LAB-5

1. Introduction

In software engineering, requirements are classified into two major types:

1.1 Functional Requirements (FRs)

Functional requirements define the core functionality of a system. They describe what the system must do and how users will interact with it.

🔹 **Examples in the Stock Market Prediction System:**

- Fetch and process historical stock market data.
- Train and test ML, DL, QML, and QNN models for stock prediction.
- Display stock trends and forecasted prices through a user interface.
- Allow users to input stock symbols and retrieve predictions.
- Compare multiple models' performance for accuracy evaluation.

1.2 Non-Functional Requirements (NFRs)

Non-functional requirements define the quality attributes of a system. They describe how the system should perform rather than what it should do.

🔹 **Examples in the Stock Market Prediction System:**

- Prediction results should be generated within **5 seconds**.
- Data security should be ensured through **AES-256 encryption**.
- The system should be **scalable**, handling **multiple concurrent users**.
- The user interface should be **responsive and mobile-friendly**.
- The system should support **real-time data updates** from stock APIs.

---

2. Practical Lab Task

Step 1: Select a Real-Time Software Project

📌 **Stock Market Prediction System using ML, DL, QML, and QNN**

ID	Functional Requirement
FR1	The system should fetch historical stock market data from APIs like Yahoo Finance.
FR2	The system should preprocess and clean stock data for feature extraction.
FR3	The system should train and evaluate ML, DL, QML, and QNN models.
FR4	Users should be able to input stock symbols and receive predictions.
FR5	The system should compare the accuracy of different predictive models.
FR6	The system should visualize stock trends using graphs and charts.
FR7	The backend should support REST API calls for data retrieval and predictions.
FR8	Users should be able to select different prediction models for comparison.

Step 2: Identify Functional RequirementsStep 3: Identify Non-Functional Requirements

ID	Non-Functional Requirement
NFR1	The system should respond to user prediction requests within **5 seconds**.

NFR2	Data should be encrypted using **AES-256 encryption** for security.
NFR3	The system should support at least **100+ concurrent users**.
NFR4	The user interface should be **mobile-friendly and accessible**.
NFR5	The system should maintain **99.9% uptime** for real-time access.
NFR6	Only **authorized users** should have access to API calls and model results.
NFR7	The system should be capable of **handling large stock datasets efficiently**.

Step 4: Documenting Requirements

A **Software Requirements Specification (SRS) document** should be prepared for the Stock Market Prediction System, ensuring all functional and non-functional requirements are properly categorized and structured.

Step 5: Group Discussion & Analysis

After discussion, the following results were obtained:

✅ **Common Functional Requirements Identified:**

- Stock data collection and preprocessing.
- Real-time stock market trend prediction.
- Comparison of ML, DL, QML, and QNN models.
- User interaction via a web-based dashboard.

✅ **Common Non-Functional Requirements Identified:**

- The system should generate predictions within **5 seconds**.
- Stock data should be stored securely with **encryption**.

- The system should be **scalable** to handle multiple stock requests.
- UI should be **responsive** for mobile and desktop users.

Each group presented their findings, highlighting **challenges faced and key observations** while differentiating between functional and non-functional requirements.

---

3. Conclusion

Through this exercise, we have learned to differentiate between functional and non-functional requirements and document them effectively for a **real-time stock market prediction system**. Well-defined requirements are essential for building **accurate, scalable, and secure** software. Understanding these concepts helps in designing **better predictive systems**, ensuring smooth development and deployment.