**IMPLEMENTATION**

**Repository Setup on GitHub**

* The GitHub repository serves as a centralized version control system for the project. All code, documentation, and resources related to the project will be stored here. This allows for easy collaboration, tracking changes, and versioning.
* Steps:
  + Create GitHub Repository: Set up a new repository on GitHub. Choose a meaningful name, such as “interactive-quiz-web-app”.
  + Initialize Git: Inside the project directory, run the following command to initialize Git:
* git init
  + Add Remote: Link the local Git repository with the remote GitHub repository:

git remote add origin <repository\_url>

* + Commit and Push: Once the initial setup and files are in place, commit the changes and push to GitHub:
* git add .
* git commit -m "Initial commit"
* git push -u origin main
  + Version Control: As development progresses, version control will be managed by Git. Each time a new feature or module is added or modified, commit the changes with descriptive commit messages.

2. Website Modules Development

* The website is developed in modules, each serving a distinct function. Each module will be developed and tracked separately through Git to ensure scalability and maintainability.

3. Homepage Module

* Purpose: The homepage will act as the landing page of the website and give users a brief overview of the interactive quiz platform.
* Features:
  + Welcoming message or introduction to the platform.
  + Quick navigation to other modules such as quizzes, about, and contact.
  + Display upcoming quizzes or recent activity.
* Technologies: HTML, CSS (for styling), JavaScript (for interactive elements).
* Git Workflow:
  + Create a new branch: git checkout -b homepage.
  + Develop the HTML structure, add CSS for styling, and use JavaScript for dynamic elements like sliders or quiz previews.
  + Commit and push changes:
* git add .
* git commit -m "Added homepage structure and initial content"
* git push origin homepage
  + - * About Page Module
* Purpose: Provides details about the quiz platform, its mission, features, and how users can benefit.
* Features:
  + A brief history of the platform.
  + Information on how the quiz engine works.
  + Any team members or contributors.
* Technologies: HTML, CSS, and potentially JavaScript if there are any animations or dynamic sections.
* Git Workflow:
  + Create a new branch for the about page development: git checkout -b about-page.
  + Develop the page content, structure, and style.
  + Commit and push:
* git add .
* git commit -m "Completed about page content and design"
* git push origin about-page
  + - * Contact Form Module
* Purpose: Provides a way for users to get in touch with the quiz platform administrators or support team.
* Features:
  + Form fields for name, email, message, etc.
  + Submit button that triggers an email or a message via the back-end.
  + Client-side validation using JavaScript.
* Technologies: HTML, CSS (for form styling), JavaScript (for form validation), and possibly a back-end technology like PHP or Node.js to handle form submissions.
* Git Workflow:
  + Create a branch for the contact form: git checkout -b contact-form.
  + Develop the form with all necessary validation and submission mechanisms.
  + Commit and push:
* git add .
* git commit -m "Added contact form and validation"
* git push origin contact-form
  + - * Authentication Module
* Purpose: This module will handle user authentication, enabling users to register, log in, and manage their profiles.
* Features:
  + Registration form (username, email, password).
  + Login form (username/email and password).
  + Password encryption and session management.
  + User profile management (view/update profile information).
* Technologies: HTML, CSS, JavaScript, back-end technologies like Node.js or PHP, and databases (e.g., MySQL, MongoDB).
* Git Workflow:
  + Create a new branch for authentication: git checkout -b authentication.
  + Develop the front-end for registration and login pages.
  + Implement back-end logic for user registration, login, password hashing, and session management.
  + Commit and push:

git add .

* git commit -m "Implemented authentication system"
* git push origin authentication
  + - Version Control Management Using Git
* Branching: Each module (homepage, about, contact form, and authentication) is developed in its own branch. This ensures that work on different modules does not interfere with each other.
* Pull Requests: After finishing development on each module, a pull request (PR) is created to merge the feature branch into the main branch. The code is reviewed, and changes are discussed before merging.
* Commit Messages: Each commit should have a clear and descriptive message to track changes. For example:
  + "Fix bug in homepage slider"
  + "Add validation to contact form"
  + "Refactor authentication logic"
* Merging: After code reviews and testing, branches are merged back into the main branch.

Interactive Quiz Features

* The interactive quiz-making project could include the following functionality:
* Creating Quizzes: Admins can create quizzes with multiple-choice or open-ended questions.
* Taking Quizzes: Users can take quizzes, with immediate feedback on their answers.
* Tracking Scores: Users can see their scores after completing the quiz.
* Leader board: Display top scorers and rankings.
* Each feature will be developed as part of the core app, likely in separate modules or sections. This can also be version-controlled via Git.