**A FIELD PROJECT REPORT ON**

**QUIZ APPLICATION**

**FOR GENERAL KNOWLEDGE ON CODING**

Submitted in partial fulfilment of the requirements for the award of the degree

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE ENGINEERING**

Submitted by

**231FA04B83 : V.YASASVI LAKSHMI SAI**

**231FA04C13 : KRISHNA**

**231FA04C70 : HEMA SRI**

**231FA04F83 : TEJA SRI**

Under the Guidance of

**DR.N SAMEERA**

****

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SCHOOL OF COMPUTING AND INFORMATICS

**VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH**

**(Deemed to be University) Vadlamudi,**

**Guntur -522213, INDIA.**

**April, 2025**

I



**CERTIFICATE**

This is to certify that the field project entitled “QUIZ APPLICATION FOR GENERAL KNOWLEDGE ON CODING” being submitted by (V.Yasasvi Lakshmi sai (231FA04B83)), (T.Krishna (231FA04C13)), (M.Hema sri(231FA04C70)) (R.Teja sri (231FA04F75) in partial fulfilment of Bachelor of Technology in the Department of Computer science engineering, Vignan’s Foundation For Science Technology & Research (Deemed to be University), Vadlamudi, Guntur District, Andhra Pradesh, India

This is a bonafide work carried out by the aforementioned students under my guidance and supervision.

Guide

Project Review Committee HOD,CSE

II



**DECLARATION**

We hereby declare that the project work described in the field project titled **“QUIZ APPLICATION FOR GENERAL KNOWLEDGE IN CODING”** is the result of our own

efforts and investigations.

This project is being submitted under the supervision of **DR.N.Sameera,** in partial fulfillment of the requirements for the Bachelor of Technology (B.Tech.) degree in Computer Science and Engineering at Vignan’s Foundation for Science, Technology and Research (Deemed to be University), Vadlamudi, Guntur, Andhra Pradesh, India.

**V.Yasasvi lakshmi sai ( 231FA04B83 ) Signature**

**T.Krishna ( 231FA04C13) Signature**

**M.Hema sri (231FA04C70) Signature**

**R.Teja sri (231FA04F83) Signature**

Date:

III

|  |  |  |  |
| --- | --- | --- | --- |
| **Chapter No.** | | **Description** | **Page No.** |
| 1 |  | Introduction |  |
|  | 1.1 | Problem Definition | 1 |
|  | 1.2 | Existing System | 1 |
|  | 1.3 | Proposed System | 1 |
| 2 |  | System Requirements | 2 |
|  | 2.1 | Hardware & Software Requirements | 2 |
|  | 2.2 | Software requirements specifications | 2 |
| 3 |  | System Design | 3 |
|  | 3.1 | Module of Systems | 4 |
|  | 3.2 | UML Diagrams | 5-7 |
| 4 |  | Implementation | 8-18 |
| 5 |  | Results | 19 |
|  | 5.1 | Output screens | 20-23 |
| 6 |  | Conclusion | 24 |
| 7 |  | References | 25 |

**4 of 80**

**1.INTRODUCTION**

In today's rapidly evolving technological landscape, a foundational understanding of programming languages is increasingly crucial. Among the plethora of languages available, Java, Python, and C stand out as widely adopted and influential tools in various domains, from web development and data science to system programming and embedded systems. This project aims to develop an interactive quiz application designed to assess and enhance general knowledge in these three fundamental programming languages.

The application will present users with a timed quiz consisting of 10 multiple-choice questions covering core concepts in Java, Python, and C. With a 10-minute time limit, the quiz encourages efficient recall and application of knowledge. Upon completion, the application will provide immediate feedback, displaying the number of attempted questions, the count of correct and incorrect answers, and a visual representation of the user's performance through a circular graph. This intuitive visual aid will offer a quick and comprehensive overview of the user's grasp of the subject matter.

This project serves as a practical exercise in [mention relevant areas like software development, user interface design, data presentation, or specific programming concepts you are focusing on]. By creating this quiz application, we aim to provide a user-friendly tool for self-assessment and knowledge reinforcement in the vital areas of Java, Python, and C programming.

* 1. **Problem Definition**

The goal is to develop a Quiz Application that enables users to test their fundamental knowledge of three popular programming languages: C, Java, and Python. The application should present a series of multiple-choice questions related to these languages and allow users to select answers, calculate scores, and view results**.**

* 1. **Existing System**

As a result, users who are beginners or looking to test their basic knowledge might find it difficult to measure their understanding effectively.

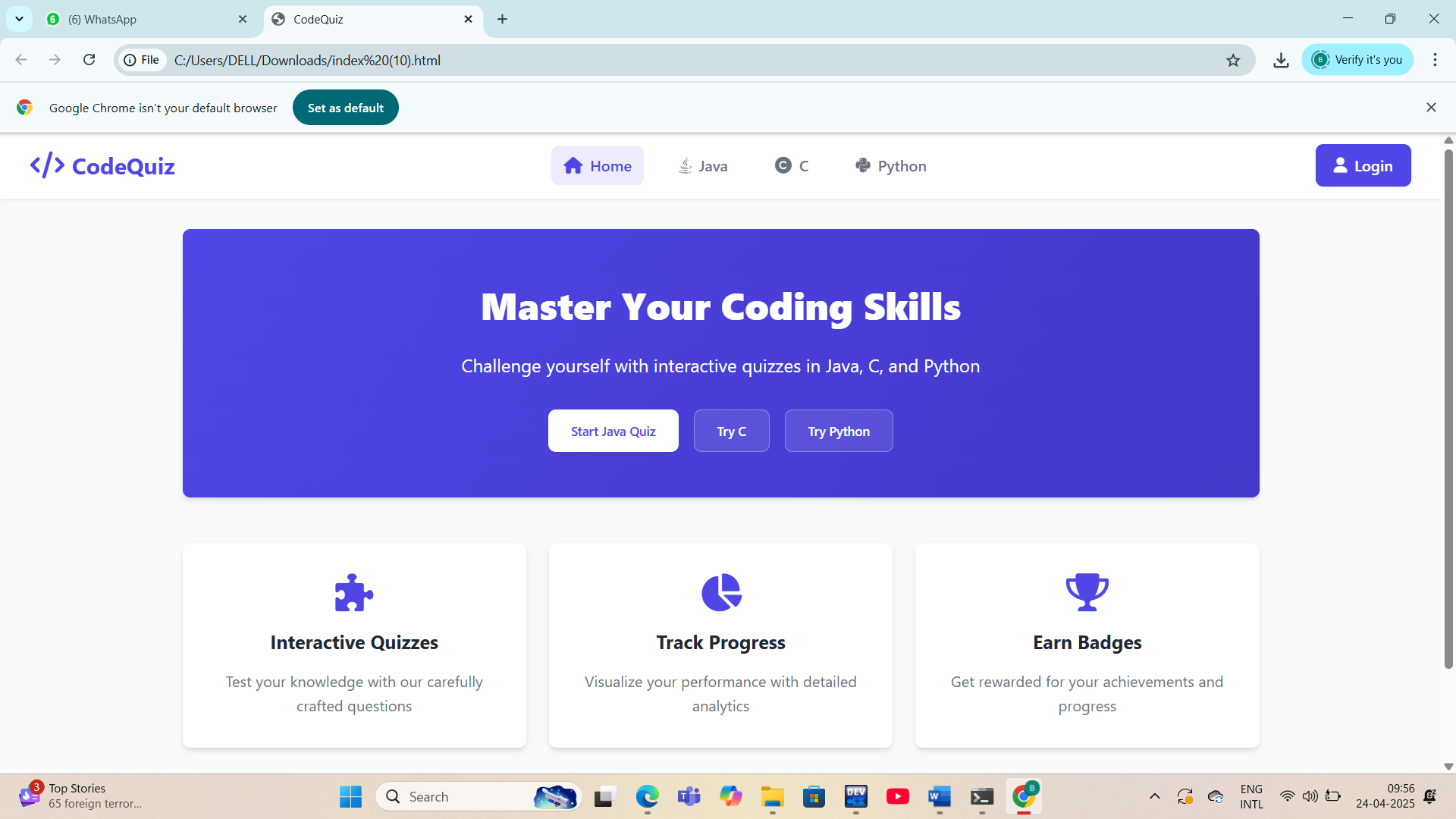
Currently, the existing systems for quiz-based applications in coding mostly rely on simple question-and-answer interfaces, where users can attempt a series of multiple-choice questions. These systems, however, often lack the following key features

* 1. **Proposed System**

1. Multiple Choice Questions (MCQs):
   * Questions are related specifically to C, Java, and Python.
   * Each question has four options with only one correct answer.
2. Language Selection:

5 of 80

* + Users can select a quiz category based on the programming language they want to be tested on.

1. Time Limit Per Question:
   * A timer is set for each question (e.g., 30 or 60 seconds).
   * If the user fails to answer within the time limit, the question is marked as unanswered.
   * Helps simulate real-time test conditions and improves time management skills.
2. Score Calculation:
   * The application calculates and displays the total score at the end based on correct answers.
3. Answer Summary Graph:
   * After the quiz, users are shown a graphical summary (e.g., bar chart or pie chart) of:
     + Total Questions
     + Questions Attempted
     + Correct Answers
     + Incorrect Answers
   * This feature helps users visually analyze their performance.
4. Result Display:
   * The final result includes the total score and optionally the time taken and accuracy percentage.
   * 

6 0f 80

**1.4 Literature Review**

Programming quizzes are becoming increasingly popular as a method for evaluating coding knowledge. A study by **Liu et al. (2019)** explored the impact of coding quizzes on learning programming languages, specifically **C** and **Java**. The research found that quizzes focusing on specific programming language concepts significantly boosted students’ understanding and retention. This highlights the importance of **language-specific quizzes**, which provide a focused, concise, and effective learning environment for beginners and advanced programmers alike.

Additionally, **Kapoor and Sharma (2020)** discussed the effectiveness of **timed coding challenges** in coding competitions, showing that time constraints during quizzes simulate real-world coding scenarios and prepare students for **interviews and exams**. This aligns with our proposed feature of time-limited questions in the application, which can enhance decision-making under pressure.

**7 of 80**

**2 : SYSTEM REQUIREMENTS**

**Front-End Requirements**

The front-end of the Event Management System is responsible for delivering a user-friendly and responsive interface. It allows users to interact with the system for event creation, registration, browsing events, and more. The following technologies are used:

1. HTML (HyperText Markup Language)

* Used to structure web pages and content.
* Forms the skeleton of the application (e.g., forms for registration, login, event details).
* Ensures semantic organization of elements (headers, buttons, tables, etc.).

2. CSS (Cascading Style Sheets)

* Used to style the HTML content.
* Responsible for the visual appearance: layout, fonts, colors, and responsiveness.
* Enables responsive design using media queries for compatibility across devices (desktop, tablet, mobile).

3. JavaScript

* Provides dynamic behavior and interactivity to the web pages.
* Used for client-side validation of forms (e.g., checking if required fields are filled).
* Handles real-time updates (e.g., countdown timers, live search).
* Can interact with APIs or back-end for real-time data fetching without reloading pages (using AJAX or Fetch API).

Back-End Requirements

The back-end of the Event Management System handles the core functionality such as processing data, managing users, storing event details, and ensuring secure transactions. It communicates with the front-end and the database to deliver a seamless user experience.

1: PHP (Hypertext Preprocessor)

PHP is a widely-used open-source scripting language suited for web development.

* Use Cases:
  + Handling form submissions (event registration, login).

8 of 80

* + Communicating with MySQL databases for CRUD operations.

2: Python (with Flask or Django)

Python is a powerful and flexible language, ideal for rapid development.

* Use Cases:
  + Flask (lightweight) or Django (full-featured) can be used for building the server-side logic.
  + REST API development to serve front-end requests.
  + Integrating third-party services (e.g., payment gateways, email APIs).

3: Node.js (JavaScript Runtime)

Node.js allows JavaScript to be used on the server-side, enabling full-stack JS development.

* Use Cases:
  + Building RESTful APIs to handle data operations.
  + Real-time features like chat support or live event updates using WebSockets.
  + Efficient handling of concurrent user requests.

**2.1 Hardware & Software Requirements**

This section outlines the minimum and recommended hardware and software configurations required to develop, deploy, and run the Event Management System efficiently.

9 of 80

Hardware Requirements

|  |  |  |
| --- | --- | --- |
| Component | Minimum Requirement | Recommended Requirement |
| Processor | Intel Core i3 or equivalent | Intel Core i5/i7 or equivalent |
| RAM | 4 GB | 8 GB or higher |
| Storage | 250 GB HDD | 512 GB SSD or higher |
| Display | 1024×768 resolution | Full HD (1920×1080) |
| Internet | Basic broadband connection | High-speed internet |

Software Requirements

|  |  |
| --- | --- |
| Category | Software |
| Operating System | Windows 10/11, macOS, or any modern Linux distribution |
| Web Browser | Google Chrome, Mozilla Firefox, Microsoft Edge |
| Front-End Tools | HTML, CSS, JavaScript, Bootstrap (optional) |
| Back-End Options | PHP (with Apache), Python (Flask/Django), or Node.js |
| Database | MySQL / PostgreSQL / MongoDB |

**10 of 80**

**3: System Design**

System Design outlines how the quiz application is structured — how its components interact, the data flow, and how users interact with the system.

**1. User Interface (UI) - What the user sees and interacts with:**

* **Quiz Start Screen:**
  + Displays instructions (e.g., time limit, number of questions).
  + Allows the user to select a programming language (Java, Python, C).
  + A "Start Quiz" button.
* **Quiz Screen:**
  + Displays the current question clearly.
  + Presents multiple-choice options for each question.
  + A timer displaying the remaining time (updates dynamically).
  + Navigation buttons (e.g., "Next," "Previous" - optional, depending on your desired flow).
  + A progress indicator (e.g., "Question 3 of 10").
* **Result Screen:**
  + Displays the total time taken.
  + Shows the number of questions attempted.
  + Indicates the number of correct answers.
  + Indicates the number of incorrect answers.
  + **Graphical Representation:** A circular chart (pie chart or donut chart) visually representing the percentage of correct and incorrect answers.
  + Option to review attempted questions (with correct answers highlighted).

**2. Application Logic (Backend) - What makes the application work:**

* **Question Management:**
  + **Storage:** A way to store the quiz questions, their options, and the correct answers for each language. This could be:
    - Simple data structures within the application

11 of 80

* + - External files (e.g., JSON, CSV).
    - A database (more scalable for a larger number of questions).
  + **Retrieval:** Logic to fetch a set of 10 random questions for the selected language.
* **Quiz Session Management:**
  + **Initialization:** When a user starts a quiz, a session is created. This involves selecting the questions and starting the timer.
  + **State Tracking:** Keeping track of the current question, the user's answers, and the time elapsed.
  + **Answer Submission:** Handling the user's selection for each question.
  + **Timer Logic:** Implementing the 10-minute timer and triggering the end of the quiz when time runs out.
  + **Scoring:** Evaluating the user's answers against the correct answers.
* **Result Generation:**
  + Calculating the number of attempted, correct, and incorrect answers.
  + Generating the data needed for the graphical representation (e.g., percentages of correct and incorrect answers).

**3. Data Storage (Optional, but recommended for scalability):**

* **Database (e.g., SQLite, PostgreSQL, MySQL):**
  + Storing questions, options, and correct answers, potentially organized by language.
  + Could also store user scores and quiz history if you want to add those features later.
* **File System (e.g., JSON, CSV files):**
  + A simpler way to store quiz data, suitable for a smaller number of questions.

**System Flow:**

1. **User Interaction:** The user opens the application and sees the "Quiz Start Screen."
2. **Language Selection:** The user selects a programming language (Java, Python, or C).
3. **Quiz Initiation:** The user clicks "Start Quiz."
4. **Backend Processing:**

12 of 80

* + The application logic retrieves 10 random questions for the selected language from the data storage.
  + A quiz session is created, and the 10-minute timer starts.

1. **Quiz Display:** The first question is displayed on the "Quiz Screen" along with the timer and options.
2. **Answering Questions:** The user selects an answer and proceeds to the next question. The application logic records their choices.
3. **Time Runs Out or All Questions Answered:**
   * If the timer reaches zero, the quiz ends automatically.
   * If the user answers all 10 questions, the quiz ends.
4. **Result Calculation:** The application logic calculates the results (attempted, correct, incorrect).
5. **Result Display:** The "Result Screen" is displayed, showing the statistics and the circular graph representing the performance.

**3.1. Modules Description**

**A. User Interface (UI) Module**

* Language selection screen
* Quiz screen (displays questions & options)
* Result screen (score + performance chart)
* Simple, beginner-friendly layout

**B. Quiz Controller**

* Controls quiz flow: next question, validate answers
* Manages the timer for each question or the full quiz
* Handles scoring logic and correctness tracking

**C. Question Bank Module**

* Stores questions categorized by language
* Each entry: question, 4 options, correct answer
* Can be stored in a file (CSV, JSON) or database (like SQLite)

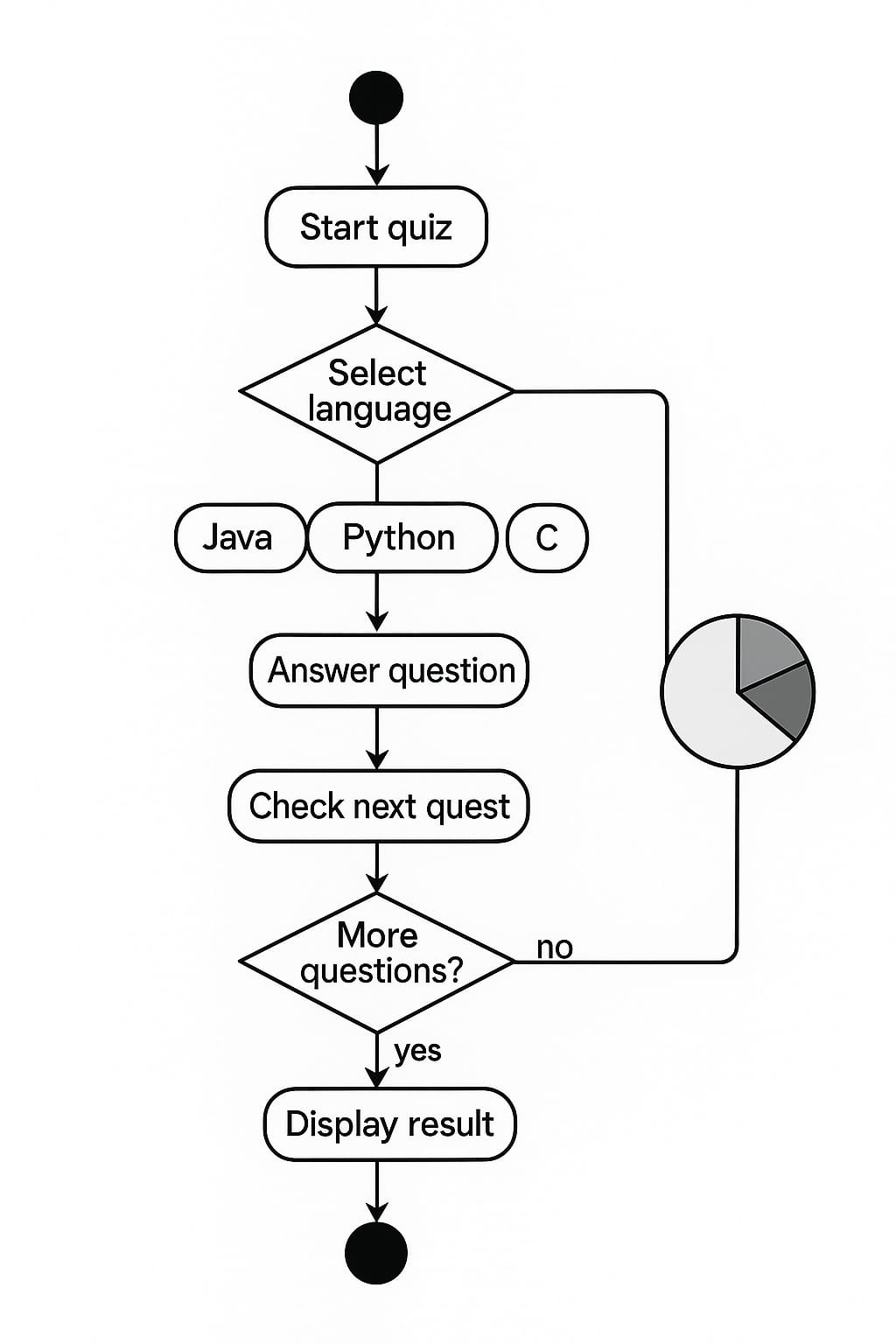
13 of 80

**D. Result & Graph Module**

* Displays result with total score and correct answers
* Uses charts (bar/pie) to show:
  + Questions attempted
  + Correct vs incorrect answers
  + Time taken (optional)

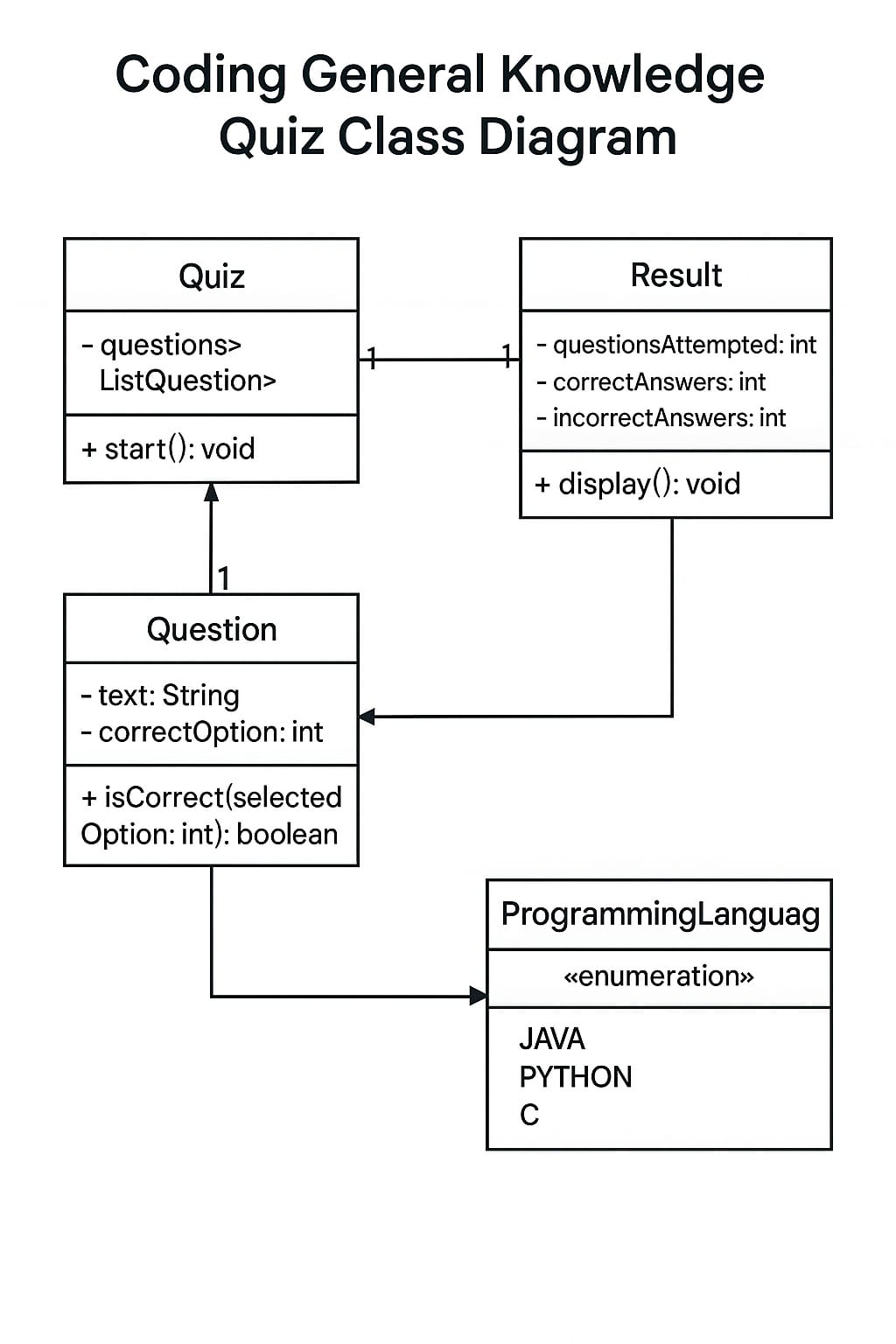
**3.2 UML DIAGRAMS:**

**USE CASE DIAGRAM**

****

**14 of 80**

**CLASS DIAGRAM**

****

**COMPONENT DIAGRAM**

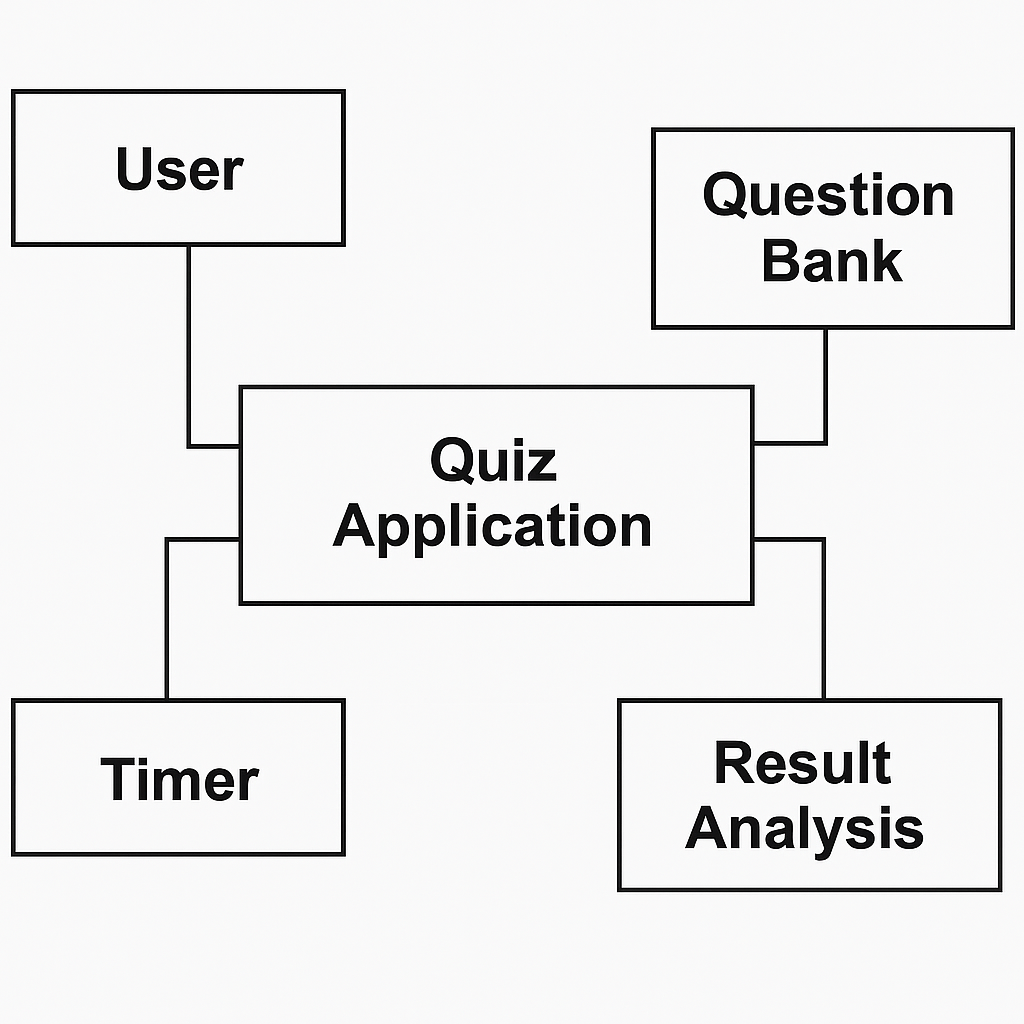
Main Components:

1. User Interface (UI) Component
   * Responsible for interacting with users.
   * Handles login, quiz display, and result visualization.
   * Sends/receives data from the application core.
2. Quiz Manager Component
   * Central control unit of the application.
   * Handles quiz initiation, question selection, and timer control.

15 of 80

* + Connects the UI with the backend logic.

1. Question Bank Component
   * Stores categorized questions based on topics like Java, Python, and C.
   * Supplies questions when requested by the Quiz Manager.
2. Timer Component
   * Tracks time for each question or the entire quiz.
   * Sends timeout events to Quiz Manager to auto-submit or skip questions.
3. Result Processor Component
   * Evaluates user answers.
   * Calculates total score, correct/wrong answers.
   * Passes data to the Graph Generator.
4. Graph Generator Component
   * Converts result data into graphical formats (bar chart, pie chart).
   * Displays performance visually to the user.

****

**16 of 80**

**SEQUENCE DIAGRAM**

Flow of Interaction:

1. User → Login Page:  
   The user initiates the interaction by providing login credentials.
2. Login Page → User Manager:  
   The login page sends the credentials to the user manager for authentication.
3. User Manager → DB:  
   The user manager queries the database to verify credentials.
4. User Manager → Login Page:  
   On success, login confirmation is sent back, and the user is directed to the quiz section.
5. User → Quiz Manager:  
   The user selects a topic and starts the quiz.
6. Quiz Manager → Question Bank:  
   The quiz manager fetches questions for the selected topic.
7. Quiz Manager → Timer:  
   A timer starts as each question is presented to the user.
8. User → Quiz Manager:  
   The user submits answers, which are evaluated in real-time.
9. Quiz Manager → Result Processor:  
   At the end of the quiz, the system calculates the score and prepares the result.
10. Result Processor → Graph Generator:  
    Graphs (correct/wrong answers) are generated and sent back to the user interface.
11. User → Logout:  
    The user logs out or exits after viewing the result.

17 of 80

**4: IMPLEMENTATION**

**4.1 : SAMPLE CODE**

**INDEX CODE**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>CodeQuiz</title>**

**<link rel="stylesheet" href="styles.css">**

**<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">**

**<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>**

**</head> <body>**

**<!-- Navbar -->**

**<nav class="navbar">**

**<div class="logo">**

**<i class="fas fa-code"></i>**

**<span>CodeQuiz</span>**

**</div>**

**<div class="nav-links">**

**<a href="#" class="nav-item active" onclick="loadHome()"><i class="fas fa-home"></i> Home</a>**

**<a href="#" class="nav-item" onclick="loadQuiz('java')"><i class="fab fa-java"></i> Java</a>**

**<a href="#" class="nav-item" onclick="loadQuiz('c')"><i class="fas fa-copyright"></i> C</a>**

**<a href="#" class="nav-item" onclick="loadQuiz('python')"><i class="fab fa-python"></i> Python</a>**

**18 of 80**

**</div>**

**<div class="login-container">**

**<a href="#" class="login-btn"><i class="fas fa-user"></i> <span>Login</span></a>**

**</div>**

**</nav>**

**<!-- Main Content -->**

**<div class="page-container">**

**<div id="content" class="content">**

**<!-- Home Content (Default) -->**

**<div class="home-content">**

**<div class="hero-section">**

**<h1>Master Your Coding Skills</h1>**

**<p>Challenge yourself with interactive quizzes in Java, C, and Python</p>**

**<div class="action-buttons">**

**<button class="primary-btn" onclick="loadQuiz('java')">Start Java Quiz</button>**

**<button class="secondary-btn" onclick="loadQuiz('python')">Try Python</button>**

**</div>**

**</div>**

**<div class="features">**

**<div class="feature-card">**

**<i class="fas fa-puzzle-piece"></i>**

**<h3>Interactive Quizzes</h3>**

**<p>Test your knowledge with our carefully crafted questions</p>**

**</div>**

**19 of 80**

**<div class="feature-card">**

**<i class="fas fa-chart-pie"></i>**

**<h3>Track Progress</h3>**

**<p>Visualize your performance with detailed analytics</p>**

**</div>**

**<div class="feature-card">**

**<i class="fas fa-trophy"></i>**

**<h3>Earn Badges</h3>**

**<p>Get rewarded for your achievements and progress</p>**

**</div>**

**</div>**

**</div>**

**</div>**

**</div>**

**<!-- Footer -->**

**<footer class="footer">**

**<div class="footer-content">**

**<p>&copy; 2025 CodeQuiz. All rights reserved.</p>**

**<div class="social-links">**

**<a href="#"><i class="fab fa-github"></i></a>**

**<a href="#"><i class="fab fa-twitter"></i></a>**

**<a href="#"><i class="fab fa-linkedin"></i></a>**

**</div>**

**</div>**

**</footer>**

**20 of 80**

**<script src="script.js"></script>**

**</body>**

**</html>**

**SCRIPT.JS**

**let currentQuestion = 0;**

**let score = 0;**

**let quizData = [];**

**let unattemptedQuestions = 0;**

**let isLoggedIn = false;**

**let currentUser = null;**

**let timer = null;**

**let timeLeft = 0;**

**let activeLanguage = null;**

**let userBadges = [];**

**// Load Home Page**

**function loadHome() {**

**clearInterval(timer);**

**activeLanguage = null;**

**document.querySelectorAll('.nav-item').forEach(item => {**

**item.classList.remove('active');**

**});**

**document.querySelector('.nav-item:first-child').classList.add('active');**

**21 of 80**

**const content = document.getElementById('content');**

**content.innerHTML = `**

**<div class="home-content">**

**<div class="hero-section">**

**<h1>Master Your Coding Skills</h1>**

**<p>Challenge yourself with interactive quizzes in Java, C, and Python</p>**

**<div class="action-buttons">**

**<button class="primary-btn" onclick="loadQuiz('java')">Start Java Quiz</button>**

**<button class="secondary-btn" onclick="loadQuiz('c')">Try C</button>**

**<button class="secondary-btn" onclick="loadQuiz('python')">Try Python</button>**

**</div>**

**</div>**

**<div class="features">**

**<div class="feature-card">**

**<i class="fas fa-puzzle-piece"></i>**

**<h3>Interactive Quizzes</h3>**

**<p>Test your knowledge with our carefully crafted questions</p>**

**</div>**

**<div class="feature-card">**

**<i class="fas fa-chart-pie"></i>**

**<h3>Track Progress</h3>**

**<p>Visualize your performance with detailed analytics</p>**

**</div>**

**<div class="feature-card">**

**<i class="fas fa-trophy"></i>**

**<h3>Earn Badges</h3>**

**21 of 80**

**<p>Get rewarded for your achievements and progress</p>**

**</div>**

**</div>**

**${isLoggedIn ? renderUserBadges() : ''}**

**</div>**

**`;**

**}**

**// Render User Badges**

**function renderUserBadges() {**

**if (!userBadges || userBadges.length === 0) {**

**return '';**

**}**

**return `**

**<div class="badges-section">**

**<h2>Your Badges</h2>**

**<div class="badges-container">**

**${userBadges.map(badge => `**

**<div class="badge-item" title="${badge.description}">**

**<div class="badge-icon ${badge.type}">**

**<i class="${badge.icon}"></i>**

**</div>**

**<span>${badge.name}</span>**

**</div>**

**`).join('')}**

**</div>**

**22 of 80**

**</div>**

**`;**

**}**

**// Award Badges Based on Performance**

**function awardBadges(language, percentage) {**

**const newBadges = [];**

**// Performance badges**

**if (percentage >= 90) {**

**newBadges.push({**

**name: `${capitalizeFirstLetter(language)} Master`,**

**type: 'gold',**

**icon: 'fas fa-crown',**

**description: `Achieved an outstanding score of ${percentage}% in the ${language.toUpperCase()} quiz`**

**});**

**} else if (percentage >= 70) {**

**newBadges.push({**

**name: `${capitalizeFirstLetter(language)} Pro`,**

**type: 'silver',**

**icon: 'fas fa-award',**

**description: `Achieved a great score of ${percentage}% in the ${language.toUpperCase()} quiz`**

**});**

**} else if (percentage >= 50) {**

**newBadges.push({**

**name: `${capitalizeFirstLetter(language)} Apprentice`,**

**23 of 80**

**type: 'bronze',**

**icon: 'fas fa-medal',**

**description: `Achieved a solid score of ${percentage}% in the ${language.toUpperCase()} quiz`**

**});**

**}**

**// First attempt badge**

**const hasTakenBefore = userBadges.some(badge =>**

**badge.name.toLowerCase().includes(language.toLowerCase())**

**);**

**if (!hasTakenBefore) {**

**newBadges.push({**

**name: `${capitalizeFirstLetter(language)} Explorer`,**

**type: 'blue',**

**icon: 'fas fa-compass',**

**description: `Completed your first ${language.toUpperCase()} quiz`**

**});**

**}**

**// Perfect score badge**

**if (percentage === 100) {**

**newBadges.push({**

**name: `Perfect ${capitalizeFirstLetter(language)}`,**

**type: 'rainbow',**

**icon: 'fas fa-star',**

**24 of 80**

**description: `Achieved a perfect 100% score in the ${language.toUpperCase()} quiz`**

**});**

**}**

**// Add new badges to user's collection**

**newBadges.forEach(badge => {**

**// Check if badge already exists**

**const badgeExists = userBadges.some(existingBadge =>**

**existingBadge.name === badge.name**

**);**

**if (!badgeExists) {**

**userBadges.push(badge);**

**}**

**});**

**return newBadges;**

**}**

**// Helper function to capitalize first letter**

**function capitalizeFirstLetter(string) {**

**return string.charAt(0).toUpperCase() + string.slice(1);**

**}**

**// Show Registration/Login Popup**

**function showLoginPopup(language) {**

**25 of 80**

**const overlay = document.createElement('div');**

**overlay.className = 'overlay';**

**const popup = document.createElement('div');**

**popup.className = 'popup';**

**popup.innerHTML = `**

**<h2>Join CodeQuiz</h2>**

**<form id="loginForm">**

**<div class="form-group">**

**<label for="username">Username</label>**

**<input type="text" id="username" placeholder="Choose a username" required>**

**</div>**

**<div class="form-group">**

**<label for="email">Email</label>**

**<input type="email" id="email" placeholder="Your email address" required>**

**</div>**

**<div class="form-group">**

**<label for="password">Password</label>**

**<input type="password" id="password" placeholder="Create a password" required>**

**</div>**

**<div class="buttons">**

**<button type="button" onclick="closePopup()">Cancel</button>**

**<button type="submit">Get Started</button>**

**</div>**

**</form>**

**`**

**document.body.appendChild(overlay);**

**26 of 80**

**document.body.appendChild(popup);**

**// Store the language to load after login**

**document.getElementById('loginForm').addEventListener('submit', function(e) {**

**e.preventDefault();**

**const username = document.getElementById('username').value;**

**const email = document.getElementById('email').value;**

**// Simple registration/login process**

**isLoggedIn = true;**

**currentUser = { username, email };**

**// Update login button in navbar**

**document.querySelector('.login-btn span').textContent = username;**

**// Close popup and load the quiz**

**closePopup();**

**initializeQuiz(language);**

**// Show welcome toast**

**showToast(`Welcome, ${username}! Your quiz is ready.`, 'success');**

**});**

**}**

**// Show Toast Notification**

**function showToast(message, type = 'info') {**

**// Remove existing toast if any**

**27 of 80**

**const existingToast = document.querySelector('.toast');**

**if (existingToast) {**

**document.body.removeChild(existingToast);**

**}**

**const toast = document.createElement('div');**

**toast.className = `toast ${type}`;**

**toast.innerHTML = `**

**<div class="toast-content">**

**<i class="${type === 'success' ? 'fas fa-check-circle' : 'fas fa-info-circle'}"></i>**

**<span>${message}</span>**

**</div>**

**<button onclick="this.parentNode.remove()">**

**<i class="fas fa-times"></i>**

**</button>**

**`;**

**document.body.appendChild(toast);**

**// Auto remove after 5 seconds**

**setTimeout(() => {**

**if (document.body.contains(toast)) {**

**document.body.removeChild(toast);**

**}**

**}, 5000);**

**}**

**28 of 80**

**// Close Popup**

**function closePopup() {**

**const overlay = document.querySelector('.overlay');**

**const popup = document.querySelector('.popup');**

**if (overlay) document.body.removeChild(overlay);**

**if (popup) document.body.removeChild(popup);**

**}**

**// Check if user is logged in before starting quiz**

**function loadQuiz(language) {**

**// Update active nav item**

**document.querySelectorAll('.nav-item').forEach(item => {**

**item.classList.remove('active');**

**});**

**document.querySelector(`.nav-item[onclick="loadQuiz('${language}')"]`).classList.add('active');**

**activeLanguage = language;**

**if (!isLoggedIn) {**

**showLoginPopup(language);**

**} else {**

**initializeQuiz(language);**

**}**

**}**

**// Initialize Quiz Data**

**29 of 80**

**function initializeQuiz(language) {**

**clearInterval(timer);**

**if (language === 'java') {**

**quizData = [**

**{**

**question: "What is the size of a char in Java?",**

**options: ["1 byte", "2 bytes", "4 bytes"],**

**answer: "2 bytes",**

**attempted: false**

**},**

**{**

**question: "Which of the following is not a Java keyword?",**

**options: ["class", "struct", "interface"],**

**answer: "struct",**

**attempted: false**

**},**

**{**

**question: "What is the default value of a boolean in Java?",**

**options: ["true", "false", "null"],**

**answer: "false",**

**attempted: false**

**},**

**{**

**question: "Which method is used to start a thread in Java?",**

**options: ["start()", "run()", "execute()"],**

**answer: "start()",**

**30 of 80**

**attempted: false**

**},**

**{**

**question: "What is the superclass of all classes in Java?",**

**options: ["Object", "Super", "Parent"],**

**answer: "Object",**

**attempted: false**

**},**

**{**

**question: "Which keyword is used to define a constant in Java?",**

**options: ["const", "final", "static"],**

**answer: "final",**

**attempted: false**

**},**

**{**

**question: "What is the output of `System.out.println(5 + '5')` in Java?",**

**options: ["55", "10", "Error"],**

**answer: "55",**

**attempted: false**

**},**

**{**

**question: "Which collection class allows duplicate elements?",**

**options: ["Set", "List", "Map"],**

**answer: "List",**

**attempted: false**

**},**

**{**

**31 of 80**

**question: "What is the default value of an int in Java?",**

**options: ["0", "1", "null"],**

**answer: "0",**

**attempted: false**

**},**

**{**

**question: "Which of the following is a marker interface in Java?",**

**options: ["Runnable", "Serializable", "Comparator"],**

**answer: "Serializable",**

**attempted: false**

**}**

**];**

**} else if (language === 'c') {**

**quizData = [**

**{**

**question: "What is the output of sizeof(int) in C?",**

**options: ["2 bytes", "4 bytes", "8 bytes"],**

**answer: "4 bytes",**

**attempted: false**

**},**

**{**

**question: "Which of the following is used to allocate memory in C?",**

**options: ["malloc", "new", "allocate"],**

**answer: "malloc",**

**attempted: false**

**},**

**{**

**32 of 80**

**question: "What is the correct syntax for a function in C?",**

**options: ["function myFunc() {}", "void myFunc() {}", "def myFunc() {}"],**

**answer: "void myFunc() {}",**

**attempted: false**

**},**

**{**

**question: "What is the output of `printf('%d', 5 / 2)` in C?",**

**options: ["2", "2.5", "Error"],**

**answer: "2",**

**attempted: false**

**},**

**{**

**question: "Which header file is used for input/output in C?",**

**options: ["<stdio.h>", "<iostream>", "<input.h>"],**

**answer: "<stdio.h>",**

**attempted: false**

**},**

**{**

**question: "What is the correct way to access a structure member?",**

**options: ["structure->member", "structure.member", "structure::member"],**

**answer: "structure.member",**

**attempted: false**

**},**

**{**

**question: "What does the 'const' keyword do in C?",**

**options: ["Defines a constant variable", "Creates a constructor", "Allocates constant memory"],**

**answer: "Defines a constant variable",**

**33 of 80**

**attempted: false**

**},**

**{**

**question: "Which operator is used for pointer dereferencing?",**

**options: ["&", "\*", "->"],**

**answer: "\*",**

**attempted: false**

**},**

**{**

**question: "What is the purpose of 'void' in a function declaration?",**

**options: ["No parameters required", "No return value", "Both A and B can be correct"],**

**answer: "Both A and B can be correct",**

**attempted: false**

**},**

**{**

**question: "Which function is used to read a character in C?",**

**options: ["getc()", "scan()", "readchar()"],**

**answer: "getc()",**

**attempted: false**

**}**

**];**

**} else if (language === 'python') {**

**quizData = [**

**{**

**question: "What is the correct way to create a function in Python?",**

**options: ["function myFunc():", "def myFunc():", "create myFunc():"],**

**answer: "def myFunc():",**

**34 of 80**

**attempted: false**

**},**

**{**

**question: "Which of these is NOT a Python data type?",**

**options: ["list", "tuple", "array"],**

**answer: "array",**

**attempted: false**

**},**

**{**

**question: "How do you create a dictionary in Python?",**

**options: ["{key:value}", "[key:value]", "dict(key=value)"],**

**answer: "{key:value}",**

**attempted: false**

**},**

**{**

**question: "What does pip stand for in Python?",**

**options: ["Preferred Installer Program", "Python Installation Package", "Package Installation Python"],**

**answer: "Preferred Installer Program",**

**attempted: false**

**},**

**{**

**question: "How is a code block indicated in Python?",**

**options: ["Brackets", "Indentation", "Parentheses"],**

**answer: "Indentation",**

**attempted: false**

**},**

**{**

**35 of 80**

**question: "Which of the following is a mutable data type in Python?",**

**options: ["String", "Tuple", "List"],**

**answer: "List",**

**attempted: false**

**},**

**{**

**question: "What is the output of `print(3 \* '7')` in Python?",**

**options: ["21", "777", "Error"],**

**answer: "777",**

**attempted: false**

**},**

**{**

**question: "Which module is used for working with regular expressions in Python?",**

**options: ["regex", "re", "pyregex"],**

**answer: "re",**

**attempted: false**

**},**

**{**

**question: "Which of the following is NOT a valid method for a list in Python?",**

**options: ["append()", "insert()", "sort()"],**

**answer: "insert()",**

**attempted: false**

**},**

**{**

**question: "What is the correct Python syntax to import the Math module?",**

**options: ["import math", "include math", "using math"],**

**answer: "import math",**

**36 of 80**

**attempted: false**

**}**

**];**

**}**

**// Reset variables**

**currentQuestion = 0;**

**score = 0;**

**unattemptedQuestions = quizData.length;**

**timeLeft = 60 \* 10; // 10 minutes in seconds**

**// Update quiz UI**

**renderQuiz();**

**startTimer();**

**}**

**// Start Quiz Timer**

**function startTimer() {**

**clearInterval(timer);**

**timer = setInterval(() => {**

**timeLeft--;**

**if (timeLeft <= 0) {**

**clearInterval(timer);**

**showResults();**

**} else {**

**updateTimerDisplay();**

**37 of 80**

**}**

**}, 1000);**

**updateTimerDisplay();**

**}**

**// Update Timer Display**

**function updateTimerDisplay() {**

**const minutes = Math.floor(timeLeft / 60);**

**const seconds = timeLeft % 60;**

**const timerElement = document.querySelector('.timer span');**

**if (timerElement) {**

**timerElement.textContent = `${minutes}:${seconds < 10 ? '0' : ''}${seconds}`;**

**}**

**}**

**// Render Quiz Question**

**function renderQuiz() {**

**const content = document.getElementById('content');**

**const question = quizData[currentQuestion];**

**const progressPercentage = (currentQuestion / quizData.length) \* 100;**

**content.innerHTML = `**

**<div class="quiz-container">**

**<div class="progress-bar">**

**<div class="progress" style="width: ${progressPercentage}%"></div>**

**38 of 80**

**</div>**

**<div class="question-header">**

**<span class="question-counter">Question ${currentQuestion + 1} of ${quizData.length}</span>**

**<div class="timer"><i class="fas fa-clock"></i> <span>10:00</span></div>**

**</div>**

**<div class="quiz-question">**

**<p>${question.question}</p>**

**<div id="options" class="options">**

**${question.options.map((option, index) => `**

**<div class="option">**

**<input type="radio" id="option${index}" name="answer" value="${option}">**

**<label for="option${index}">${option}</label>**

**</div>**

**`).join('')}**

**</div>**

**</div>**

**<div class="button-container">**

**<button class="button skip-button" onclick="skipQuestion()">**

**<i class="fas fa-forward"></i> Skip**

**</button>**

**<button class="button next-button" onclick="checkAnswer()">**

**<i class="fas fa-arrow-right"></i> ${currentQuestion === quizData.length - 1 ? 'Finish' : 'Next'}**

**39 of 80**

**</button>**

**</div>**

**</div>**

**`;**

**updateTimerDisplay();**

**}**

**// Skip Question**

**function skipQuestion() {**

**quizData[currentQuestion].attempted = true;**

**currentQuestion++;**

**if (currentQuestion >= quizData.length) {**

**showResults();**

**} else {**

**renderQuiz();**

**}**

**}**

**// Check Answer**

**function checkAnswer() {**

**const selectedOption = document.querySelector('input[name="answer"]:checked');**

**if (selectedOption) {**

**const answer = selectedOption.value**

**if (answer === quizData[currentQuestion].answer) {**

**40 of 80**

**score++;**

**}**

**quizData[currentQuestion].attempted = true;**

**unattemptedQuestions--;**

**currentQuestion++;**

**if (currentQuestion >= quizData.length) {**

**showResults();**

**} else {**

**renderQuiz();**

**}**

**} else {**

**showToast("Please select an answer!", "info");**

**}**

**}**

**// Show Quiz Results**

**function showResults() {**

**clearInterval(timer);**

**const content = document.getElementById('content');**

**const percentage = Math.round((score / quizData.length) \* 100);**

**const attempted = quizData.length - unattemptedQuestions;**

**// Award badges**

**const newBadges = awardBadges(activeLanguage, percentage);**

**41 of 80**

**content.innerHTML = `**

**<div class="result-container">**

**<div class="result-header">**

**<h2>Quiz Completed!</h2>**

**<p>Here's how you performed in the ${activeLanguage.toUpperCase()} quiz</p>**

**</div>**

**<div class="user-info">**

**<h3>${currentUser.username}</h3>**

**<p>${currentUser.email}</p>**

**</div>**

**<div class="score-summary">**

**<div class="score-card">**

**<h3>${attempted}</h3>**

**<p>Questions Attempted</p>**

**</div>**

**<div class="score-card">**

**<h3>${score}</h3>**

**<p>Correct Answers</p>**

**</div>**

**<div class="score-card total">**

**<h3>${percentage}%</h3>**

**<p>Overall Score</p>**

**</div>**

**</div>**

**42 of 80**

**<div class="chart-container">**

**<canvas id="resultsChart"></canvas>**

**</div>**

**${newBadges.length > 0 ? `**

**<div class="new-badges-container">**

**<h3>Badges Earned</h3>**

**<div class="badges-row">**

**${newBadges.map(badge => `**

**<div class="badge-item" title="${badge.description}">**

**<div class="badge-icon ${badge.type}">**

**<i class="${badge.icon}"></i>**

**</div>**

**<span>${badge.name}</span>**

**</div>**

**`).join('')}**

**</div>**

**</div>**

**` : ''}**

**<button class="action-button" onclick="loadHome()">**

**<i class="fas fa-home"></i> Back to Home**

**</button>**

**<button class="action-button" onclick="loadQuiz('${activeLanguage}')">**

**<i class="fas fa-redo"></i> Try Again**

**</button>**

**</div>**

**43 of 80**

**;**

**// Create results chart**

**createResultsChart();**

**}**

**// Create Results Chart**

**function createResultsChart() {**

**const ctx = document.getElementById('resultsChart').getContext('2d');**

**const correct = score;**

**const incorrect = quizData.length - unattemptedQuestions - score;**

**const skipped = unattemptedQuestions;**

**new Chart(ctx, {**

**type: 'doughnut',**

**data: {**

**labels: ['Correct', 'Incorrect', 'Skipped'],**

**datasets: [{**

**data: [correct, incorrect, skipped],**

**backgroundColor: [**

**'#4CAF50', // Bright green for correct**

**'#F44336', // Bright red for incorrect**

**'#9E9E9E' // Grey for skipped**

**],**

**borderWidth: 0**

**}]**

**},**

**44 of 80**

**options: {**

**responsive: true,**

**maintainAspectRatio: false,**

**plugins: {**

**legend: {**

**position: 'bottom'**

**}**

**},**

**cutout: '65%'**

**}**

**});**

**}**

**// Add styles for toast notifications and badges**

**const customStyles = document.createElement('style');**

**customStyles.textContent = `**

**.toast {**

**position: fixed;**

**bottom: 20px;**

**right: 20px;**

**padding: 1rem;**

**border-radius: var(--border-radius);**

**background-color: var(--white);**

**box-shadow: 0 4px 12px rgba(0, 0, 0, 0.15);**

**display: flex;**

**align-items: center;**

**justify-content: space-between;**

**45 of 80**

**z-index: 1000;**

**min-width: 300px;**

**max-width: 400px;**

**animation: slideIn 0.3s ease forwards;**

**}**

**.toast.success {**

**border-left: 4px solid var(--success);**

**}**

**.toast.error {**

**border-left: 4px solid var(--error);**

**}**

**.toast.info {**

**border-left: 4px solid var(--primary);**

**}**

**.toast-content {**

**display: flex;**

**align-items: center;**

**gap: 0.8rem;**

**}**

**.toast-content i {**

**font-size: 1.5rem;**

**}**

**46 of 80**

**.toast-content i.fa-check-circle {**

**color: var(--success);**

**}**

**.toast-content i.fa-info-circle {**

**color: var(--primary);**

**}**

**.toast button {**

**background: none;**

**border: none;**

**color: var(--gray);**

**cursor: pointer;**

**font-size: 1rem;**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**padding: 0.3rem;**

**}**

**.toast button:hover {**

**color: var(--dark);**

**}**

**@keyframes slideIn {**

**from {**

**47 of 80**

**transform: translateX(100%);**

**opacity: 0;**

**}**

**to {**

**transform: translateX(0);**

**opacity: 1;**

**}**

**}**

**.option {**

**position: relative;**

**margin-bottom: 0.8rem;**

**}**

**.option input[type="radio"] {**

**position: absolute;**

**opacity: 0;**

**}**

**.option label {**

**display: block;**

**padding: 1rem;**

**border: 1px solid #e5e7eb;**

**border-radius: var(--border-radius);**

**cursor: pointer;**

**transition: var(--transition);**

**}**

**48 of 80**

**.option label:hover {**

**background-color: rgba(79, 70, 229, 0.05);**

**border-color: var(--primary);**

**}**

**.option input[type="radio"]:checked + label {**

**background-color: rgba(79, 70, 229, 0.1);**

**border-color: var(--primary);**

**font-weight: 500;**

**}**

**/\* Badge styles \*/**

**.badges-section {**

**margin-top: 2rem;**

**padding: 2rem;**

**background-color: var(--white);**

**border-radius: var(--border-radius);**

**box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);**

**}**

**.badges-section h2 {**

**margin-bottom: 1.5rem;**

**text-align: center;**

**}**

**.badges-container, .badges-row {**

**display: flex;**

**49 of 80**

**flex-wrap: wrap;**

**gap: 1.5rem;**

**justify-content: center;**

**}**

**.badge-item {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**width: 100px;**

**cursor: pointer;**

**transition: transform 0.2s;**

**}**

**.badge-item:hover {**

**transform: translateY(-5px);**

**}**

**.badge-icon {**

**width: 60px;**

**height: 60px;**

**border-radius: 50%;**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**margin-bottom: 0.5rem;**

**font-size: 1.8rem;**

**50 of 80**

**color: white;**

**}**

**.badge-icon.gold {**

**background: linear-gradient(45deg, #FFD700, #FFA500);**

**box-shadow: 0 0 10px rgba(255, 215, 0, 0.5);**

**}**

**.badge-icon.silver {**

**background: linear-gradient(45deg, #C0C0C0, #A9A9A9);**

**box-shadow: 0 0 10px rgba(192, 192, 192, 0.5);**

**}**

**.badge-icon.bronze {**

**background: linear-gradient(45deg, #CD7F32, #8B4513);**

**box-shadow: 0 0 10px rgba(205, 127, 50, 0.5);**

**}**

**.badge-icon.blue {**

**background: linear-gradient(45deg, #1E90FF, #0000CD);**

**box-shadow: 0 0 10px rgba(30, 144, 255, 0.5);**

**}**

**.badge-icon.rainbow {**

**background: linear-gradient(45deg, #FF0000, #FF7F00, #FFFF00, #00FF00, #0000FF, #4B0082, #8B00FF);**

**box-shadow: 0 0 10px rgba(255, 0, 0, 0.5);**

**}**

**51 of 80**

**.badge-item span {**

**font-size: 0.9rem;**

**font-weight: 500;**

**text-align: center;**

**}**

**.new-badges-container {**

**margin: 2rem 0;**

**padding: 1.5rem;**

**background-color: #FFF9E5;**

**border-radius: var(--border-radius);**

**border-left: 4px solid #FFD700;**

**}**

**.new-badges-container h3 {**

**margin-bottom: 1rem;**

**color: #B8860B;**

**}**

**@keyframes badgePulse {**

**0% { transform: scale(1); }**

**50% { transform: scale(1.1); }**

**100% { transform: scale(1); }**

**}**

**.new-badges-container .badge-item {**

**52 of 80**

**animation: badgePulse 1.5s infinite;**

**}**

**`;**

**document.head.appendChild(customStyles);**

**// Initialize the application**

**document.addEventListener('DOMContentLoaded', function() {**

**loadHome();**

**});**

**/\* Modern CSS Reset \*/**

**\* {**

**margin: 0;**

**padding: 0;**

**box-sizing: border-box;**

**font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;**

**}**

**:root {**

**--primary: #4f46e5; /\* Indigo \*/**

**--primary-dark: #4338ca;**

**--secondary: #10b981; /\* Emerald \*/**

**--accent: #f59e0b; /\* Amber \*/**

**--dark: #1f2937;**

**--light: #f3f4f6;**

**--white: #ffffff;**

**53 of 80**

**--gray: #6b7280;**

**--error: #ef4444;**

**--success: #22c55e;**

**--border-radius: 8px;**

**--box-shadow: 0 4px 6px -1px rgba(0, 0, 0, 0.1), 0 2px 4px -1px rgba(0, 0, 0, 0.06);**

**--transition: all 0.3s ease;**

**}**

**body {**

**background-color: #f9fafb;**

**color: var(--dark);**

**line-height: 1.6;**

**min-height: 100vh;**

**display: flex;**

**flex-direction: column;**

**}**

**/\* Modern Navbar \*/**

**.navbar {**

**background-color: var(--white);**

**box-shadow: 0 2px 4px rgba(0, 0, 0, 0.05);**

**padding: 0.8rem 2rem;**

**display: flex;**

**justify-content: space-between;**

**align-items: center;**

**position: sticky;**

**top: 0;**

**54 0f 80**

**z-index: 100;**

**}**

**.logo {**

**display: flex;**

**align-items: center;**

**gap: 0.5rem;**

**font-size: 1.5rem;**

**font-weight: 700;**

**color: var(--primary);**

**}**

**.logo i {**

**font-size: 1.8rem;**

**}**

**.nav-links {**

**display: flex;**

**gap: 1.5rem;**

**}**

**.nav-item {**

**text-decoration: none;**

**color: var(--gray);**

**font-weight: 500;**

**padding: 0.5rem 0.8rem;**

**border-radius: var(--border-radius);**

**55 of 80**

**transition: var(--transition);**

**display: flex;**

**align-items: center;**

**gap: 0.5rem;**

**}**

**.nav-item i {**

**font-size: 1.1rem;**

**}**

**.nav-item:hover {**

**color: var(--primary);**

**background-color: rgba(79, 70, 229, 0.05);**

**}**

**.nav-item.active {**

**color: var(--primary);**

**background-color: rgba(79, 70, 229, 0.1);**

**}**

**.login-container {**

**display: flex;**

**align-items: center;**

**}**

**.login-btn {**

**display: flex;**

**56 of 80**

**align-items: center;**

**gap: 0.5rem;**

**background-color: var(--primary);**

**color: var(--white);**

**padding: 0.6rem 1.2rem;**

**border-radius: var(--border-radius);**

**text-decoration: none;**

**font-weight: 500;**

**transition: var(--transition);**

**box-shadow: 0 1px 3px rgba(0, 0, 0, 0.1);**

**}**

**.login-btn:hover {**

**background-color: var(--primary-dark);**

**transform: translateY(-1px);**

**box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);**

**}**

**/\* Page Container \*/**

**.page-container {**

**flex: 1;**

**padding: 2rem;**

**max-width: 1200px;**

**margin: 0 auto;**

**width: 100%;**

**}**

**/\* Home Content \*/**

**57 of 80**

**.home-content {**

**margin-bottom: 2rem;**

**}**

**.hero-section {**

**background: linear-gradient(120deg, var(--primary), var(--primary-dark));**

**color: var(--white);**

**padding: 3rem;**

**border-radius: var(--border-radius);**

**margin-bottom: 2rem;**

**text-align: center;**

**box-shadow: var(--box-shadow);**

**}**

**.hero-section h1 {**

**font-size: 2.5rem;**

**margin-bottom: 1rem;**

**font-weight: 800;**

**}**

**.hero-section p {**

**font-size: 1.2rem;**

**margin-bottom: 2rem;**

**max-width: 600px;**

**margin-left: auto;**

**margin-right: auto;**

**}**

**58 of 80**

**.action-buttons {**

**display: flex;**

**gap: 1rem;**

**justify-content: center;**

**}**

**.primary-btn {**

**background-color: var(--white);**

**color: var(--primary);**

**border: none;**

**padding: 0.8rem 1.5rem;**

**border-radius: var(--border-radius);**

**font-weight: 600;**

**cursor: pointer;**

**transition: var(--transition);**

**box-shadow: 0 1px 3px rgba(0, 0, 0, 0.1);**

**}**

**.primary-btn:hover {**

**transform: translateY(-2px);**

**box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);**

**}**

**.secondary-btn {**

**background-color: rgba(255, 255, 255, 0.1);**

**color: var(--white);**

**border: 1px solid rgba(255, 255, 255, 0.3);**

**padding: 0.8rem 1.5rem;**

**59 of 80**

**border-radius: var(--border-radius);**

**font-weight: 600;**

**cursor: pointer;**

**transition: var(--transition);**

**}**

**.secondary-btn:hover {**

**background-color: rgba(255, 255, 255, 0.2);**

**transform: translateY(-2px);**

**}**

**.features {**

**display: grid;**

**grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));**

**gap: 1.5rem;**

**margin-top: 3rem;**

**}**

**.feature-card {**

**background-color: var(--white);**

**padding: 2rem;**

**border-radius: var(--border-radius);**

**box-shadow: var(--box-shadow);**

**text-align: center;**

**transition: var(--transition);**

**}**

**.feature-card:hover {**

**60 of 80**

**transform: translateY(-5px);**

**box-shadow: 0 10px 15px -3px rgba(0, 0, 0, 0.1);**

**}**

**.feature-card i {**

**font-size: 2.5rem;**

**color: var(--primary);**

**margin-bottom: 1rem;**

**}**

**.feature-card h3 {**

**font-size: 1.25rem;**

**margin-bottom: 0.8rem;**

**color: var(--dark);**

**}**

**.feature-card p {**

**color: var(--gray);**

**}**

**/\* Footer \*/**

**.footer {**

**background-color: var(--dark);**

**color: var(--light);**

**padding: 1.5rem 2rem;**

**margin-top: auto;**

**}**

**61 of 80**

**.footer-content {**

**display: flex;**

**justify-content: space-between;**

**align-items: center;**

**max-width: 1200px;**

**margin: 0 auto;**

**}**

**.social-links {**

**display: flex;**

**gap: 1rem;**

**}**

**.social-links a {**

**color: var(--light);**

**font-size: 1.2rem;**

**transition: var(--transition);**

**}**

**.social-links a:hover {**

**color: var(--primary);**

**}**

**/\* Overlay and Popup Styles \*/**

**.overlay {**

**position: fixed;**

**top: 0;**

**62 of 80**

**left: 0;**

**width: 100%;**

**height: 100%;**

**background-color: rgba(0, 0, 0, 0.7);**

**z-index: 998;**

**backdrop-filter: blur(4px);**

**}**

**.popup {**

**position: fixed;**

**top: 50%;**

**left: 50%;**

**transform: translate(-50%, -50%);**

**background-color: var(--white);**

**padding: 2rem;**

**border-radius: var(--border-radius);**

**box-shadow: 0 10px 25px rgba(0, 0, 0, 0.2);**

**z-index: 999;**

**width: 400px;**

**max-width: 90%;**

**}**

**.popup h2 {**

**text-align: center;**

**margin-bottom: 1.5rem;**

**color: var(--primary);**

**font-weight: 700;**

**position: relative;**

**63 of 80**

**}**

**.popup h2:after {**

**content: '';**

**position: absolute;**

**left: 50%;**

**bottom: -8px;**

**transform: translateX(-50%);**

**width: 50px;**

**height: 3px;**

**background-color: var(--primary);**

**border-radius: 2px;**

**}**

**.form-group {**

**margin-bottom: 1.2rem;**

**}**

**.form-group label {**

**display: block;**

**margin-bottom: 0.5rem;**

**font-weight: 500;**

**color: var(--dark);**

**}**

**.form-group input {**

**width: 100%;**

**padding: 0.8rem;**

**64 of 80**

**border: 1px solid #e5e7eb;**

**border-radius: var(--border-radius);**

**transition: var(--transition);**

**}**

**.form-group input:focus {**

**outline: none;**

**border-color: var(--primary);**

**box-shadow: 0 0 0 3px rgba(79, 70, 229, 0.2);**

**}**

**.buttons {**

**display: flex;**

**justify-content: space-between;**

**margin-top: 2rem;**

**}**

**.buttons button {**

**padding: 0.8rem 1.5rem;**

**border: none;**

**border-radius: var(--border-radius);**

**cursor: pointer;**

**font-weight: 600;**

**transition: var(--transition);**

**}**

**.buttons button[type="button"] {**

**background-color: #e5e7eb;**

**65 of 80**

**color: var(--dark);**

**}**

**.buttons button[type="button"]:hover {**

**background-color: #d1d5db;**

**}**

**.buttons button[type="submit"] {**

**background-color: var(--primary);**

**color: var(--white);**

**}**

**.buttons button[type="submit"]:hover {**

**background-color: var(--primary-dark);**

**}**

**/\* Quiz Container Styles \*/**

**.quiz-container {**

**background-color: var(--white);**

**padding: 2rem;**

**border-radius: var(--border-radius);**

**box-shadow: var(--box-shadow);**

**max-width: 800px;**

**margin: 0 auto;**

**}**

**.progress-bar {**

**height: 8px;**

**66 of 80**

**background-color: #e5e7eb;**

**border-radius: 4px;**

**overflow: hidden;**

**margin-bottom: 1rem;**

**}**

**.progress {**

**height: 100%;**

**background: linear-gradient(90deg, var(--primary), var(--secondary));**

**transition: width 0.3s ease;**

**}**

**.question-header {**

**display: flex;**

**justify-content: space-between;**

**align-items: center;**

**margin-bottom: 1.5rem;**

**}**

**.question-counter {**

**background-color: var(--primary);**

**color: var(--white);**

**padding: 0.4rem 0.8rem;**

**border-radius: var(--border-radius);**

**font-size: 0.85rem;**

**font-weight: 600;**

**display: inline-block;**

**}**

**67 of 80**

**.timer {**

**display: flex;**

**align-items: center;**

**gap: 0.5rem;**

**color: var(--dark);**

**font-weight: 500;**

**}**

**.timer i {**

**color: var(--accent);**

**}**

**.quiz-question {**

**margin-bottom: 1.5rem;**

**}**

**.quiz-question p {**

**font-weight: 600;**

**font-size: 1.25rem;**

**margin-bottom: 1.5rem;**

**color: var(--dark);**

**line-height: 1.4;**

**}**

**.quiz-question label {**

**display: block;**

**padding: 1rem;**

**border: 1px solid #e5e7eb;**

**68 of 80**

**border-radius: var(--border-radius);**

**margin-bottom: 0.8rem;**

**cursor: pointer;**

**transition: var(--transition);**

**position: relative;**

**}**

**.quiz-question label:hover {**

**background-color: rgba(79, 70, 229, 0.05);**

**border-color: var(--primary);**

**}**

**.quiz-question input[type="radio"] {**

**margin-right: 0.8rem;**

**}**

**.quiz-question input[type="radio"]:checked + label {**

**background-color: rgba(79, 70, 229, 0.1);**

**border-color: var(--primary);**

**font-weight: 500;**

**}**

**.button-container {**

**display: flex;**

**justify-content: space-between;**

**margin-top: 1.5rem;**

**}**

**69 of 80**

**.button {**

**padding: 0.8rem 1.5rem;**

**border: none;**

**border-radius: var(--border-radius);**

**cursor: pointer;**

**font-weight: 600;**

**transition: var(--transition);**

**display: flex;**

**align-items: center;**

**gap: 0.5rem;**

**}**

**.skip-button {**

**background-color: #e5e7eb;**

**color: var(--dark);**

**}**

**.skip-button:hover {**

**background-color: #d1d5db;**

**}**

**.next-button {**

**background-color: var(--primary);**

**color: var(--white);**

**}**

**.next-button:hover {**

**Background-color: var(--primary-dark);**

**70 of 80**

**}**

**/\* Results Page Styles \*/**

**.result-container {**

**text-align: center;**

**padding: 2rem;**

**}**

**.result-header {**

**margin-bottom: 2rem;**

**}**

**.result-header h2 {**

**font-size: 2rem;**

**margin-bottom: 0.5rem;**

**color: var(--primary);**

**}**

**.user-info {**

**background-color: rgba(79, 70, 229, 0.05);**

**padding: 1.5rem;**

**border-radius: var(--border-radius);**

**margin-bottom: 2rem;**

**text-align: left;**

**border-left: 4px solid var(--primary);**

**}**

**.score-summary {**

**71 of 80**

**display: grid;**

**grid-template-columns: repeat(auto-fit, minmax(180px, 1fr));**

**gap: 1rem;**

**margin-bottom: 2rem;**

**}**

**.score-card {**

**background-color: var(--white);**

**padding: 1.5rem;**

**border-radius: var(--border-radius);**

**box-shadow: var(--box-shadow);**

**}**

**.score-card.total {**

**background: linear-gradient(120deg, var(--primary), var(--primary-dark));**

**color: var(--white);**

**}**

**.score-card h3 {**

**font-size: 2.5rem;**

**margin-bottom: 0.5rem;**

**}**

**.score-card p {**

**font-size: 0.9rem;**

**color: inherit;**

**opacity: 0.9;**

**}**

**72 of 80**

**.chart-container {**

**height: 300px;**

**margin: 2rem 0;**

**padding: 1rem;**

**background-color: var(--white);**

**border-radius: var(--border-radius);**

**box-shadow: var(--box-shadow);**

**}**

**.action-button {**

**background-color: var(--primary);**

**color: var(--white);**

**padding: 0.8rem 1.5rem;**

**border: none;**

**border-radius: var(--border-radius);**

**font-weight: 600;**

**cursor: pointer;**

**transition: var(--transition);**

**display: inline-flex;**

**align-items: center;**

**gap: 0.5rem;**

**margin-top: 1.5rem;**

**}**

**.action-button:hover {**

**background-color: var(--primary-dark);**

**transform: translateY(-2px);**

**box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);**

**73 of 80**

**}**

**/\* Responsive Adjustments \*/**

**@media (max-width: 768px) {**

**.navbar {**

**padding: 0.8rem 1rem;**

**flex-wrap: wrap;**

**}**

**.nav-links {**

**order: 3;**

**width: 100%;**

**justify-content: space-between;**

**margin-top: 0.8rem;**

**overflow-x: auto;**

**padding-bottom: 0.5rem;**

**}**

**.nav-item {**

**white-space: nowrap;**

**}**

**.hero-section {**

**padding: 2rem 1rem;**

**}**

**.action-buttons {**

**flex-direction: column;**

**74 of 80**

**gap: 0.8rem;**

**}**

**.features {**

**grid-template-columns: 1fr;**

**}**

**.footer-content {**

**flex-direction: column;**

**gap: 1rem;**

**text-align: center;**

**}**

**.social-links {**

**justify-content: center;**

**}**

**.popup {**

**width: 90%;**

**padding: 1.5rem;**

**}**

**}**

**@media (max-width: 480px) {**

**.page-container {**

**padding: 1rem;**

**.quiz-container {**

**padding: 1.5rem;**

**75 of 80**

**}**

**.button-container {**

**flex-direction: column;**

**gap: 0.8rem;**

**}**

**.button {**

**width: 100%;**

**justify-content: center;**

**}**

**}**

**76 of 80**

**4.2 Test Cases**

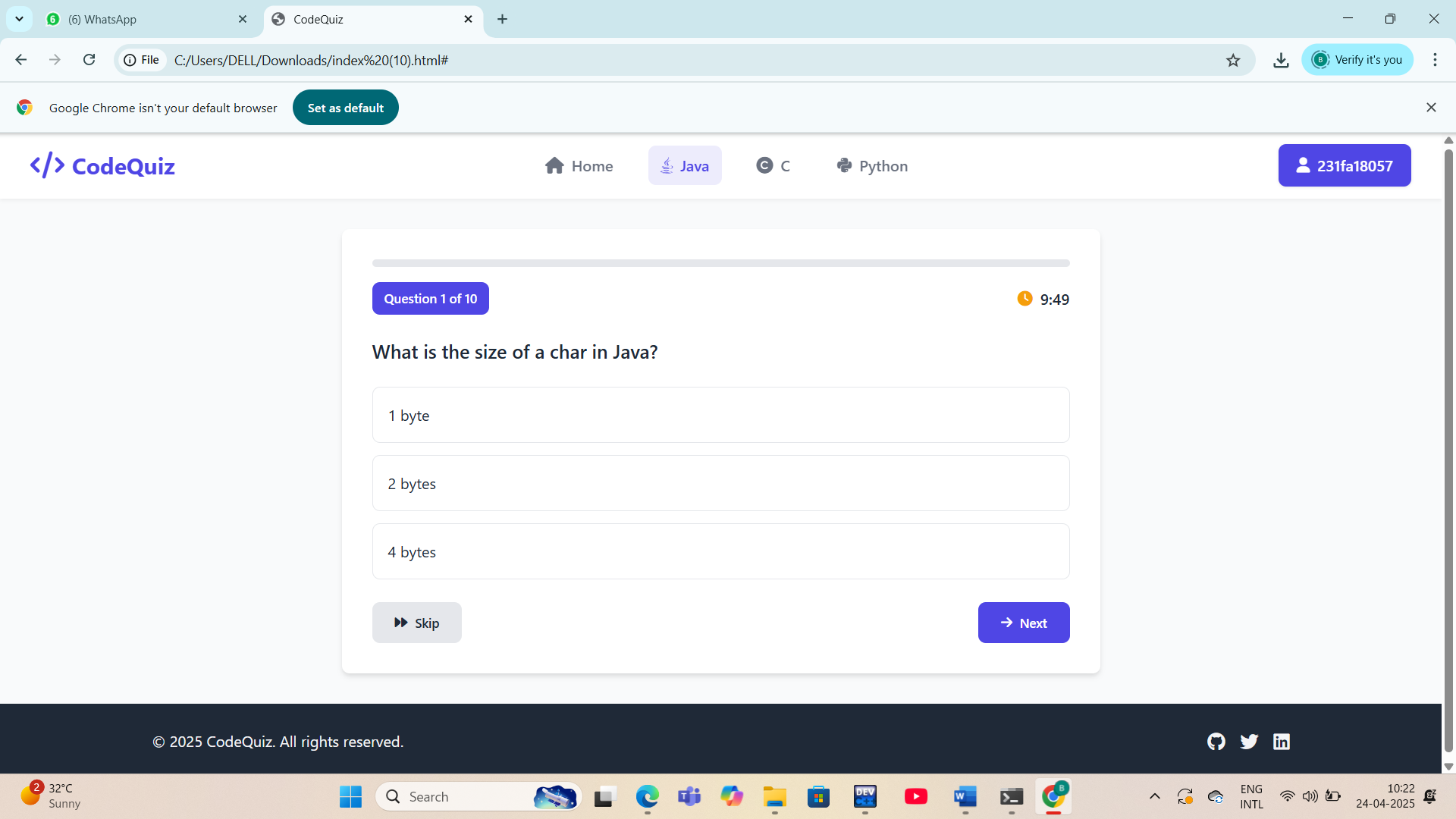
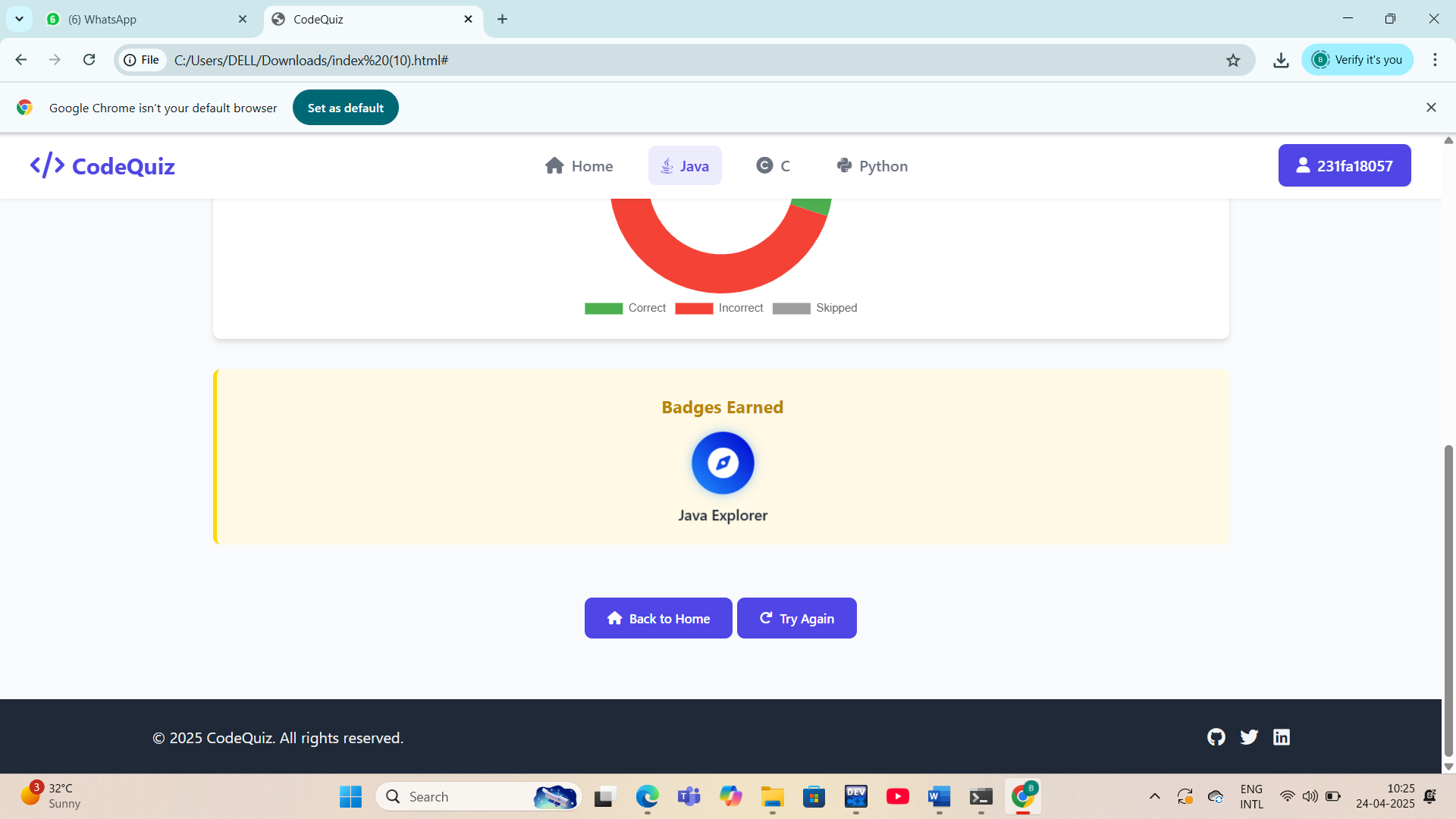
1. **User Registration:**
   * Test if the user can register successfully.
   * Test if the user receives an error message for duplicate usernames.
2. **Invitation Creation:**
   * Test if the user can create a quiz invitation.
   * Test if the invitation contains the correct quiz details.
3. **RSVP Tracking:**
   * Test if the user can RSVP to the quiz invitation.
   * Test if the RSVP status is updated correctly in the system.
4. **Quiz Functionality:**
   * Test if the quiz loads the first question correctly.
   * Test if the user can select an answer and proceed to the next question.
   * Test if the final score is calculated and displayed correctly.

77 of 80

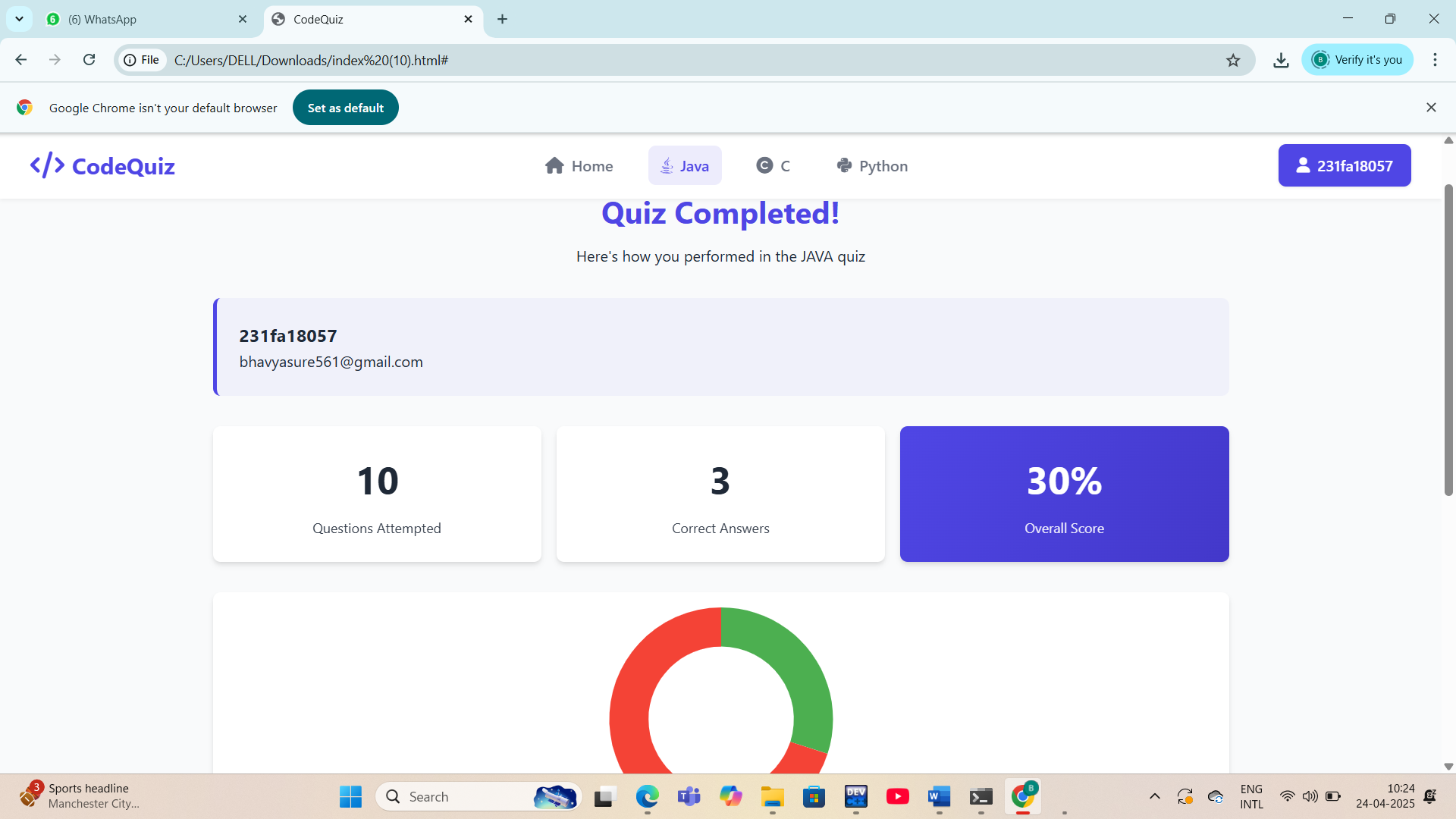
**5 : RESULTS**

**5.1 Output Screens**

Screenshots of the system’s user interface



78 of 80



79 of 80

**6: CONCLUSION**

The Code Quiz project successfully delivers an interactive, user-friendly platform that helps users enhance their programming knowledge through quizzes in Java, C, and Python. With a visually engaging interface, progress tracking, a badge reward system, and intuitive navigation, it provides a motivating environment for learning and self-assessment. Features like a login system, quiz analytics, and personalized feedback make it suitable for both individual learners and educational environments. This project not only demonstrates effective web development skills but also emphasizes user engagement and educational impact.

**REFERENCE**

* **HTML and CSS:** [**W3Schools**](https://www.w3schools.com/)
* **JavaScript:** [**MDN Web Docs**](https://developer.mozilla.org/)
* **Icons:** [**Font Awesome**](https://fontawesome.com/)
* **Charts:** [**Chart.js**](https://www.chartjs.org/)

**80 of 80**