

Chapter No 3: Analysis and design

3.1 Requirements

A requirement is a thing that must be accomplished by the system. It is a goal or state that the software must fulfill in order to be counted as successful. When designing a software system, designers will normally formulate requirements in functional terms, which means they will state what the system should do, but not how it should be done. The designers will take care of the details. Sometimes, designers will formulate these requirements in too much detail or just plain unrealistically, without considering the practicalities. There are two types of requirements given below:

3.2 Functional Requirements

Functional requirement: This describes what the system has to do. They define the services, behaviors, and functions that a system must provide to its users. The requirements are focused on systems, action, inputs, processing, and output, and do not describe how the system is technically implemented.

In many situations, when user requirements are written for the requester rather than the end user, functional requirements are often merged with non-functional requirements. This practice is common in organizations that have a strong Information Technology department responsible for developing and implementing the system. The functional requirement of Learnity is the following:

- User Authentication: Registration and login system with different access levels for Students, Teachers, and Administrators.
- Multimedia Content Management: Functionality for teachers to upload, organize recorded video lectures and PDF study materials.
- Sindh Board Filtering: Search and filter engine for users to browse content specifically by Grade (1-12) and Subject

- Dual Learning Modes: Support for both asynchronous (recorded) video watching and synchronous (live) classroom interactions.
- Monetization & Local Payments: System to manage "Free" and "Paid" content options with JazzCash and Easypaisa verification
- Evaluation Tools: Automated quiz modules and digital portal for assignment submission and teacher feedback.

3.1.2 Non-Functional Requirement

In software engineering, non-functional requirements focus on ensuring overall system quality rather than specific functionalities. Software systems are expected to demonstrate key quality attributes such as accuracy, performance, security, and ease of modification. Despite their importance, non-functional requirements are often challenging to define and implement in many projects, even though effective methods exist for addressing functional requirements that deliver the intended system behavior.

Non-functional requirements are a very useful reference for software practitioners, researchers, and students as they cover areas such as system performance, external interface requirements, design constraints, and overall software quality attributes. Non-functional requirements are not easily testable, and hence they are evaluated by subjective evaluation instead of objective measurement

The website Learnity has the following non-functional requirements

3.2 Design of Learnity Website

1.1.1 USE CASE FOR TEACHERS

To use the website the teacher have to first register themselves on the website, if he/she is not registered on the website. Then the teacher can upload video lectures, notes and overall study material, they can also conduct the live classes for their students. After sometime teacher can turn on the monetization on their account and get subscription from their students

1.1.2 USE CASE FOR STUDENT

The student use case explains how students can use the platform to access learning content. The student creates an account and logs into the system, searches for available teachers, and subscribes to a teacher's account to get access to learning content. Once the student subscribes to the teacher's account, the student can participate in live video lectures, view video lectures that the teacher has already uploaded, and view or download study materials that the teacher has provided. The student use case highlights how the platform allows students to learn remotely and access both live and pre-recorded learning content.

1.1.3 USE CASE FOR ADMIN

The use case for the administrator explains the function of overseeing and managing all activities that take place on the website. The administrator has complete control over the system and is responsible for managing student and teacher accounts, including the ability to upload, delete, and manage content and user activities. The administrator is also responsible for monitoring all materials that have been uploaded by teachers and ensuring that they comply with community guidelines; if there is any material that does not comply with these guidelines, the administrator has the right to limit, control, or delete it. The administrator is also responsible for the maintenance of the website to ensure that everything runs smoothly and that the website provides a safe environment for all users.

1.1.4 ACTIVITY DIAGRAM FOR TEACHER

The process starts with the teacher logging into the system. After that, the teacher uploads videos and learning materials and can also have live classes for the students. After all these are done, the teacher logs out of the system, which is the end of the process. As shown in the activity figure

1.1.5 ACTIVITY DIAGRAM FOR STUDENTS

Activity diagram for the student shows that the student logs into the system using valid credentials. After logging into the system, the student browses through the list of available teachers and subscribes to the account of a teacher of their choice. After the student has subscribed to the teacher's account, they are able to participate in live video lectures, pre-recorded video lectures, and view notes and study materials uploaded by the teacher. After the student is done with the learning process, they log out of the system, thus concluding the process.

1.1.6 **ACTIVITY DIAGRAM FOR ADMIN**

After logging in to the website, the admin can perform activities like managing the teacher and student accounts. He reviews the content uploaded by teachers and controls all the activity and settings in the website as shown in the figure

1.1.7 ENTITY RELATIONSHIP DIAGRAM

Entity Relationship Diagram (ERD) is a graphical tool that is used in database design to represent entities in a system and the relationships between them. Entities are real-world objects like users, courses, or payments, while attributes are descriptions of the characteristics of each entity. Relationships are used to describe how entities are related to each other, for example, one-to-one, one-to-many, or many-to-many. An ERD can assist a developer or designer in understanding the structure of a database, organizing data effectively, and establishing the correct relationships before creating a database.

1.1.8 **SYSTEM ARCHITECTURE**