What is Software Requirements?

According to IEEE standard 729, a requirement is defined as follows:

•A condition or capability needed by a user to solve a problem or achieve an objective

•A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification or other formally imposed documents

•A documented representation of a condition or capability, as in 1 and 2.

Software Requirements are mainly classified into three types:

- Functional requirements
- Non-functional requirements
- Domain requirements

Functional Requirements:

Examples:

User Authentication: The system must allow users to log in using a username and password.

Search Functionality: The software should enable users to search for products by name or category.

Report Generation: The system should be able to generate sales reports for a specified date range.

Explanation: Functional requirements specify the actions that the software needs to perform.

Non-functional Requirements:

Non-functional requirements describe how the software performs a task rather than what it should do.

Examples:

Performance: The system should process 1,000 transactions per second.

Usability: The software should be easy to use and have a user-friendly interface.

Reliability: The system must have 99.9% uptime.

Security: Data must be encrypted during transmission and storage.

Explanation: Non-functional requirements are about the system’s behavior, quality, and constraints.

Domain Requirements:

Definition: Domain requirements are specific to the domain or industry in which the software operates.

Examples:

Healthcare: The software must comply with HIPAA regulations for handling patient data.

Finance: The system should adhere to GAAP standards for financial reporting.

E-commerce: The software should support various payment gateways like PayPal, Stripe, and credit cards.

Explanation: Domain requirements reflect the unique needs and constraints of a particular industry.

Requirement Analysis:

involves gathering, analyzing, and documenting software requirements to ensure the final product meets user needs.

Key activities in requirement analysis:

Elicitation: Gathering requirements from stakeholders, users, and domain experts through interviews, surveys, and workshops.

Analysis: Critically reviewing the gathered requirements to ensure they are complete, consistent, and feasible. This involves identifying and resolving any ambiguities, inconsistencies, or missing information.

Evaluation and Synthesis: Judging whether the requirements are valuable, determining all necessary functions, and understanding the overall system behavior, data flow, and constraints.

Specification: Documenting the requirements in a clear, concise, and unambiguous manner, often in a Software Requirements Specification (SRS) document.

Validation: Reviewing the documented requirements with stakeholders to confirm that they accurately reflect the needs of the project and are complete.

Management: Continuously managing requirements throughout the development lifecycle to document and communicate any changes or updates

Why requirement analysis is important:

Stakeholder alignment: Ensures the final product meets the needs of its intended users, leading to higher satisfaction.

Clear scope: Defines project objectives and constraints, which helps prevent misunderstandings and scope creep later in the project.

Risk reduction: Identifies potential issues and risks early in the process, allowing for proactive management.

Improved communication: Promotes collaboration and a clear understanding among all team members and stakeholders.

Resource efficiency: Helps prioritize requirements, enabling the effective allocation of resources to deliver key features on time and within budget.

Requirement Traceability Matrix:

What is RTM?

A document that describes the relationship between customer requirements and test artifacts, especially test cases

Key components of an RTM

Requirement ID: A unique identifier for each requirement.

Requirement Description: A clear explanation of the requirement.

Source: Where the requirement originated, such as a stakeholder, document, or user story.

Design Element ID: The identifier for the design component that implements the requirement.

Test Case ID: The unique ID for the test case that verifies the requirement.

Test Result: The status of the test case (e.g., pass or fail).

Defect ID: Information about any defects found during testing, linked to the specific requirement and test case.

Status: The current progress of the requirement (e.g., in progress, completed).

Benefits of using an RTM:

Ensures full test coverage: Guarantees that every requirement has at least one test case.

Tracks and verifies functionality: Helps in identifying any requirements or functionalities that have been missed during testing.

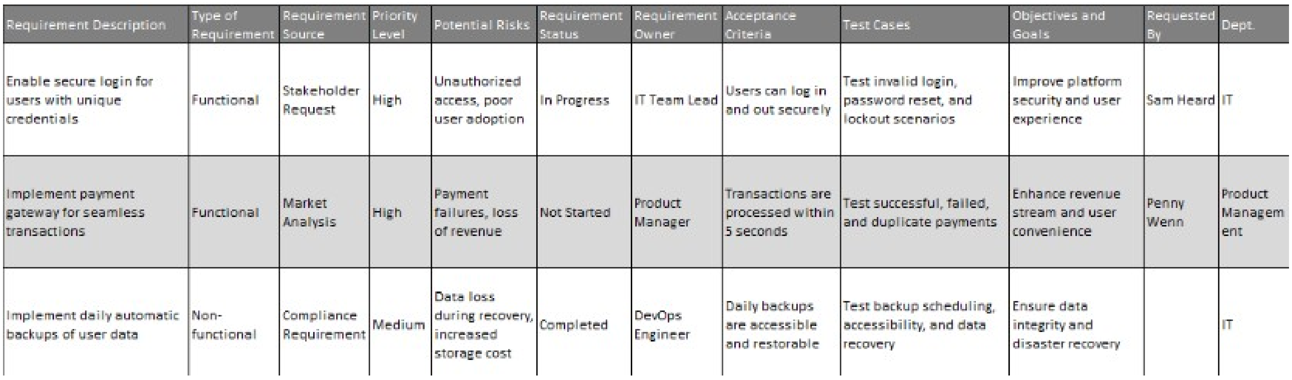
Provides a complete audit trail: Documents the relationship between requirements and their implementation and verification.

Supports change management: Makes it easier to see the impact of changes to requirements on other parts of the project.

Aids in project management: Helps in project planning, task separation, and tracking progress.

Improves collaboration: Makes traceability accessible to all team members, helping to identify gaps and risks early

RTM Sample:



Requirement Description	Type of Requirement	Requirement Source	Priority Level	Potential Risks	Requirement Status	Requirement Owner	Acceptance Criteria	Test Cases	Objectives and Goals	Requested By	Dept.
Enable secure login for users with unique credentials	Functional	Stakeholder Request	High	Unauthorized access, poor user adoption	In Progress	IT Team Lead	Users can log in and out securely	Test invalid login, password reset, and lockout scenarios	Improve platform security and user experience	Sam Heard	IT
Implement payment gateway for seamless transactions	Functional	Market Analysis	High	Payment failures, loss of revenue	Not Started	Product Manager	Transactions are processed within 5 seconds	Test successful, failed, and duplicate payments	Enhance revenue stream and user convenience	Penny Wenn	Product Management
Implement daily automatic backups of user data	Non-functional	Compliance Requirement	Medium	Data loss during recovery, increased storage cost	Completed	DevOps Engineer	Daily backups are accessible and restorable	Test backup scheduling, accessibility, and data recovery	Ensure data integrity and disaster recovery		IT