**Requirement Analysis Phase:**

**1.Indroduction**

The **Requirement Analysis Phase** is one of the most important stages of software development because it lays the foundation for everything that follows. It involves gathering, studying, and defining the needs of the project so that the final product delivers exactly what users expect. Without a proper analysis of requirements, projects often suffer from scope creep, delays, or misalignment with user needs.

For the Educational AI Assistant, this phase is particularly crucial because the system is intended to serve a **diverse group of users** such as students, teachers, parents, and self-learners. Each of these users has unique expectations, and the assistant must be carefully designed to fulfill them. For instance:

* **Students** want clear explanations of concepts and practice exercises.
* **Teachers** need quick quiz generation tools to save preparation time.
* **Parents** may want simplified answers to help their children with homework.
* **Self-learners** expect an easy way to explore new topics without formal guidance.

Requirement Analysis ensures that these expectations are transformed into **functional requirements** (what the system must do) and **non-functional requirements** (how well the system must perform).

Another reason this phase is important is the project’s dependency on **AI and NLP technologies**. Since the assistant relies on a pre-trained model (ibm-granite/granite-3.2-2b-instruct), it is essential to define hardware requirements, performance expectations, and interface simplicity. This helps avoid technical limitations that might otherwise impact usability.

In short, this phase acts as a **blueprint** that ensures the project has:

* A clear definition of goals.
* Well-documented requirements.
* Identified constraints and challenges.
* Alignment between user needs and system design.

By completing the Requirement Analysis phase thoroughly, the project reduces risks, improves planning accuracy, and sets the stage for a successful implementation of the Educational AI Assistant.

**2. Functional Requirements – Educational AI Assistant**

1. Load Pretrained Model

The system must load the ibm-granite/granite-3.2-2b-instruct language model using the Hugging Face transformers library.It should detect if GPU is available and load the model with the appropriate settings for optimized performance.

2. Explain Concepts:

The user must be able to input any concept (e.g., “machine learning”) into the interface.The system should generate a detailed explanation of the concept with examples.

3. Generate Quizzes

The user must be able to input a topic (e.g., “physics”) into the interface.The system should generate 5 quiz questions related to the topic, including various question types like multiple choice, true/false, and short answer.At the end of the quiz, the system should provide the correct answers.

4. Interactive Web Interface

The system must allow users to input concepts or topics using textboxes.Buttons should be provided to trigger the explanation or quiz generation processes.The results should be displayed in textboxes below the input fields without needing to reload the page.

5. Token Handling

The tokenizer should ensure that a padding token is available during generation to avoid processing errors.

6. Adapt to Hardware Environments

The system must support running on both CPU and GPU configurations.It should automatically adjust its settings based on the available hardware.

**3.Non-Functional Requirements – Educational AI Assistant**

1. Performance

The system should generate explanations and quizzes quickly, ideally within a few seconds per request, depending on the user’s hardware.It should efficiently handle multiple requests without freezing or crashing.

2. Usability

The interface must be simple and intuitive, requiring no technical knowledge from the user.Input fields should have clear labels and placeholders to guide users.Results should be easy to read and understand, with proper formatting.

3. Scalability

The system should be designed so that new features (e.g., summarization, translation) can be added in the future without major changes to the codebase.

4. Portability

The application should run on different operating systems (Windows, macOS, Linux) and in environments with or without GPU support.

5. Reliability

The system should handle unexpected or incorrect inputs without crashing.It should maintain a stable connection during user interactions.

6. Maintainability

The code should be organized and documented so future developers can understand, debug, and update it easily.

7. Accessibility

The interface should be accessible via web browsers on both desktop and mobile devices.

**4. Constraints – Educational AI Assistant**

1. Model Size and Resource Limitations

The ibm-granite/granite-3.2-2b-instruct model is large and requires significant memory (RAM or VRAM) for inference.

On systems without a GPU or with limited memory, the model may run slower or fail to load.

2. Hardware Dependency

The system’s performance is dependent on the availability of a GPU for faster processing.If GPU is unavailable, the system must fall back to CPU, which may lead to reduced performance.

3. Internet Connectivity

The model is downloaded from Hugging Face’s repository at runtime, so internet access is required during setup and potentially for updates.

4. Model Limitations

The accuracy and relevance of the generated explanations and quizzes depend on the pretrained model’capabilities.The model may not always provide correct or contextuallyperfect information.

5. Input Limitations

The input text is truncated if it exceeds the maximum length (512 tokens), which may lead to incomplete processing.Users need to input clear and concise concepts/topics to get meaningful results.

6. Security and Privacy

he system must not store or misuse user input, as sensitive data might be entered.Hosting the application via Gradio’s sharing feature may expose it to public access unless secured.

7. Scalability and Concurrency

The system is not designed for high-traffic usage or large numbers of simultaneous users without additional infrastructure.

8. Framework Dependency

The system depends on Python libraries (transformers, torch, gradio) which need to be compatible with the user’s environment.Library versions must be maintained to avoid conflicts or deprecations.

**5. Conclusion**

The Educational AI Assistant is designed to help users easily access detailed explanations and interactive quizzes on various concepts and topics. By leveraging a powerful pretrained language model (ibm-granite/granite-3.2-2b-instruct) and a user-friendly web interface built with Gradio, the system fulfills its goal of providing educational support in an accessible and efficient manner.The functional requirements ensure that users can input concepts or topics and receive explanations and quizzes tailored to their needs. The non-functional requirements address aspects such as performance, usability, scalability, and reliability, making the system practical and dependable for everyday use. The defined constraints, including hardware limitations, dependency on internet connectivity, and model accuracy, highlight areas where careful planning and management are required to ensure smooth operation.

In conclusion, this assistant serves as a flexible, scalable, and approachable educational tool. It is suitable for learners, educators, and developers aiming to enhance knowledge-sharing and interactive learning, while also acknowledging limitations that must be considered during deployment and usage. With thoughtful design and proper infrastructure, the system can be expanded and improved to meet evolving educational demands.