NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

A MINI PROJECT REPORT

on

WEB DESIGN TECHNOLOGIES (24CSE361)

ONLINE QUIZ APPLICATION

Submitted by

NAME:K Sumanth Reddy
USN:1NH24CS098

Under the guidance of
Ms. Divyanshi Chhabra
Assistant Professor

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

Academic Year: 2025-26 (ODD SEM)

NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

CERTIFICATE

This is to certify that the mini project work titled **"Online Quiz Application"** is a bonafide work carried out by K Sumanth Reddy **(1NH24CS098)** in partial fulfillment of the degree of **Bachelor of Engineering** in **Computer Science and Engineering** of the New Horizon College of Engineering during the year **2025-2026.**

Signature of Guide   Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner	*Signature with date*
1.________________________	
2.	

ABSTRACT

The rapid growth of web technologies has significantly transformed the way information is shared, accessed, and evaluated. In the field of education, web-based applications have become an essential tool for learning and self-assessment. Quiz-based systems, in particular, provide an effective method for evaluating knowledge, improving understanding, and encouraging active participation among learners. This mini project focuses on the design and development of an **Interactive Quiz Website** using Web Design Technologies.

The Quiz Website is developed using core front-end technologies such as HTML, CSS, and JavaScript. HTML is used to create the structure and layout of the web pages, CSS is applied to enhance the visual appearance and user interface, and JavaScript is used to add interactivity and dynamic behavior. The application allows users to select a subject from a list of available programming and web technology topics, including HTML, CSS, JavaScript, C, Java, and Python. After selecting a subject, users can attempt a series of multiple-choice questions related to that subject.

The system dynamically loads questions, records user responses, and evaluates the answers in real time. At the end of the quiz, the final score is displayed instantly, enabling users to assess their performance. The application is designed to be simple, user-friendly, and accessible, without requiring any user registration or database connectivity. This makes the system lightweight and easy to use for beginners as well as advanced learners.

The primary objective of this project is to gain practical knowledge of Web Design Technologies and understand how HTML, CSS, and JavaScript work together to build a complete and functional web application. The project also helps in improving logical thinking, problem-solving skills, and hands-on experience in front-end development. Overall, this mini project serves as an effective learning tool and demonstrates the practical implementation of fundamental web development concepts.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, who's constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman, New Horizon Educational Institutions, for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal, New Horizon College of Engineering, for the constant support and encouragement.

I would like to thank **Dr. Anandhi R J**, Professor and Dean-Academics, NHCE, for her valuable guidance.

I would also like to thank **Dr. B. Rajalakshmi,** Professor and HOD, Department of Computer Science and Engineering, for the constant support.

I also express my gratitude to **Ms.Divyanshi Chhabra**, Assistant Professor , Department of Computer Science and Engineering, my mini project reviewer, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

K Sumanth Reddy

1NH24CS098

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

1.1 PROBLEM DEFINITION

In today's academic environment, students increasingly depend on online resources to support learning and self-evaluation. Although several quiz and learning platforms are available, many of them are complex in nature, require mandatory user registration, or depend on backend systems such as databases and servers. These factors make such platforms less suitable for beginners and for quick practice sessions. Students often need a simple and easily accessible application that allows them to test their knowledge without unnecessary complexity.

Another major issue with existing systems is the lack of clear subject-wise organization. Learners usually prefer to practice questions from specific subjects such as HTML, CSS, JavaScript, C, Java, or Python. However, many applications either mix different topics or do not provide smooth navigation between subjects, which results in confusion and reduced learning effectiveness. Additionally, some platforms do not provide immediate feedback, making it difficult for users to analyze their performance and identify areas for improvement.

Traditional classroom learning methods also offer limited opportunities for continuous self-assessment. Without interactive tools, students may find it difficult to evaluate their understanding independently. Hence, there is a need for an interactive and user-friendly quiz system that enables learners to practice multiple-choice questions and receive instant results.

The problem addressed in this mini project is the absence of a lightweight, browser-based quiz application that is easy to use, subject-oriented, and accessible without registration. This project aims to overcome these issues by developing an interactive quiz website using HTML, CSS, and JavaScript, focusing on simplicity, usability, and effective knowledge assessment.

1.2.OBJECTIVE

The main objective of this mini project is to design and develop an interactive quiz website using Web Design Technologies that helps students evaluate their knowledge in an easy and effective manner. The project aims to provide a simple, user-friendly platform that enables learners to practice multiple-choice questions related to various programming and web technology subjects.

Another important objective of this project is to gain practical exposure to front-end web development by implementing HTML for structuring the web pages, CSS for designing and enhancing the visual appearance, and JavaScript for adding interactivity and dynamic behaviour.

The project also aims to demonstrate how these technologies can be integrated to build a complete and functional web application.

The project further focuses on allowing users to select subjects of their choice, navigate through quiz questions smoothly, and receive instant feedback in the form of scores. By developing this application, the objective is also to improve logical thinking, problem-solving skills, and understanding of client-side scripting concepts. Overall, this mini project aims to strengthen the practical knowledge of Web Design Technologies and their real-world applications.

1.3.METHODOLOGIES TO BE FOLLOWED

1. Requirement Analysis

• Collected requirements such as subject selection, question display, multiple-choice options, navigation buttons, score calculation, and result display.

• Understood the target users—students preparing for exams, interviews, and self-assessment.

• Identified functional requirements such as quiz flow, answer validation, and result generation, and non-functional requirements such as usability, responsiveness, and performance.

2. Planning and Design

• Planned the overall layout of the quiz website including header, subject selection section, quiz area, and result section.

• Designed the flow of the application from subject selection to quiz completion.

• Selected a simple color scheme, readable fonts, and clean design for better user experience.

• Planned a responsive layout to ensure proper display on mobile, tablet, and desktop devices.

3. Front-End Development

• Implemented the basic structure of the website using HTML.

• Designed the visual appearance using CSS, including grids, buttons, spacing, colors, and hover effects.

• Added interactive functionality using JavaScript such as Subject selection, Dynamic loading of quiz questions, Option selection and highlighting, Navigation between questions, Score calculation and result display.

• Ensured responsiveness using flexible layouts and CSS styling.

4. Content Integration

• Added quiz questions, options, and correct answers for each subject such as HTML, CSS, JavaScript, C, Java, and Python.

• Organized questions subject-wise for easy navigation and clarity.

• Ensured that the content is well-structured, readable, and relevant to the quiz objectives.

5. Testing

• Tested the quiz website for Functionality (subject selection, question navigation, score calculation), Responsiveness on mobile, tablet, and desktop devices, Performance and smooth interaction, Browser compatibility.

• Checked for logical errors, incorrect scoring, and UI issues.

6. Optimization

• Optimized the code by properly separating HTML, CSS, and JavaScript files.

• Improved loading performance by using efficient coding practices.

• Ensured good user experience with clear buttons, readable text, and consistent design.

7. Deployment and Maintenance

• Prepared the quiz website files for execution in a browser environment.

• Tested the application in a live browser setup.

CHAPTER 2

FUNDAMENTALS OF THE LANGUAGES USED

2.1 HTML

HTML is the foundational markup language used to create and structure web pages on the World Wide Web. It defines the layout of a webpage by using elements enclosed within tags such as <h1>, <p>, and <div>. HTML was originally developed by Tim Berners-Lee in 1989 as part of the World Wide Web project at CERN, helping researchers share documents through hyperlinked text. The earliest version of HTML, introduced in 1991, contained just 18 basic tags.

HTML has evolved significantly over time. In 1995, HTML 2.0 became the first standardized version and introduced essential features such as forms and tables. HTML 3.2 (1997) expanded support for styling through CSS, while HTML 4.01 (1999) added scripting support and improved structural markup. In the early 2000s, XHTML attempted to bring stricter XML rules to web development but faced adoption challenges. The release of HTML5 in 2014 transformed modern web development by introducing semantic tags, built-in multimedia support, canvas elements, and improved cross-platform compatibility. Today, HTML remains the backbone of every website on the internet.

Key Features:

- Structure: Organizes content using headings, paragraphs, lists, images, and sections.
- Hyperlinking: Allows navigation across documents using <a> tags.
- Forms: Collects user input through text fields, buttons, checkboxes, and other controls.
- Semantic Elements: Tags like <header>, <article>, <section>, and <footer> improve clarity and accessibility.
- Media Support: Embeds audio, video, and graphics without external plugins.

2.2 HTML TAGS

TAG	DESCRIPTION
<!DOCTYPE HTML>	Declares that the document is an HTML5 file.
<HTML>	The root element of the webpage; all content is written inside it.
<HEAD>	Contains metadata, page title, and links to CSS and JS files.
<META>	Defines metadata such as character set and viewport settings.
<TITLE>	Sets the title of the webpage (shown on browser tab).
<LINK>	Links external CSS file (style.css) to the HTML document.
<BODY>	Contains the visible content of the webpage.
<HEADER>	Defines the top section of the page, usually for titles and introductions.
<H1>	Defines the main heading (largest heading).
<H2>, <H3>	Define subheadings and smaller titles for sections or cards.
<P>	Defines a paragraph of text.
<NAV>	Represents the navigation section (menu bar).
<UL>	Defines an unordered list (used for navigation menu).
<LI>	Defines list items inside the unordered list.
<A>	Defines a hyperlink (used for navigation links).
<SECTION>	Groups related content together (like Home, Events, Gallery, Contact).
<DIV>	Generic container used for grouping elements and styling with CSS.
<IMG>	Embeds an image in the webpage.
<FORM>	Defines an input form for user data (like contact form).
<LABEL>	Describes an input field.
<INPUT>	Creates a textbox or email box for user input.

<TEXTAREA>	Creates a larger text box for typing messages.
<BUTTON>	Creates a clickable button (for submitting forms).
<FOOTER>	Defines the footer section at the bottom of the page.
<SCRIPT>	Links or embeds JavaScript code (here it connects to script.js).

2.3.CSS

CSS is the stylesheet language used to control the presentation, layout, and visual appearance of HTML documents. While HTML defines the structure of a webpage, CSS determines how that structure looks—its colors, fonts, alignment, spacing, borders, animations, and overall design. CSS was first introduced by Håkon Wium Lie in 1994 while working with Tim Berners-Lee at CERN. The goal was to separate content from design and allow designers more control over webpage styling.

The first official specification, CSS1, was released in 1996, providing basic features such as fonts, colors, and text formatting. CSS2, introduced in 1998, expanded capabilities by adding positioning, zindex, media types, and improved layout control. With the growth of modern, dynamic websites, CSS evolved into CSS3, which was released in modules starting from 2011. CSS3 introduced major advancements such as transitions, animations, shadows, flexbox, grid layout, and responsive design features—greatly improving how modern websites look and behave. Today, CSS is essential for creating responsive, interactive, and visually rich web experiences.

Key Features:

- Styling and Formatting: Controls colors, backgrounds, borders, fonts, and spacing.
- Layout Management: Uses flexbox, grid, floats, and positioning to arrange page elements.
- Responsive Design: Adapts webpages to different screen sizes using media queries.
- Reusability: A single stylesheet can style multiple webpages, reducing duplication.
- Animations and Effects: Provides transitions, keyframe animations, shadows, and transformations.

• **CSS PROPERTY**	**DESCRIPTION**
BACKGROUND	Sets background color, gradient, or image.
BACKGROUND-COLOR	Applies solid background color.
BACKGROUND-IMAGE	Adds image or gradient backgrounds.
COLOR	Sets text color.
FONT-FAMILY	Sets the font style for text.

FONT-SIZE	Sets the size of text.
FONT-WEIGHT	Defines text thickness (bold, normal).
MARGIN	Creates space outside an element.
PADDING	Creates space inside an element.
BORDER	Adds border around elements.
BORDER-RADIUS	Rounds the corners of elements.
BOX-SHADOW	Adds shadow effect around cards and buttons.
WIDTH	Sets the width of an element.
HEIGHT	Sets the height of an element.
DISPLAY	Controls how an element behaves (block, flex, grid, inline).
FLEX	Allows flexible layout inside flex containers.
GAP	Controls spacing between flex or grid items.
JUSTIFY-CONTENT	Aligns items horizontally inside flex containers.
ALIGN-ITEMS	Aligns items vertically in flex containers.
GRID-TEMPLATE-COLUMNS	Defines column structure in grid layouts.
OBJECT-FIT	Controls how images adjust within containers.
CURSOR	Changes the cursor icon (e.g., pointer).
POSITION	Defines element positioning (relative, absolute, fixed).
TOP, LEFT, RIGHT, BOTTOM	Positions elements with position property.
Z-INDEX	Controls stacking order of elements.
OVERFLOW	Handles content overflow (hidden, scroll).
TEXT-ALIGN	Aligns text (left, center, right).
LIST-STYLE	Removes or modifies list bullets.

TRANSITION	Adds smooth animation effects.
OPACITY	Controls transparency of elements.
FILTER	Applies visual effects (blur, brightness).
LINE-HEIGHT	Sets spacing between lines of text.
MAX-WIDTH	Sets maximum width of elements.
MIN-WIDTH	Sets minimum width.
BOX-SIZING	Defines how width and height are calculated.

2.3 JAVASCRIPT

JavaScript is a high-level, lightweight, and versatile programming language primarily usedto create interactive and dynamic web applications. It enables developers to add featuressuch as animations, form validations, and real-time content updates. Initially developed byBrendan Eich at Netscape in 1995, it was originally called Mocha, later renamedLiveScript, and eventually JavaScript to align with the popularity of Java at the time.JavaScript gained widespread adoption with the introduction of ECMAScript standards in1997, which ensured consistency across implementations. Over the years, JavaScript hasevolved significantly, adding modern features like asynchronous programming,modularization, and extensive libraries, making it a cornerstone of modern web development alongside HTML and CSS.

Key Features and examples:

- **Basics:** Understanding variables (var, let, const), data types, operators, and expressions.
- **Functions:** Writing reusable blocks of code to perform specific tasks.
- **DOM Manipulation:** Selecting and modifying elements dynamically.
- **Event Handling:** Responding to user actions like clicks and keypresses.
- **Form Validation:** Ensuring proper user input.

CHAPTER 3

REQUIREMENT SPECIFICATION

SOFTWARE REQUIREMENTS:

1. Operating System

- Windows 10 / Windows 11
- macOS
- Linux (Ubuntu recommended)

Any OS that supports a web browser and code editor can run the project.

2. Web Browser

A modern browser is required to view and test the website:

- Google Chrome (recommended)
- Mozilla Firefox
- Microsoft Edge
- Safari

These browsers support HTML5, CSS3, and JavaScript required for the website.

3. Code Editor / IDE

To create and edit the website files:

- Visual Studio Code (recommended)
- Sublime Text
- Notepad++
- Atom
- Brackets
- Basic Notepad (for simple editing)

4. Languages/Technologies Used

- HTML5 – for creating the structure and layout of the website
- CSS3 – for styling and design
- JavaScript (ES6) – for interactivity and dynamic features

5. Optional Tools (for better workflow)

- Live Server Extension (VS Code) for real-time preview
- Git & GitHub for version control
- Browser DevTools (Inspect Element) for debugging CSS/JS
- Image optimization tools (e.g., TinyPNG)

6. Runtime Environment

- No special environment required.
- The project runs directly in the browser without installation.

7. Additional Libraries (If needed)

- Google Fonts (for typography)
- Icons or graphic resources (Freepik/Icons8)

CHAPTER 4

DESIGN

4.1 DESIGN GOALS

The main design goals of the **Interactive Quiz Website** focus on creating a platform that is simple, user-friendly, visually appealing, and functionally efficient. The objective is to provide students and learners with a smooth and engaging experience while attempting subject-wise quizzes and viewing results clearly.

The primary design goal of the Interactive Quiz Website is to create a simple, user-friendly, and efficient platform for knowledge assessment. The website is designed to allow users to select subjects easily, answer multiple-choice questions, and view results without any complexity.

The design focuses on providing a clean and organized layout so that users can navigate through the quiz smoothly. The interface is kept minimal to avoid distractions and to ensure that the user's attention remains on the quiz content. Consistent colors, readable fonts, and clear buttons are used to enhance usability and visual clarity.

Another important design goal is responsiveness. The website is designed to work effectively on different devices such as desktops, laptops, tablets, and mobile phones. Responsive layouts ensure a consistent experience across all screen sizes.

Performance and efficiency are also key design goals. The website should load quickly and respond instantly to user actions such as selecting answers or navigating between questions. JavaScript is used to handle dynamic content without page reloads, improving user experience.

Finally, the design aims to maintain a modular and maintainable structure. The separation of HTML, CSS, and JavaScript ensures that the code is easy to understand, modify, and extend in the future by adding new subjects or questions.

CHAPTER 5

IMPLEMENTATION

The implementation of the **Interactive Quiz Website** follows a structured and systematic approach, combining HTML, CSS, and JavaScript to create a complete and interactive web-based application. The implementation is carried out in multiple stages to ensure clarity, usability, performance, and a smooth quiz experience for users.

1. HTML Structure Development

The overall layout of the quiz website is developed using semantic HTML5 tags such as <header>, <main>, <section>, and <footer> to improve structure and readability.
• Unique IDs and class names (such as subject buttons, quiz section, question container, options container, and result box) are assigned so that JavaScript can dynamically update content during the quiz.
• Button elements are used for subject selection, option selection, navigation (Next and Submit), and result actions.
• Proper heading and paragraph tags are used to clearly display questions, instructions, and scores.

2. CSS Styling and Page Layout

• CSS is used to design a clean and visually appealing interface for the quiz website.
• Flexbox and CSS Grid are applied to align subject buttons, quiz containers, and option lists in a responsive manner.
• Consistent color schemes, fonts, spacing, and button styles are used across all pages to maintain a professional appearance.
• Hover effects and selected option highlights improve interactivity and user feedback.
• Media queries are used where required to ensure proper display on mobile, tablet, and desktop devices.

3. JavaScript-Based Dynamic Rendering

• Quiz questions for each subject are stored in JavaScript arrays, making the system modular and easy to update.
• JavaScript functions dynamically load questions, options, and question numbers using DOM manipulation.
• Event listeners handle subject selection, option selection, and navigation between questions.
• The quiz content is updated without page reloads, providing a smooth user experience.

4. Question Navigation and Answer Validation

• The quiz allows users to navigate through questions using “Next” and “Submit” buttons.
• Selected answers are validated by comparing user choices with predefined correct answers.
• The system ensures accurate tracking of the current question index and selected options.

5. Score Calculation and Result Display

• JavaScript logic is used to calculate the score based on correct answers.
• After quiz completion, the final score is displayed dynamically in the result section.
• Users are provided options to restart the quiz or return to the subject selection page

6. Interactive User Feedback

• Visual feedback is provided by highlighting selected options.
• Buttons and transitions enhance user interaction and make the quiz more engaging.
• Clear messages and score display help users understand their performance.

7. Smooth User Experience

• The quiz interface updates dynamically without refreshing the page.
• Visibility of sections such as subject selection, quiz area, and result box is controlled using CSS classes and JavaScript.
• This ensures a seamless and uninterrupted quiz flow.

8. Performance Optimization

• Lightweight HTML and CSS ensure fast page loading.
• JavaScript code is optimized to handle only required operations during quiz execution.
• No external libraries or heavy resources are used, improving performance.

9. Responsive & Cross-Device Testing

• The quiz website is tested on multiple screen sizes to ensure responsiveness.
• It is verified to work correctly on major browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge.

10. Code Organization & Maintainability

• HTML, CSS, and JavaScript are maintained in separate files to ensure clean code structure.
• JavaScript functions are modular and reusable, making it easy to add new subjects or questions.
• Logical grouping of variables and functions improves readability and future scalability.

CHAPTER 6

RESULTS
WEBSITE SNAPSHOTS

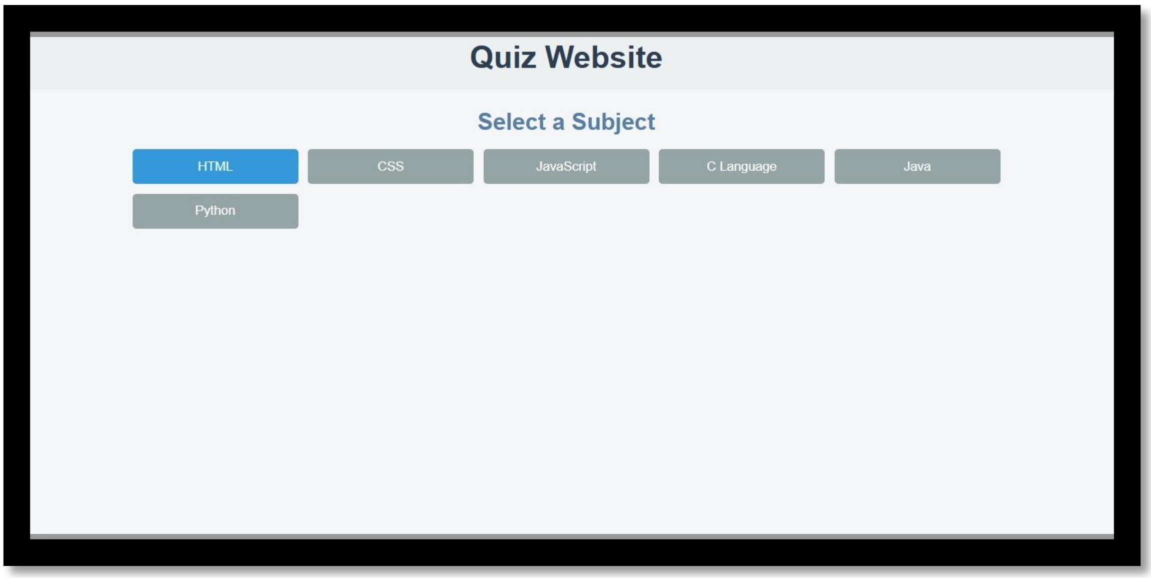
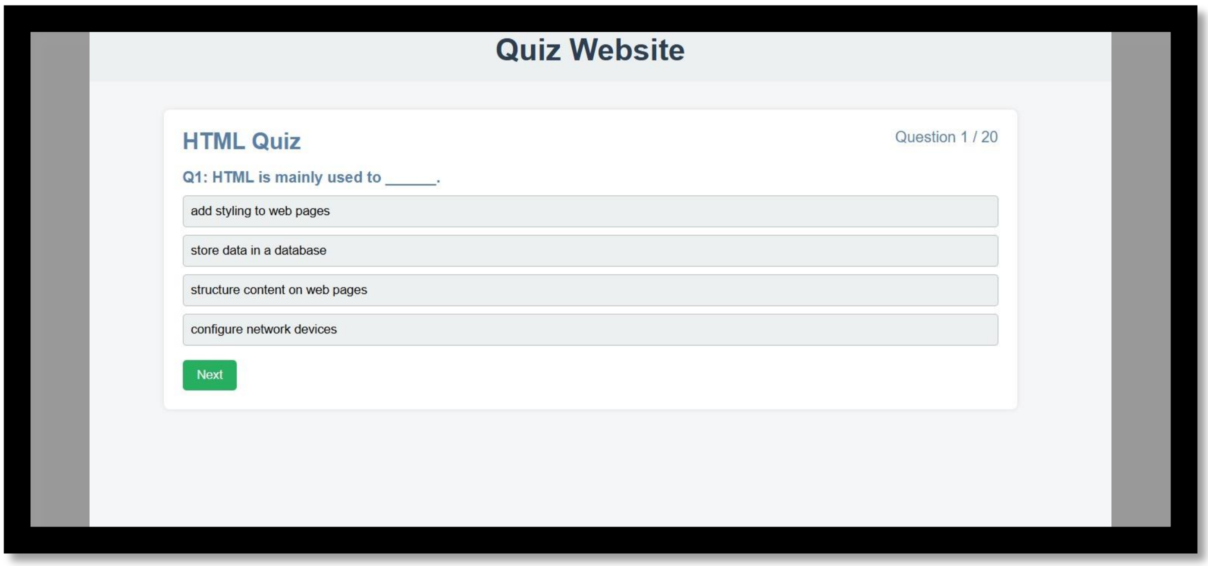


Fig.NO.6.1 HOME PAGE



HTML Quiz

Question 1 / 20

Q1: HTML is mainly used to ______.

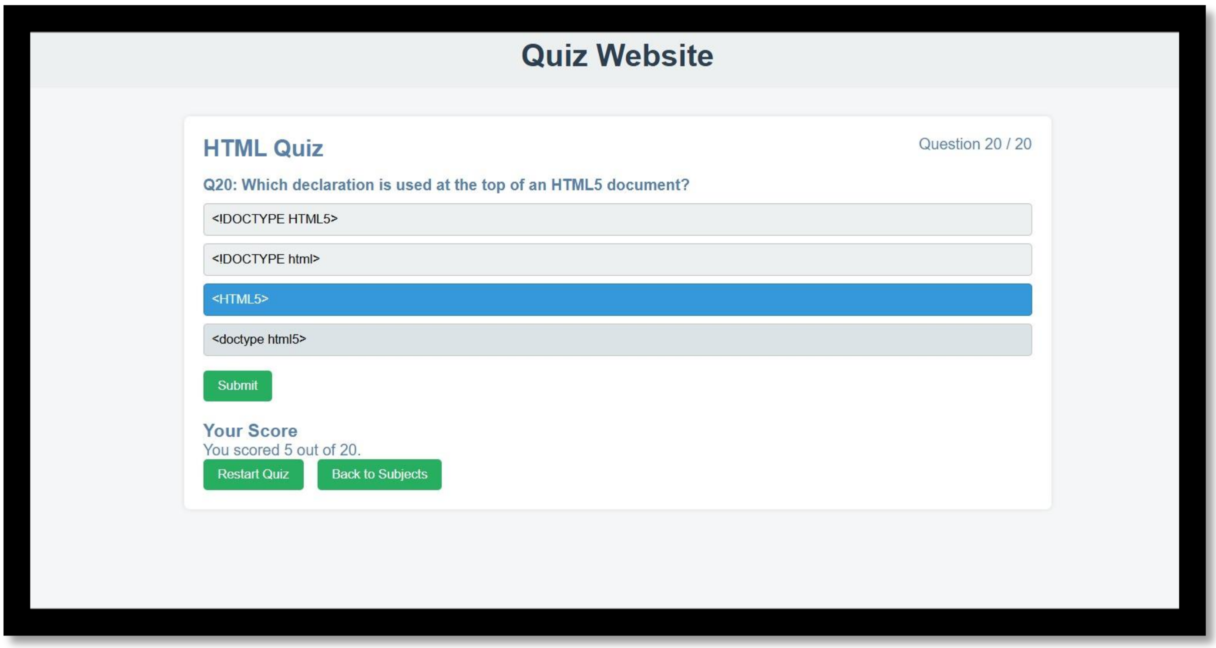
add styling to web pages

store data in a database

structure content on web pages

configure network devices

Fig.NO.6.2 SELECTED QUIZ PAGE



Quiz Website

HTML Quiz

Question 20 / 20

Q20: Which declaration is used at the top of an HTML5 document?

<!DOCTYPE HTML5>

<!DOCTYPE html>

<HTML5>

<doctype html5>

Submit

Your Score

You scored 5 out of 20.

Restart Quiz

Back to Subjects

Fig.NO.6.3 QUIZ RESULT/END PAGE

CHAPTER 7

CONCLUSION

The Interactive Quiz Website mini project successfully demonstrates the practical application of Web Design Technologies using HTML, CSS, and JavaScript. The project achieves its objective of providing a simple, user-friendly, and interactive platform for knowledge assessment. By allowing users to select subjects, answer multiple-choice questions, and view results instantly, the system enhances self-learning and evaluation.

The use of HTML provides a well-structured layout, CSS ensures an attractive and responsive user interface, and JavaScript adds dynamic behavior such as question loading, answer validation, and score calculation. The integration of these technologies highlights the importance of front-end development concepts and their real-world implementation.

This project also helped in improving practical skills such as logical thinking, problem-solving, and client-side scripting. The modular structure of the code makes the application easy to maintain and extend in the future by adding new subjects or questions. Overall, the Interactive Quiz Website serves as an effective learning tool and provides a strong foundation for developing more advanced web-based applications.

REFERENCES

1. MDN Web Docs, *HTML Documentation*.

   Available at: https://developer.mozilla.org/en-US/docs/Web/HTML

2. MDN Web Docs, *CSS Documentation*.

   Available at: https://developer.mozilla.org/en-US/docs/Web/CSS

3. MDN Web Docs, *JavaScript Documentation*.

   Available at: https://developer.mozilla.org/en-US/docs/Web/JavaScript

4. W3Schools, *HTML, CSS and JavaScript Tutorials*.

   Available at: https://www.w3schools.com

5. Ecma International, *ECMAScript® Language Specification (ES6)*.

   Available at: https://www.ecma-international.org

6. Google Developers, *Web Fundamentals*.

   Available at: https://developers.google.com/web