Introduction to Requirements Engineering

What is Requirements Engineering?

- **Definition**: Requirements Engineering (RE) is the process of gathering, documenting, analyzing, and managing the requirements of a software system, ensuring that the right software is built for the right stakeholders.
- **Example**: Think of developing a **Mobile Banking Application**. The stakeholders are the bank's customers, security experts, and business managers. Requirements Engineering helps to identify what features (like balance checking, fund transfer, or security measures) must be included in the app.

Why is Requirements Engineering Critical?

- **Example**: In **Airline Reservation Systems**, incorrect requirements (e.g., how to handle booking during peak travel times) can cause flight delays, customer dissatisfaction, and lost revenue.

Types of Requirements

Functional Requirements:

- **Definition**: These describe the system's behavior and functionality.
- **Example**: For an **E-commerce Website**:
  - "The system must allow users to add items to their shopping cart."
  - "The system must send an order confirmation email once payment is completed."

Non-Functional Requirements:

- **Definition**: These describe how the system should behave (performance, security, etc.).
- **Example**: For the same **E-commerce Website**:
  - "The system must support 500 concurrent users without a significant drop in performance."
  - "The website must comply with PCI DSS standards for payment security."

System Requirements:

- **Example**: For **Smart Home Automation**:
  - "The system must be able to integrate with existing home security systems, HVAC systems, and lighting systems."

User Requirements:

- **Example**: For a **Fitness App**:
    - "The app must allow users to track their steps, calories burned, and exercise routines."
    - "The app should allow users to set fitness goals and track progress over time."

Business Requirements:

- **Example**: For a **Healthcare System**:
    - "The system must improve patient scheduling efficiency by reducing appointment booking time."
    - "The system must adhere to local healthcare regulations for patient privacy (e.g., HIPAA in the U.S.)."

The Requirements Engineering Process

Step 1: Requirements Elicitation

- **Definition**: Gathering requirements from stakeholders through interviews, surveys, and other techniques.
- **Real-world Example**: For a **Hospital Management System**, you might conduct interviews with doctors, nurses, and administrative staff to understand their needs (e.g., scheduling, patient tracking, medical history management).
- **Techniques**:
    - **Interviews**: Direct conversations with stakeholders.
    - **Surveys**: Get input from a larger group of end users.
    - **Workshops**: Gather a team of stakeholders (e.g., doctors, IT staff) to brainstorm needs and requirements in real-time.
    - **Observation**: Watch how current systems are used in practice, e.g., observing how users interact with existing ticketing systems in a **Travel Booking Application**.

Step 2: Requirements Analysis

- **Definition**: Analyze the gathered requirements to ensure they are clear, consistent, and feasible.
- **Real-world Example**: After eliciting requirements for a **Food Delivery App**, the team analyzes if all requirements align with the business model (e.g., geographic areas for delivery, payment methods).
- **Conflict Resolution Example**: If one stakeholder demands a system that supports real-time tracking of delivery drivers, while another insists on minimizing app performance

overhead, you'd need to find a balance by considering technical constraints like network latency and battery usage.

Step 3: Requirements Specification

- **Definition**: Writing down the requirements in a formal document, such as a Software Requirements Specification (SRS).
- **Real-world Example**: A detailed **SRS** for a **Mobile Banking App** might contain specific sections like:
    - Functional Requirements: Account balance retrieval, fund transfers.
    - Non-functional Requirements: The app should be available 99.9% of the time.

Step 4: Requirements Validation

- **Definition**: Ensuring the requirements are correct, complete, and feasible.
- **Real-world Example**: In the **Healthcare Records System**, requirements would be validated by conducting walkthroughs with healthcare professionals and IT experts to ensure the system complies with medical regulations and meets users' needs.
- **Techniques**:
    - **Reviews**: Team reviews the SRS document to ensure completeness.
    - **Prototyping**: Build a small version of the product to demonstrate how a feature works.
    - **Test Cases**: Ensure the requirements are verifiable through testing.

Step 5: Requirements Management

- **Definition**: Tracking and managing changes to the requirements throughout the project lifecycle.
- **Real-world Example**: For an **Online Learning Platform**, you may have to update the requirements based on stakeholder feedback, such as adding live video streaming support after initial requirements are set.
- **Challenges**: Managing scope creep, especially in large projects like **E-Government Services**, where new laws and policies may affect requirements after project initiation.

Best Practices for Requirements Engineering

SMART Requirements:

- **Example**: A **Weather Forecasting System** might have a non-functional requirement like: "The system must process and display weather data in under 3 seconds."
  This is **Measurable** (processing time), **Specific** (display weather data), **Achievable**, **Realistic**, and **Time-bound**.

Traceability:

- **Example**: In a **Banking System**, you should be able to trace every business requirement (e.g., "The system must allow users to transfer money to an external account") through to the corresponding test cases and implementation.

Prioritization:

- **Example**: In the **E-Commerce Website**, the must-have features might include "checkout functionality," while "user profile customization" could be a nice-to-have feature that comes later.
- **Techniques**: MoSCoW (Must-have, Should-have, Could-have, Won't-have).

Stakeholder Involvement:

- **Example**: For a **Customer Relationship Management (CRM) System**, continuous collaboration with sales teams ensures the system effectively supports sales strategies.
- Involve stakeholders early and continuously (e.g., through regular feedback sessions) to avoid miscommunication and ensure alignment with user expectations.

Version Control:

- **Example**: For a **Social Media Application**, as new features (e.g., photo-sharing) are added, version control (e.g., Git) ensures that different development teams can update features independently without interfering with each other's work.

Tools for Requirements Engineering

- **JIRA**: Widely used for tracking requirements and managing changes in projects like **Agile Development for an Online Retail Platform**.
- **IBM Rational DOORS**: Useful for large projects like **Automated Car Systems**, where managing requirements across multiple teams and components is necessary.
- **UML Diagrams**: Use case diagrams for modeling user interactions (e.g., in **Student Information Systems**).

Case Study: Building a Mobile E-Commerce App

Step-by-Step Walkthrough:

1. **Elicitation**:
    - Conduct interviews with users, business managers, and IT team members.

    - Gather requirements such as “The system must allow customers to filter products by category” and “The app must support secure payment gateways like PayPal.”
2. **Analysis**:
    - Analyze conflicting requirements (e.g., speed vs. security) and prioritize features based on business goals.
    - Resolve conflicts and ensure all stakeholders are aligned.
3. **Specification**:
    - Draft a detailed SRS document that includes functional and non-functional requirements for the app.
4. **Validation**:
    - Conduct a prototype review with stakeholders (e.g., a clickable prototype of the app) to validate the user interface.
5. **Management**:
    - Track changes to the requirements as new features (e.g., adding multi-currency support) are requested during the project.