**Functional vs. Non-Functional Requirements**

In software development, requirements are typically divided into **functional** and **non-functional** categories. Both are critical to the successful design and implementation of a system, but they address different aspects of the system’s behavior and quality.

Let’s break down both types:

**Functional Requirements:**

Functional requirements describe what the system **should do**—the features, actions, and behavior expected from the system. They focus on **specific functionality** the system must provide to meet the user's needs.

**Examples of Functional Requirements:**

1. **User Authentication:**
   * The system must allow users to log in using their credentials (email and password).
   * Users must be able to reset their password using an email verification link.
2. **Course Management:**
   * Teachers must be able to create, edit, and delete courses.
   * Teachers should be able to upload materials (documents, videos) to their courses.
3. **Student Enrollment:**
   * Students should be able to enroll in available courses by clicking the "Enroll" button.
   * The system must ensure that a student cannot enroll in a course if it exceeds the course’s capacity.
4. **Assignment Submission:**
   * Students must be able to submit assignments and projects directly through the system.
   * Teachers should be able to grade assignments and provide feedback.
5. **Progress Tracking:**
   * The system must track each student’s progress in terms of completed lessons, assignments, and quizzes.
   * Teachers and admins must be able to generate reports that show student progress over time.
6. **Communication Tools:**
   * Users should be able to send and receive messages within the platform (student-to-teacher, student-to-student, etc.).
   * The system should support push notifications for upcoming deadlines, announcements, and messages.
7. **Grading and Feedback:**
   * Teachers must be able to grade assignments, quizzes, and exams.
   * Students should be able to view their grades and teacher feedback for completed work.

**Non-Functional Requirements:**

Non-functional requirements describe how the system **should perform**—the qualities or attributes of the system that support overall user experience and system stability. These requirements often focus on **system performance, security, scalability, and reliability**, among other characteristics.

**Examples of Non-Functional Requirements:**

1. **Performance:**
   * The system must handle up to 10,000 concurrent users without significant performance degradation.
   * A page should load in less than 3 seconds under normal load conditions.
2. **Scalability:**
   * The system should be scalable to accommodate an increasing number of users, courses, and data.
   * The system must be able to scale horizontally by adding more servers as the number of users grows.
3. **Reliability:**
   * The system must have an uptime of 99.9% or higher.
   * The platform must be resilient to failures and provide fallback mechanisms if a service goes down.
4. **Security:**
   * User data must be encrypted both at rest and in transit.
   * The system must enforce strong password policies (e.g., minimum length, complexity).
   * Sensitive data (e.g., grades, student information) must be stored securely and only accessible by authorized users.
5. **Usability:**
   * The system must have an intuitive and easy-to-navigate user interface.
   * It must be accessible to users with disabilities, conforming to WCAG (Web Content Accessibility Guidelines).
   * The platform should be mobile-friendly and responsive, working seamlessly across desktop, tablet, and mobile devices.
6. **Maintainability:**
   * The codebase should be modular and well-documented to facilitate future updates and bug fixes.
   * The system must support easy updates and patches without significant downtime.
7. **Backup and Recovery:**
   * The system must back up all user data, including course materials and student progress, daily.
   * In the event of a system failure, data should be recoverable within 24 hours.
8. **Compliance:**
   * The system must comply with local data protection laws and regulations, such as GDPR (General Data Protection Regulation) in Europe.
   * It must support data retention policies and give users the ability to delete or export their data.

**Key Differences Between Functional and Non-Functional Requirements:**

| **Aspect** | **Functional Requirements** | **Non-Functional Requirements** |
| --- | --- | --- |
| **Definition** | Describe the **features** and **capabilities** of the system. | Describe the **performance**, **quality**, and **constraints** of the system. |
| **Focus** | Focuses on the **behavior** of the system. | Focuses on **how well** the system performs. |
| **Examples** | Login process, course creation, user enrollment, messaging. | Security, performance, scalability, reliability, usability. |
| **Nature** | Specific and **measurable actions** the system must carry out. | **Qualitative** or **system-wide constraints** (e.g., performance, security). |
| **Impact** | Directly impacts user interactions with the system. | Affects the overall quality, reliability, and user satisfaction. |

**Why Both Are Important:**

* **Functional Requirements** ensure the system performs the tasks and processes that users need.
* **Non-Functional Requirements** ensure the system does so in a way that meets the quality expectations (e.g., fast, secure, easy to use) and can scale over time.

Both sets of requirements are essential for creating a well-rounded and successful system. Functional requirements define the **what** the system should do, while non-functional requirements define the **how well** it should do it.