1. Identify System and Related Actors

1.1 Understand the System and Its Requirements

Analyze the main functions and user interactions for the module you have selected.

1.2 Identify Actors

Identify who (users) or what (external systems) interacts with your chosen system.

1.3 Identify Requirements

1.3.1 Functional Requirements (What the system must do)

The system must provide **at least 20 functional requirements.**

Example Functional Requirements

- **FR1**: The system shall allow students to create accounts and log in securely before taking the assessment.
- **FR2**: The system shall deliver separate test modules for reading, writing, listening, and speaking skills.
- **FR3**: The system shall accept voice input for the speaking module.
- **FR4**: The system shall evaluate responses and assign levels (A1–C2) for each skill based on scoring algorithms.

1.3.2 Non-Functional Requirements (How the system should perform)

The system must provide **at least 5 non-functional requirements.**

Example Non-Functional Requirements

- **NFR1**: The system shall complete test analysis and generate reports within 10 seconds of test submission.
- **NFR2**: The user interface shall be very user friendly so that users with minimal technical skills can complete assessments without guidance.
- **NFR3**: The system shall support simultaneous test-taking by at least 500 users without performance degradation.**Note:**

The functional and non-functional requirements you identify at this stage will form the foundation for the next phases of your project.
You will use these requirements to:

- Design your system architecture
- Create use case diagrams and scenarios
- Build UML diagrams (e.g., class, sequence)
- Develop prototypes and interfaces

Therefore, take special care to define clear, complete, and relevant requirements.
Ambiguous or missing requirements will lead to problems in later stages.

2. Define and Describe Use Cases

2.1 Define User-Goal Level Use Cases

List each primary use case that achieves a user-level goal. Group CRUD (Create, Read, Update, Delete) operations into a single “<<Manage ...>>” use case when applicable.
For example:

- **Manage Products**: Covers adding, viewing, updating, and deleting products.
- **View Product Catalog**
- **Place an Order**
- **Process Payments**
- **Generate Reports**
- **Handle Customer Support Requests**

NOTE: Each use case defined in this section should correspond to one of the **20 functional requirements** identified in Section **1.3.1**.This ensures that all functional requirements are represented through appropriate user-goal use cases.

2.2 Draw a UML Use Case Diagram

- Represent each use case with an oval labeled with its name.
- Draw actors as stick figures outside the system boundary and connect each actor to the use cases they interact with.
- Use <<extend>> or <<include>> relationships where needed, e.g., if one use case is a subset of another.
- Ensure the diagram includes **all use cases.**
- Please ensure that **UML diagrams are created using draw.io or similar online tools.** These platforms provide the necessary symbols and features for accurately representing UML diagrams.

2.3 Write Use Cases in Brief Format

Provide **brief descriptions for all 20 use cases** identified earlier, including the following details:

- **Title**: Name of the use case.
- **Primary Actor**: Main user or system interacting with your chosen system.
- **Goal**: Purpose of the use case.
- **Brief Description**: A short summary of what the use case accomplishes.

Example:

- **Title**: Place an Order
  - **Primary Actor**: Customer
  - **Goal**: To purchase selected items.
  - **Brief Description**: The customer selects items, proceeds to checkout, and completes payment to place an order.

2.4 Complete Fully Dressed Use Case

For each use case, write a fully detailed version — including all core components (for all 20 use cases). For example, “Manage Orders”. Describe each in detail, including:

- **Title:** Name of the use case.
- **Primary Actor:** Main user or system interacting with your chosen system.
- **Goal in Context**: Describe why this use case is important to the system.
- **Stakeholders and Interests**: List other parties involved and their interests.
- **Preconditions**: Conditions required for the use case to start.
- **Success Guarantee**: What is achieved if the use case completes successfully.
- **Main Success Scenario**: Step-by-step description of the process.
- **Extensions**: Cover at least two common alternate paths or errors that could occur.

Example:

- **Title**: Manage Orders
- **Primary Actor**: Admin
- **Goal in Context**: Ensures orders are tracked, updated, and completed accurately.
- **Stakeholders and Interests**:
  - **Customer**: Wants accurate order updates and timely delivery.
  - **Shipping Partner**: Relies on accurate order information for delivery.
- **Preconditions**: Admin is logged in with the correct permissions.
- **Success Guarantee**: The order list is updated and reflects current status.
- **Main Success Scenario**:
  - Admin logs into the system and opens the order management module.
  - Admin views all orders and selects one to update.
  - Admin changes the order status to “Shipped.”
  - System updates the order status and notifies the customer.
- **Extensions**:
  - **3a**: If the order is canceled, the system confirms before updating.
  - **4a**: If the status cannot be updated, an error message is displayed.