizGenerator quizGenerator;
    private JTextArea questionTextArea;
    private ButtonGroup optionGroup;

public QuizApp() {
        quizGenerator = new QuizGenerator();
        setTitle("Quiz App");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        initUI();
    }
```

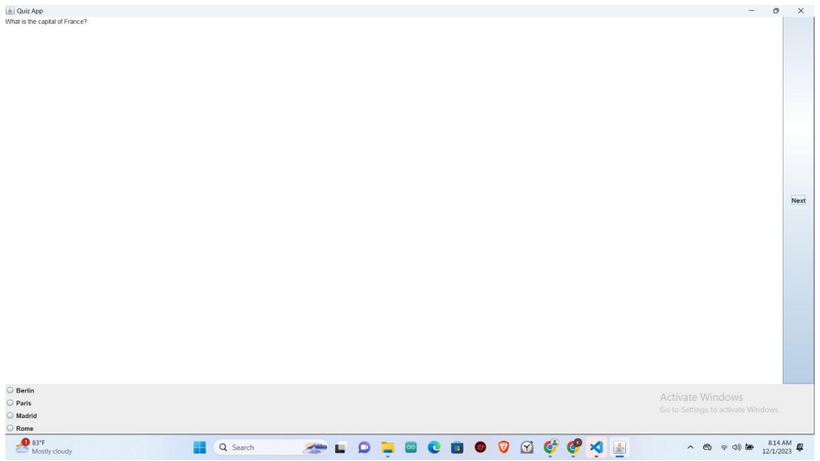
```
private void initUI() {
        setLayout(new BorderLayout());

        questionTextArea = new JTextArea();
        questionTextArea.setEditable(false);
        add(questionTextArea, BorderLayout.CENTER);

        JPanel optionsPanel = new JPanel();
        optionsPanel.setLayout(new GridLayout(4, 1));
        optionGroup = new ButtonGroup();

        for (int i = 0; i < 4; i++) {
            JRadioButton optionButton = new JRadioButton();
            optionGroup.add(optionButton);
            optionsPanel.add(optionButton);
        }
```

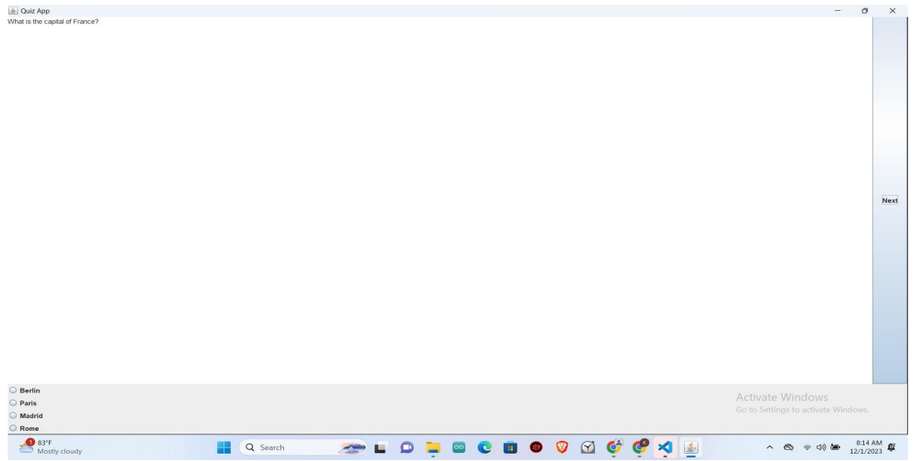
```
add(optionsPanel, BorderLayout.SOUTH);

        JButton nextButton = new JButton("Next");
        nextButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                displayNextQuestion();
            }
        });

        add(nextButton, BorderLayout.EAST);

        displayNextQuestion();
    }
```

```
private void displayNextQuestion() {
        List<Question> quizQuestions = quizGenerator.generateQuiz(1);

        if (!quizQuestions.isEmpty()) {
            Question currentQuestion = quizQuestions.get(0);
            questionTextArea.setText(currentQuestion.getQuestionText());

            List<String> options = currentQuestion.getOptions();
            List<AbstractButton> optionButtons = Collections.list(optionGroup.getElements());

            for (int i = 0; i < options.size(); i++) {
                optionButtons.get(i).setText(options.get(i));
            }
        } else {
            JOptionPane.showMessageDialog(this, "End of quiz!");
            System.exit(0);
        }
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new QuizApp().setVisible(true);
            }
        });
    }
}
```

SCREENSHOTS





DEMONSTRATION

Step 1: Define the Question Class The Question class represents the question that is to be answered. The class includes methods to set and raise questions.

Step 2: Define the QuizGenerator Class The QuizGenenrator class represents questions which is to be generated.. For each person, it maintains a list of options.

Step 3: QuizGenerator() Method This QuizGenerator method involves with the creation of questions along with the options as well. We can generate any number of questions in this QuizGenerator Method.

Step 4: generateQuiz Method() In this method, the questions which are generated in the QuizGenerator Method will be shuffled. by using this method, we can shuffle the number of questions which are generated

Step 5: Define the QuizApp Class The QuizApp class will implement the number of questions and the options which are generated. The questions and the respective options will already be generated in the QuizGenerator Class.

Step 6: initUI() Method According to the count of options mentioned in the QuizGenerator class, here we can set the number of options which are to be generated in the result of our generation of quizzes.

Step 7: Create the Main Class The Main class is the entry point of the program. It creates a JFrame to host the Quiz Generation panel. It prompts the user for input such as the number of Questions, and options.

Step 8: Running the Program When you run the program, it will display a set of quiz questions which are generated.. You can observe the individual questions and their respective options that are generated as the generation of these quizzes progresses. This code provides a simple generation of quizzes and serves as a starting point to create any number of questions that can be generated. The simulation's can be adjusted to explore different types of quizzes.

CONCLUSION

In conclusion, the provided code for the Automatic Quiz Generation using a simple network representation does a commendable job of creating and accessing Quiz format. These quiz which are generated can be much more intricate, involving the application of Network Analysis Techniques. Nevertheless, this code provides a valuable starting point for those interested in exploring these generation of various quizzes.

REFERENCES

- https://www.scribd.com/document/415439848/Quiz-on-Test-Automation
- YOUTUBE : https://www.youtube.com/watch?v=HtXJe1rAZaI

THANK YOU