# PROJECT DESIGN PHASE

## ****1. Introduction****

The Project Designing Phase defines the structural and functional blueprint of the Educational AI Assistant. While the Ideation and Project Description phases focus on the idea and scope, the design phase explains **how the system is organized and how the components interact with each other**. It bridges the gap between conceptual planning and actual coding, ensuring that the final implementation is both efficient and user-friendly. A well-thought-out design also allows for easier debugging, scalability, and the addition of new features in future versions of the system.

## ****2. System Architecture****

The architecture of the Educational AI Assistant follows a **modular design** that separates the back-end processing from the front-end interface. At the back end, the IBM Granite 3.2 2B Instruct model is loaded using Hugging Face Transformers and executed with PyTorch. This model is responsible for generating responses to user inputs. At the front end, Gradio provides a browser-based interface that allows users to interact with the assistant through textboxes and buttons.

The system is designed to automatically detect whether the hardware supports GPU computation. If a GPU is available, PyTorch offloads the model execution to it, enabling faster response times. If not, the system defaults to CPU execution, ensuring compatibility across different environments. This architecture ensures that the assistant can be deployed on both high-performance systems and standard personal computers without major adjustments.

## ****3. Workflow Design****

The workflow begins with the **user input**, which is entered into a textbox on the Gradio interface. This input is passed to a specific function, depending on whether the user requests a concept explanation or a quiz. The input is then transformed into a **prompt** — for example, “Explain the concept of gravity with examples” or “Generate 5 quiz questions about Newton’s Laws of Motion.”

The prompt is processed by the **tokenizer**, which converts it into tokens that can be understood by the model. PyTorch handles the computational logic, running the model inference and generating output tokens. These tokens are then decoded back into natural language text. A cleaning step ensures that the original prompt is removed from the response, leaving only the assistant’s output. Finally, the result is displayed in the Gradio output textbox. This workflow ensures a smooth cycle from input to output, hiding the complexity of the underlying AI operations from the user.

## ****4. User Interface Design****

The interface design is centered on simplicity and accessibility. The Gradio interface is divided into two **tabs**: one for Concept Explanation and one for Quiz Generation. Each tab has an input textbox, a button to trigger the response, and an output textbox to display the result. This separation ensures that users can easily switch between learning and assessment modes without confusion. The layout is designed to minimize clutter, making it usable for students of all age groups, including beginners with little technical knowledge.

The Concept Explanation tab produces detailed multi-paragraph answers enriched with examples, while the Quiz Generator tab produces five questions in varied formats along with a dedicated “ANSWERS” section. By organizing the interface in this way, the system ensures clarity and maintains a structured learning experience.

## ****5. Data Flow Design****

The data flow of the system can be visualized in three main stages:

* **Input Stage** → The user enters a concept or topic in the textbox.
* **Processing Stage** → The system tokenizes the input, runs inference using the IBM Granite model, and generates output tokens.
* **Output Stage** → The tokens are decoded, cleaned, and displayed as human-readable text in the output textbox.

This simple yet effective flow ensures that even complex NLP processes occur seamlessly from the user’s perspective.

## ****6. Design Goals****

The design of the Educational AI Assistant was guided by several key goals:

* **Modularity**: Functions like generate\_response, concept\_explanation, and quiz\_generator are independent, making the code easier to maintain and extend.
* **Scalability**: The system can handle both CPU and GPU execution, making it adaptable to various environments.
* **Usability**: The interface is intuitive, with minimal inputs required from the user.
* **Reliability**: Built-in safeguards, such as ensuring the tokenizer has a pad token, reduce the risk of errors during execution.
* **Extensibility**: The modular design allows for easy addition of new features, such as summarization or flashcard generation, in future versions.

## ****7. Conclusion****

The Project Designing Phase demonstrates how the Educational AI Assistant balances advanced AI capabilities with a user-friendly design. By clearly defining the architecture, workflow, interface, and data flow, the system ensures reliable performance while remaining accessible to its target audience. The thoughtful design choices lay a strong foundation for future enhancements and establish the assistant as a practical and scalable solution for modern education.