Effective Integration of Generative AI into programming learning environments

A Project Report for the Software Engineering (Milestone 2)

Submitted By:

Himanshu Choudhary 22f1001119@ds.study.iitm.ac.in

- **IITM Online BS Degree Program,**
- **Indian Institute of Technology, Madras, Chennai Tamil Nadu, India, 600036**

Storyboard

A GRAPHICAL APPROACH TO DESIGNING THE USER INTERFACE OF OUR APPLICATION.

JOHN'S DILEMMA

Excited to learn, John opens his laptop and gets ready to dive into his studies.

John starts working on his programming assignments, focusing on writing if-else statements

Frustration sets in as John encounters errors in his code and realizes some test cases are missing.

John notices a new button on his screen: 'Take help from AI.' Could this be the solution?

An AI assistant appears, pointing out the issues in John's code and suggesting fixes.

With the AI's help, John corrects his code, passes all test cases, and feels a sense of accomplishment.

FULL STORYBOARD

Excited to learn, John opens his laptop and gets ready to dive into his studies.

John starts working on his programming assignments, focusing on writing if-else statements.

Frustration sets in as John encounters errors in his code and realizes some test cases are missing.

John notices a new button on his screen: 'Take help from AI.' Could this be the solution?

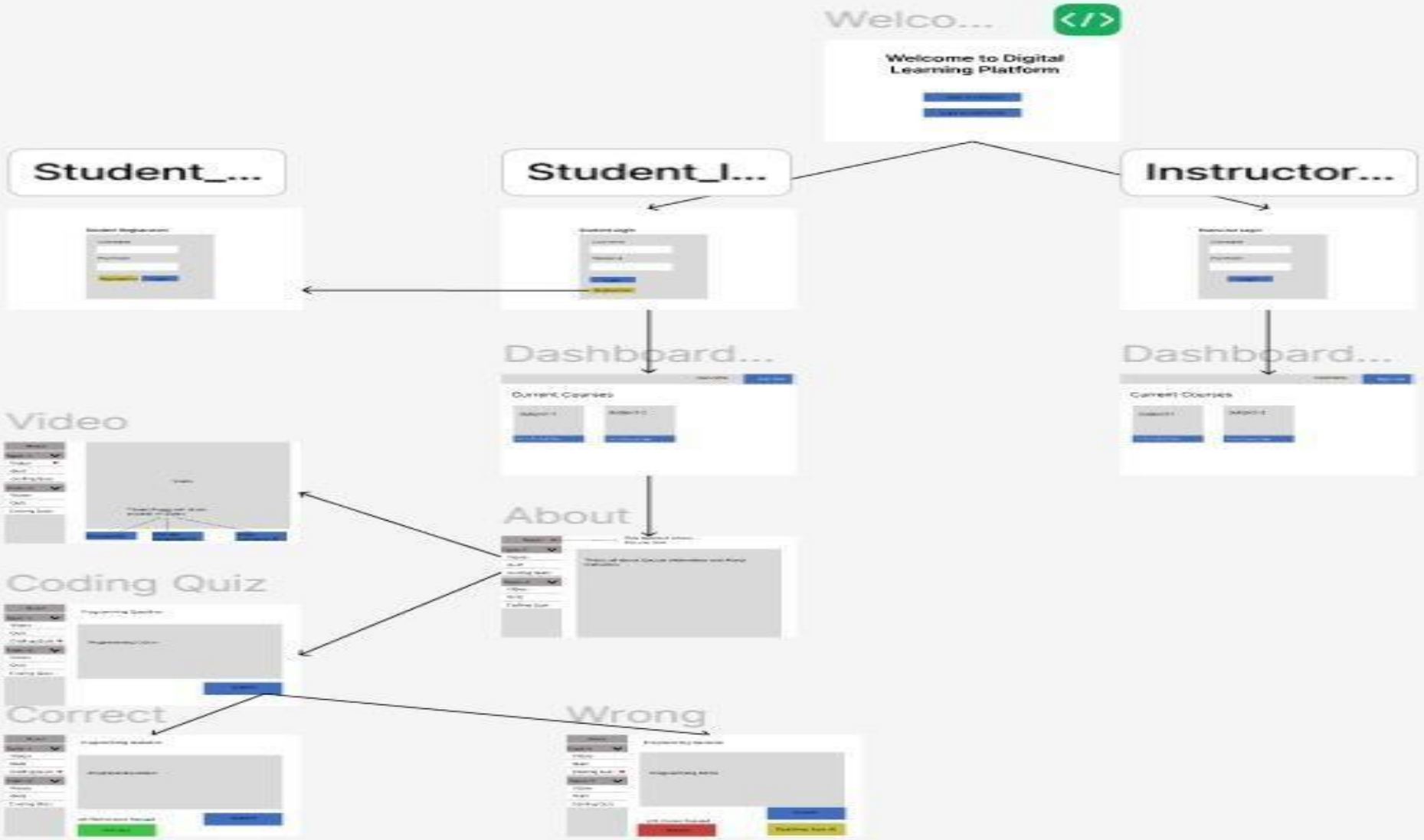
An AI assistant appears, pointing out the issues in John's code and suggesting fixes.

With the AI's help, John corrects his code, passes all test cases, and feels a sense of accomplishment.

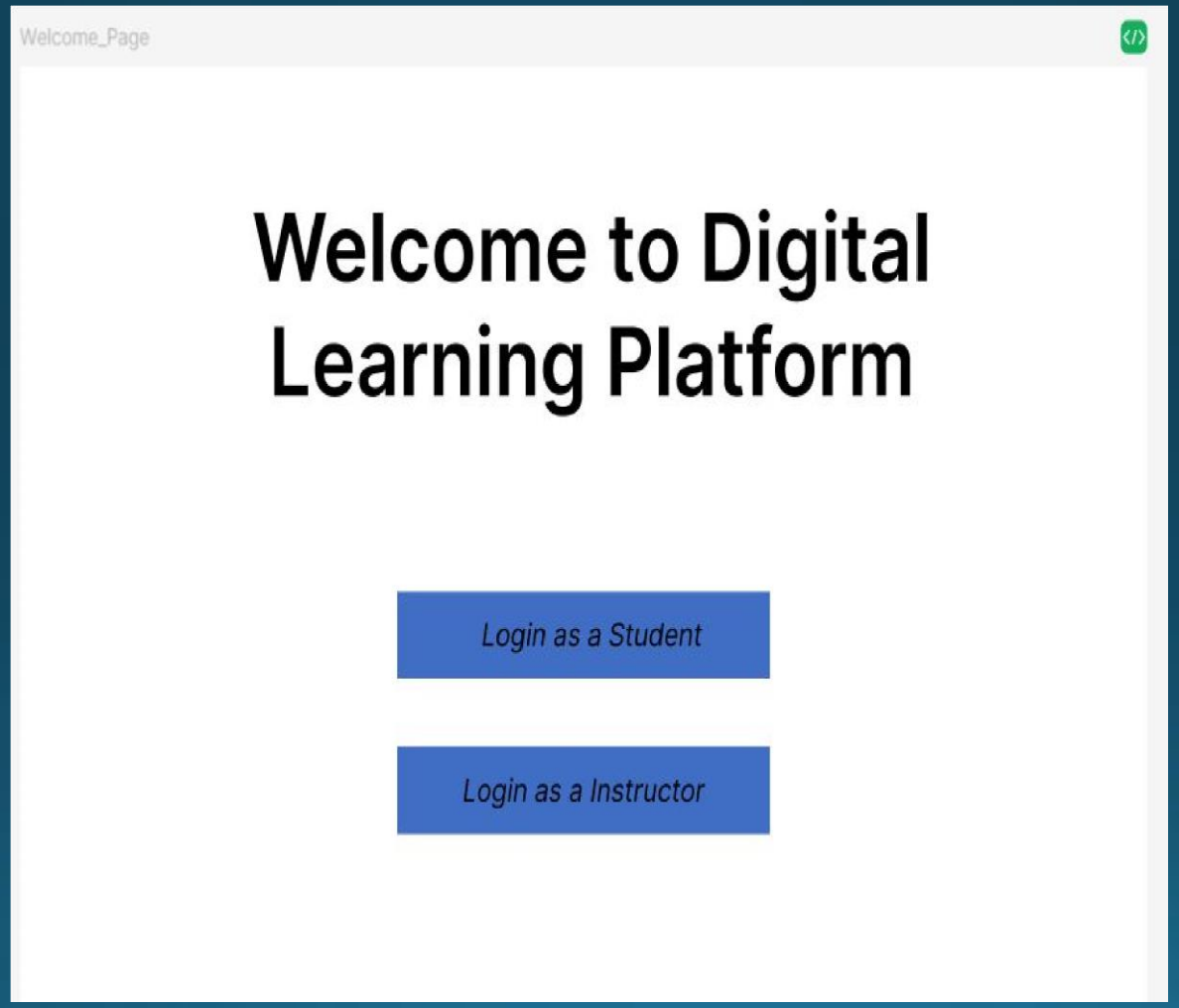
Wireframe

A DIGITAL PROTOTYPE TO MAP OUT THE USER INTERFACE OF OUR APPLICATION

Overview Of Wireframe

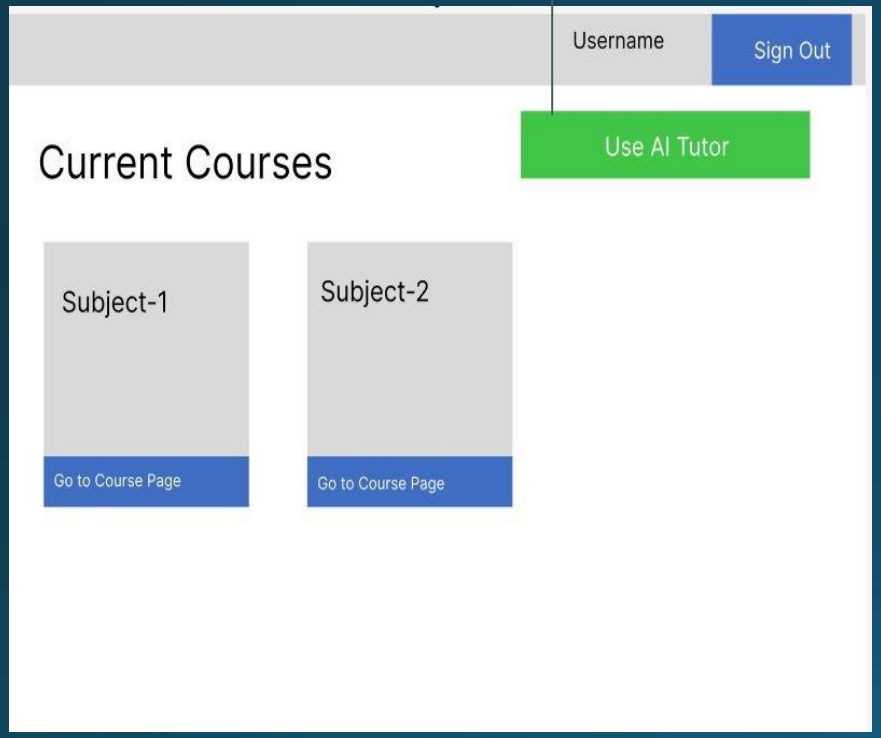


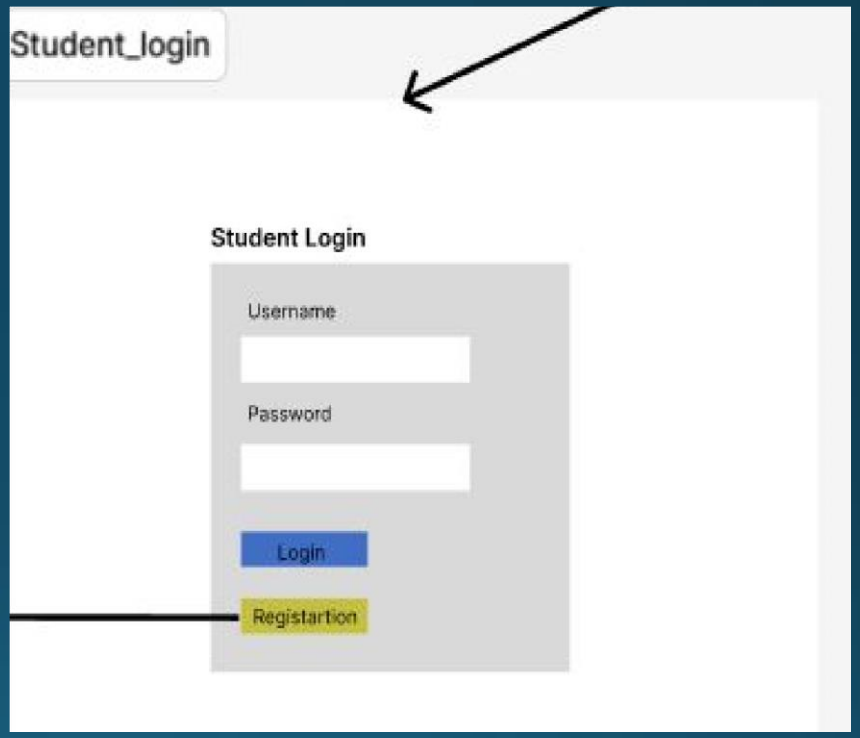
Wireframe: Welcome Page



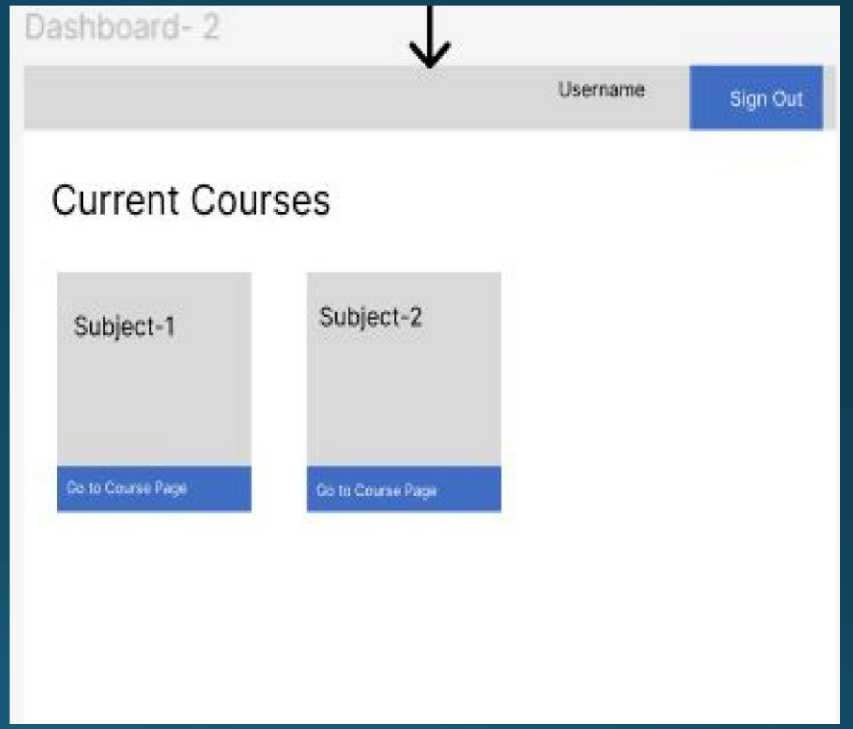
Wireframe: Student Login & Dashboard

Tutor is like ChatGPT

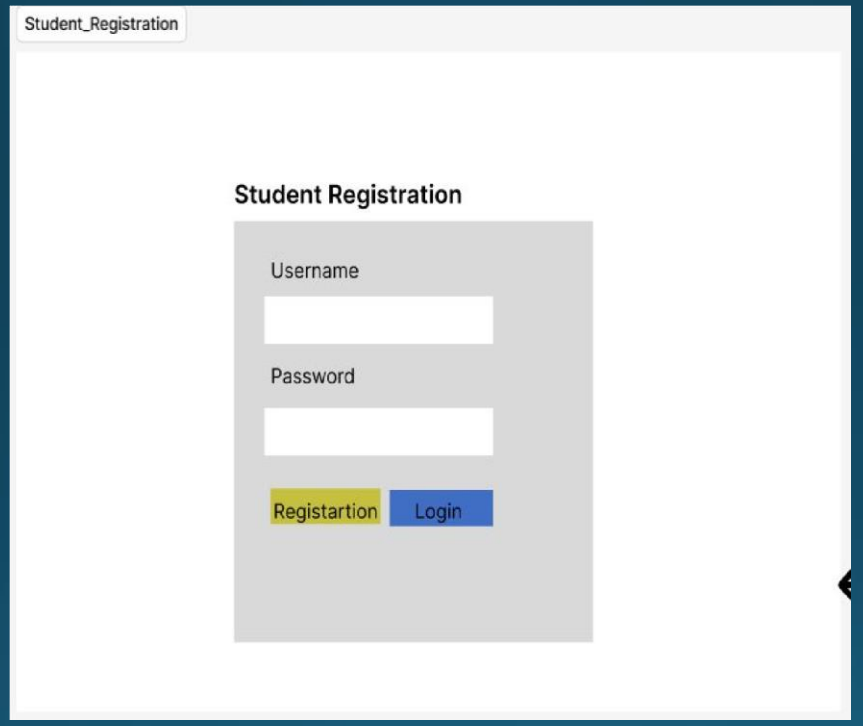




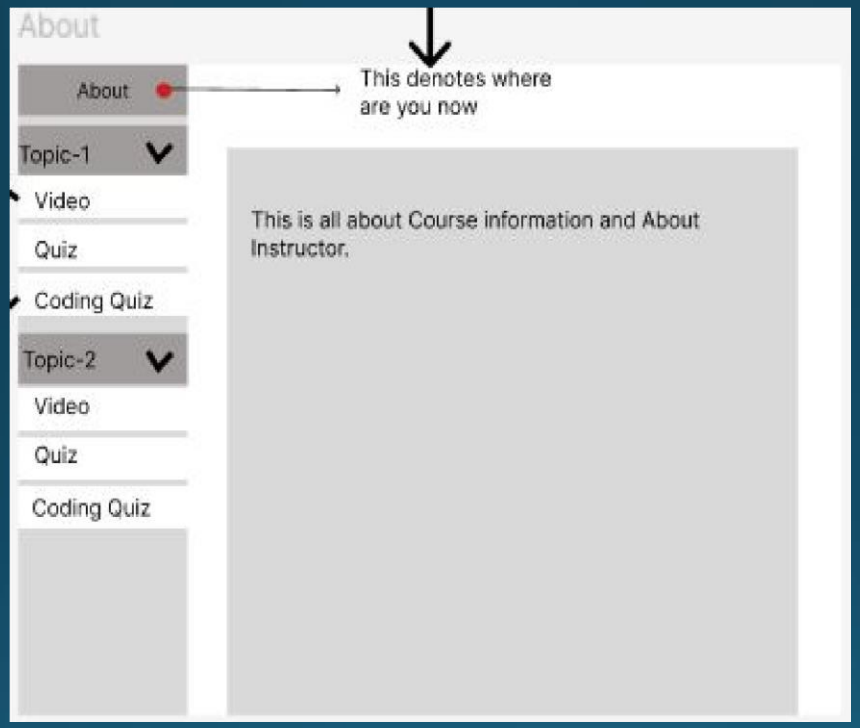
Wireframe: Instructor Login & Dashboard



Wireframe: Student Registration

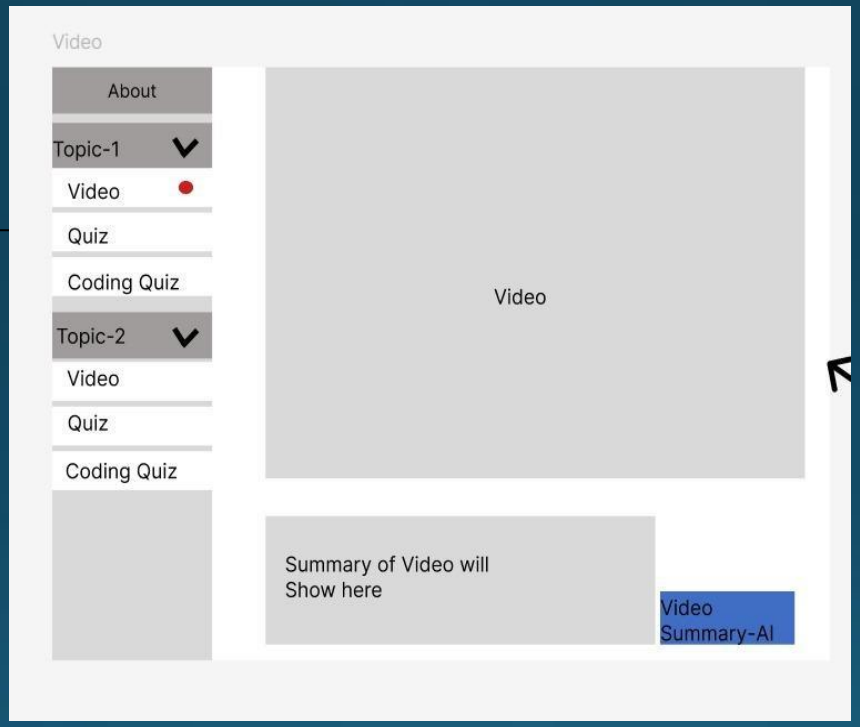


Wireframe: Course Page



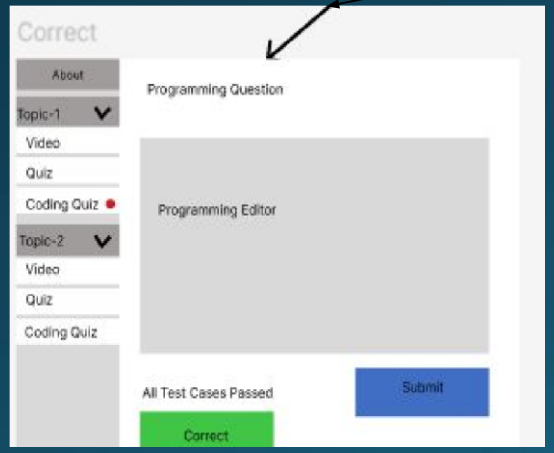
Wireframe: Course Videos

Quiz can contain multiple Questions with multiple options



Wireframe: Course Coding Quiz

Correct Answer



Wrong Answer

Usability Design Guidelines and Heuristics

1. Visibility of system status: Keep users informed about what is going on through appropriate feedback within a reasonable time.
2. Match between system and the real world: Use language and concepts familiar to the user, rather than system-oriented terms.
3. User control and freedom: Provide ways for users to easily exit or undo actions.
4. Consistency and standards: Ensure the interface is consistent with common conventions and within itself.
5. Error prevention: Design to prevent errors from occurring and provide helpful error messages when they do.
6. Recognition rather than recall: Make objects, actions, and options visible. Users should not have to remember information from one part of the dialogue to another.
7. Flexibility and efficiency of use: Allow both novices and experienced users to perform tasks efficiently.
8. Aesthetic and minimalist design: Only include relevant information. Excessive information can lead to cognitive overload.
9. Help users recognize, diagnose, and recover from errors: Error messages should be expressed in plain language, precisely indicating the problem and suggesting a solution.
10. Help and documentation: Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation.