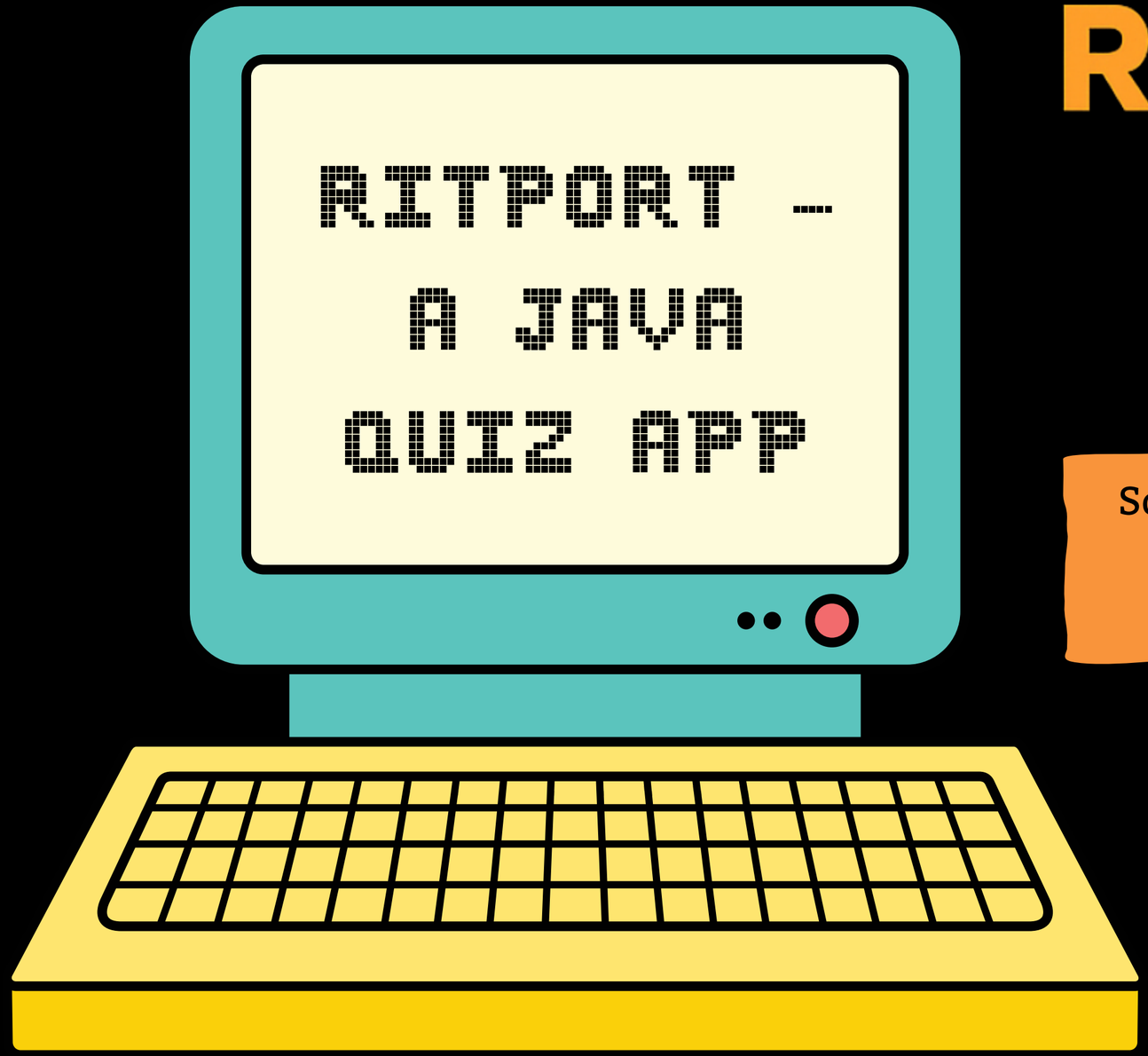
RITPORT - A JAVA QUIZ APP



RITport

Software Development& Problem Solving ||

Spring 2025

Ahmad Saeed
Hishaam Basheer
Sharifa AlMalik

PROJECT OVERVIEW

- Quick summary of the app
- What it's used for (RIT knowledge quiz)
- Tools used: Java, GUI (JavaFX), JUnit

RITport is an airport-themed quiz app designed for RIT students to test their general knowledge. It features a user-friendly interface, leaderboard, timer, and brand elements.

WHAT'S NEW IN ASSIGNMENT 3 AND 4

List all the updates from Assignment 2:

- Leaderboard sorting
- Timer using Java Threads
- JUnit test cases
- New GUI enhancements
- User survey + improvements
- Brand identity: RITport

Leaderboard Enhancement

```java
public String[][] updateLeaderboard(String filename, String username, String score, String timetaken){
    ArrayList<ArrayList<String>> leaderboard = new ArrayList<>();

    ArrayList<String> temp = new ArrayList<>();
    temp.add(username);
    temp.add(score);
    temp.add(timetaken);
    leaderboard.add(temp);

    try{

        BufferedReader bufferedReader = new BufferedReader(new FileReader(filename));
        while ((bufferedReader.readLine() != null)){

            String entry = bufferedReader.readLine();

            String tempStr = "";
            ArrayList<String> tempArr = new ArrayList<>();

            for (int i = 0; i< entry.length(); i++){
                if (entry.charAt(i) == ','){
                    tempArr.add(tempStr);
                    tempStr = "";
                }
                else{
                    tempStr += entry.charAt(i);
                }
            }
            tempArr.add(tempStr);
            leaderboard.add(tempArr);
        }
        leaderboard.sort((arraylist1, arraylist2) -> compare(arraylist1, arraylist2));
        bufferedReader.close();


        FileWriter bufferedWriter = new FileWriter(filename);
        int rank = 1;
        String[][] board = new String[leaderboard.size()][4];
        for (ArrayList<String> entry : leaderboard) {
            bufferedWriter.write("\n"+entry.get(index:0)+","+entry.get(index:1)+","+entry.get(index:2)+"\n");
            board[rank-1][0] = "["+rank+"]";
            for  (int i = 0; i < 3; i++){
                board[rank-1][i+1] = entry.get(i);
            }
            rank++;
        }
        bufferedWriter.flush();
        bufferedWriter.close();
        return board;

    }catch (Exception e) {
        e.printStackTrace();
    }

    return null;
}
```

SORTED BY SCORE AND TIME

Leaderboard Enhancement

```
public int compare(ArrayList<String> list1, ArrayList<String> list2) {
    int num1 = Integer.parseInt(list1.get(index:1));
    int num2 = Integer.parseInt(list2.get(index:1));
    String time1 = list1.get(index:2);
    String time2 = list2.get(index:2);

    if (num1 == num2 && (!(time1.equals(anObject:"Times Up!")) || !(time1.equals(anObject:"Times Up!")))){
        int time1Int = 0;
        int time2Int = 0;
        String tempStr = "";

        for (int i = 0; i< time1.length(); i++){
            if (time1.charAt(i) == ':'){
                time1Int += Integer.parseInt(tempStr)*60;
                tempStr = "";
            }
            else{
                tempStr += time1.charAt(i);
            }
        }
       time1Int += Integer.parseInt(tempStr);
       tempStr = "";

        for (int i = 0; i< time2.length(); i++){
            if (time2.charAt(i) == ':'){
                time1Int += Integer.parseInt(tempStr)*60;
                tempStr = "";
            }
            else{
                tempStr += time2.charAt(i);
            }
        }
        return time1Int-time2Int;
    }
    else return num2-num1;
}
```

This code uses an ArrayList to store and compare the leaderboard data. It allows us to efficiently organize, update, and sort the players' names and completion times in real-time, ensuring the leaderboard is always up-to-date and accurate.

- ArrayList stores player names and times
- Compares and sorts times to determine rankings
- Dynamically updates leaderboard after each quiz completion

LEADERBOARD

This is how the leaderboard will be displayed after completing the quiz, showing the top users along with their completion times.

Leaderboard			
[1]	kjmnj	3	5:52
[2]	we	2	2:55
[3]	testing	1	0:17
[4]	www	1	0:5
[5]	dd	1	0:58
[6]	3d	1	0:58
[7]	ddd	0	Times Up!
[8]	dddd	0	Times Up!
[9]	3dd	0	Times Up!
[10]	ded3	0	0:0
[11]	`d	0	0:57

These are the two leaderboards from Assignment 2 and Assignment 3. The first one shows the original version, and the second one shows the updated version after applying improvements and fixes

Leaderboard			
[1]	E	4	0:9
[2]	a	4	5:52
[3]	c	4	5:53
[5]	fd	2	0:5

Old Version

Updated

TIMER IMPLEMENTATION

- Countdown timer runs during quiz
- Auto-submits answers when time ends
- Faster completion improves ranking
- Time recorded in leaderboard

```
class countdownTimer extends Thread{

    private final Integer numOfQuestions;
    private Label timerLabel;
    private String currtime;
    private int remainingtime;

    public countdownTimer(Integer numOfQuestions, Label timerLabel){
        this.numOfQuestions = numOfQuestions;
        this.timerLabel = timerLabel;
    }

    @Override
    public void run(){

        try{

            remainingtime = numOfQuestions*60;
            do{
                remainingtime--;
                currtime = (remainingtime/60)+":"+(remainingtime%60);
                Platform.runLater(() -> {timerLabel.setText(currtime);});
                Thread.sleep(millis:1000);
                }
            while(remainingtime > 0);
            Platform.runLater(() -> {
                timerLabel.setText("Times Up!");
                func.loadEndCard(username, q1, answers, currIndex, cont, gp, hiddenscore, timerLabel);
                });

        }catch (Exception e) {
        e.printStackTrace();
        }
    }

}

countdownTimer countingDown = new countdownTimer(numOfQuestions,timerLabel);
countingDown.start();
func.loadQuestion(username, currIndex, quizTime, q1, answers, currscore, numOfQuestions, cont, gp, scorecard, hiddenscore, timerLabel);
```

JUNIT TESTING

```
    @Test
    public void testLeaderboardSorting() {
        QuizFunctions functions = new
QuizFunctions();

        ArrayList<String> highScore = new
ArrayList<>();
        highScore.add("P1");
        highScore.add("5");
        highScore.add("2:30");

        ArrayList<String> lowScore = new
ArrayList<>();
        lowScore.add("P2");
        lowScore.add("3");
        lowScore.add("3:00");

        int result = functions.compare(highScore,
lowScore);
        assertTrue("Higher score", result < 0);

        result = functions.compare(lowScore,
highScore);
        assertTrue("Lower score", result > 0);
    }
```

Explanation:

This test simulates two players with different scores and times.

The compare() method checks if sorting is working properly.

Expected Outcome: Players with higher scores should rank above players with lower scores.

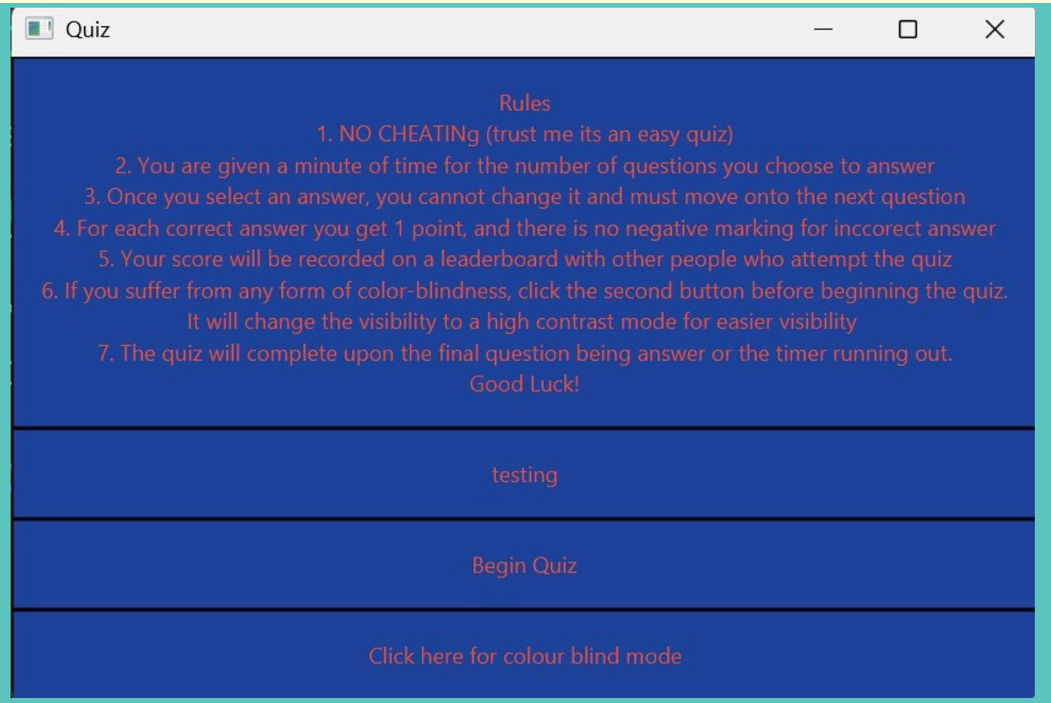
Result:

This test simulates two players with different scores and times.

GRAPHICAL USER INTERFACE (GUI)

Description:

- **This is the main interface of the quiz application.**
- **Includes a list of rules, clearly informing the user about how the quiz works.**
- **Accessible design with a color-blind mode for better visibility.**
- **Buttons included:**
  - **Testing**
  - **Begin Quiz**
  - **Colour Blind Mode**

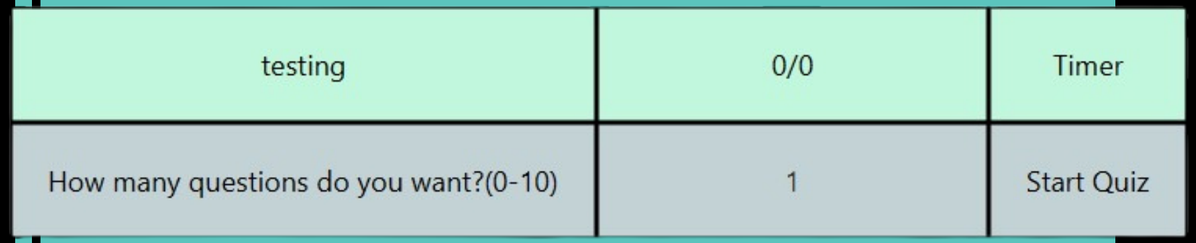


Improvements Made:

- Improved GUI color contrast for better readability
- Added timer warning before quiz starts
- Optimized code for faster performance

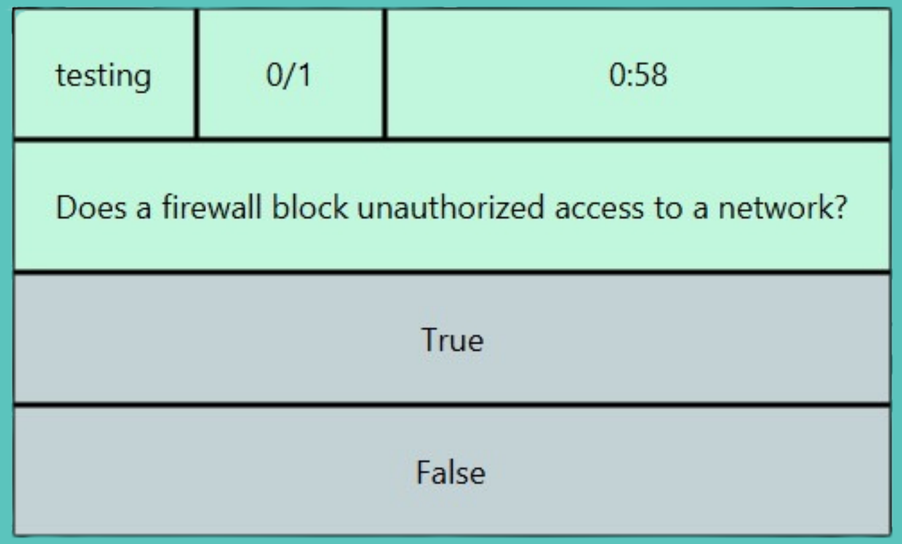
START QUIZ – QUESTION SELECTION SCREEN

- **Choose how many questions to answer (0–10)**
- **Gives user control over quiz length**
- **Clean interface for easy navigation**
- **Encourages flexible learning pace**



QUESTION DISPLAY WITH COUNTDOWN TIMER

- **Timer shown on-screen to track time**
- **Auto-submits when time runs out**



BRANDING - IDENTITY

RITPort

"Your boarding pass to better grades!"

Target Audience : RIT students

Brand Values:

- **Educational – Focused on helping students revise and improve their academic performance.**
- **Engaging – Fun and gamified interface to encourage participation.**
- **Inclusive – Features like color-blind mode promote accessibility.**
- **Efficient – Quick quizzes and leaderboard keep things exciting and competitive.**

BRANDING – LOGO AND BUDGET

Tools Used

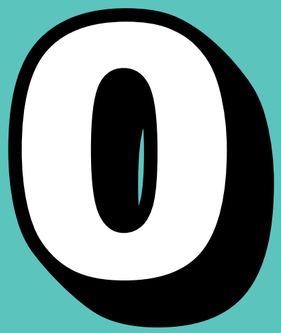
RITport

Logo Design

Survey tool

Visual Studio

Student Dev Time



Total Cost:

CHALLENGES FACED

Sorting leaderboard correctly

Coordinating team work

Incorporating a color-blind mode without affecting the rest of the GUI.

Implementing a reliable timer that syncs accurately with quiz progression.

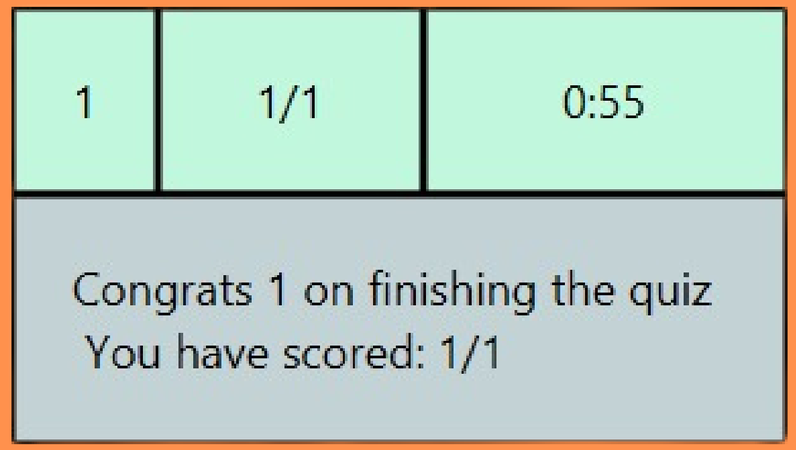
WHAT WE LEARNED

Better understanding of Java GUI

Improved teamwork and time management

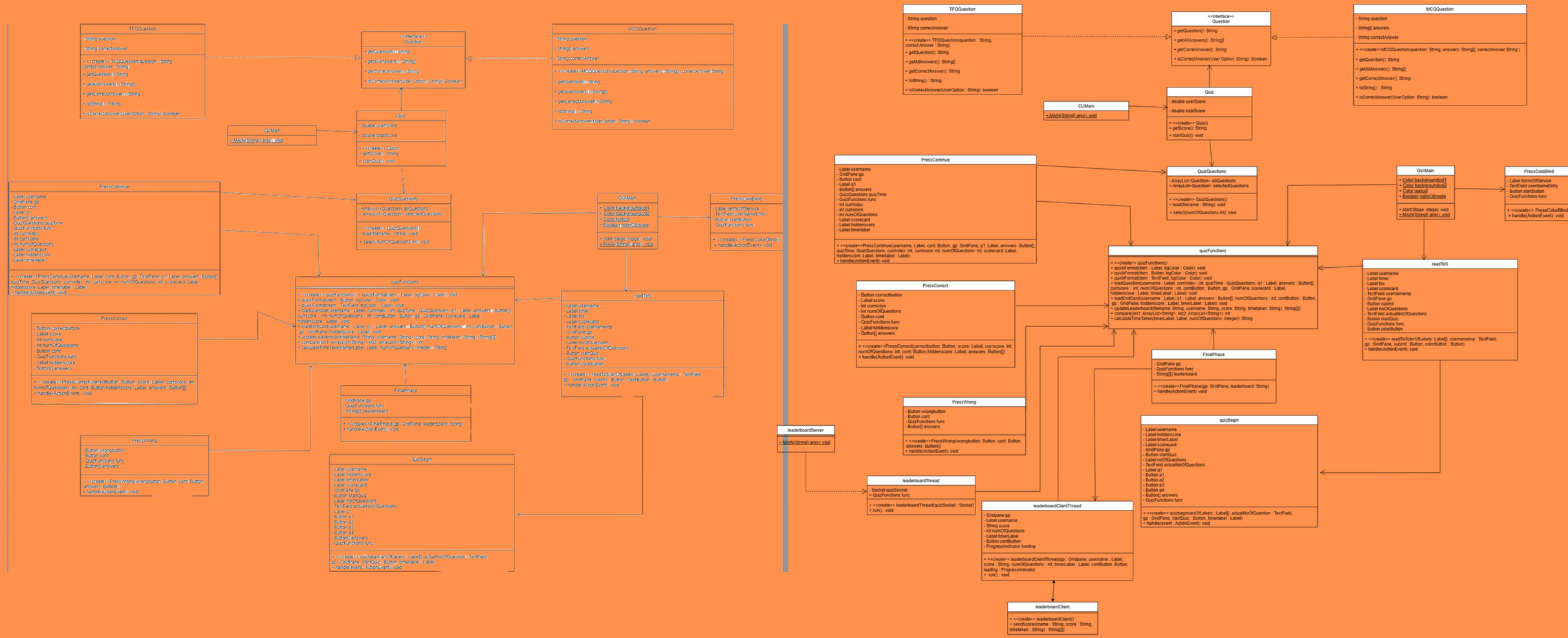
Realized importance of user feedback

SERVER/CLIENT IMPLEMENTATION



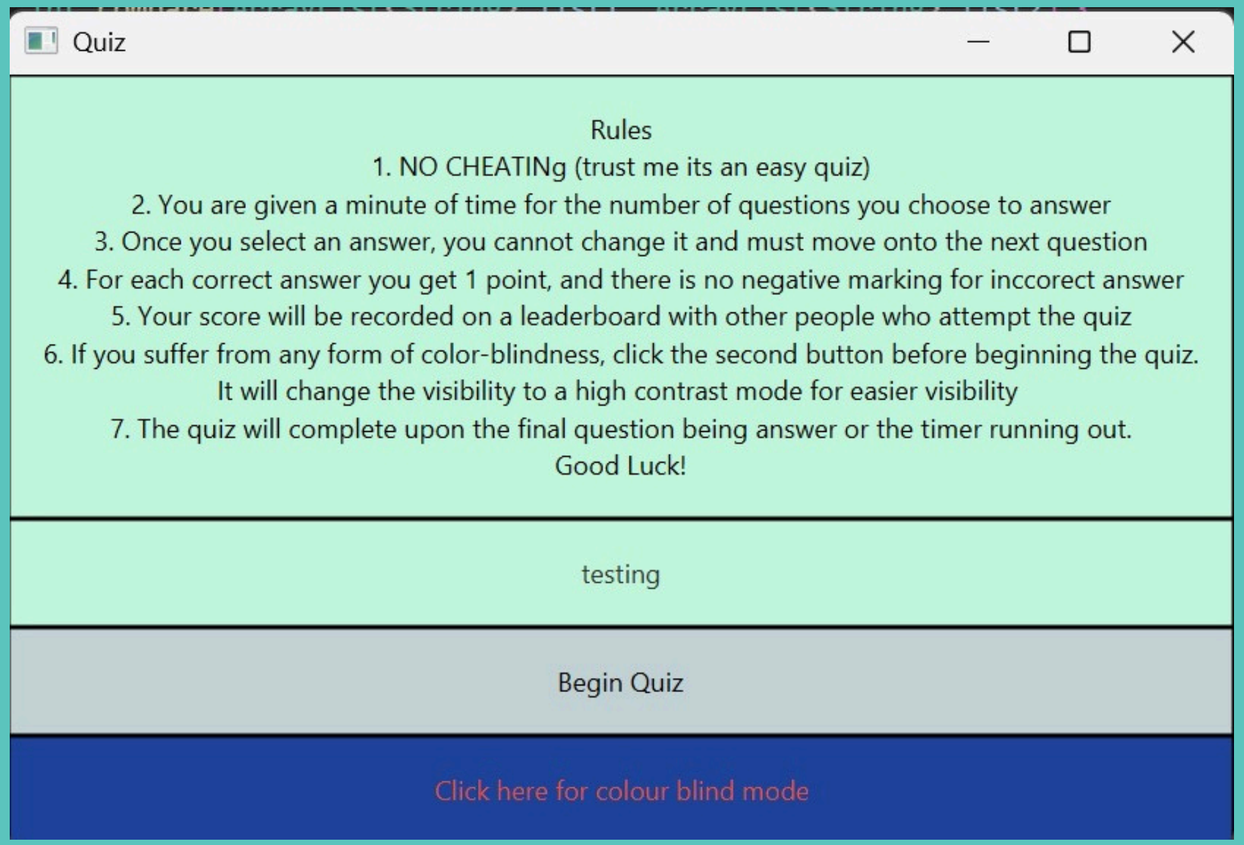
This loading screen appears while the client waits for a response from the server

UML DIAGRAM



This is the UML (Unified Modeling Language) diagram for our project. It shows the structure of the classes, their attributes, methods, and how they are connected. This helps visualize the overall design and logic of our code.

FINAL THOUGHTS



Future Ideas :

- **Music**
- **Add animations**
- **More categories**
- **Smart Text-to-Speech Assistant for People with Reading Disabilities**