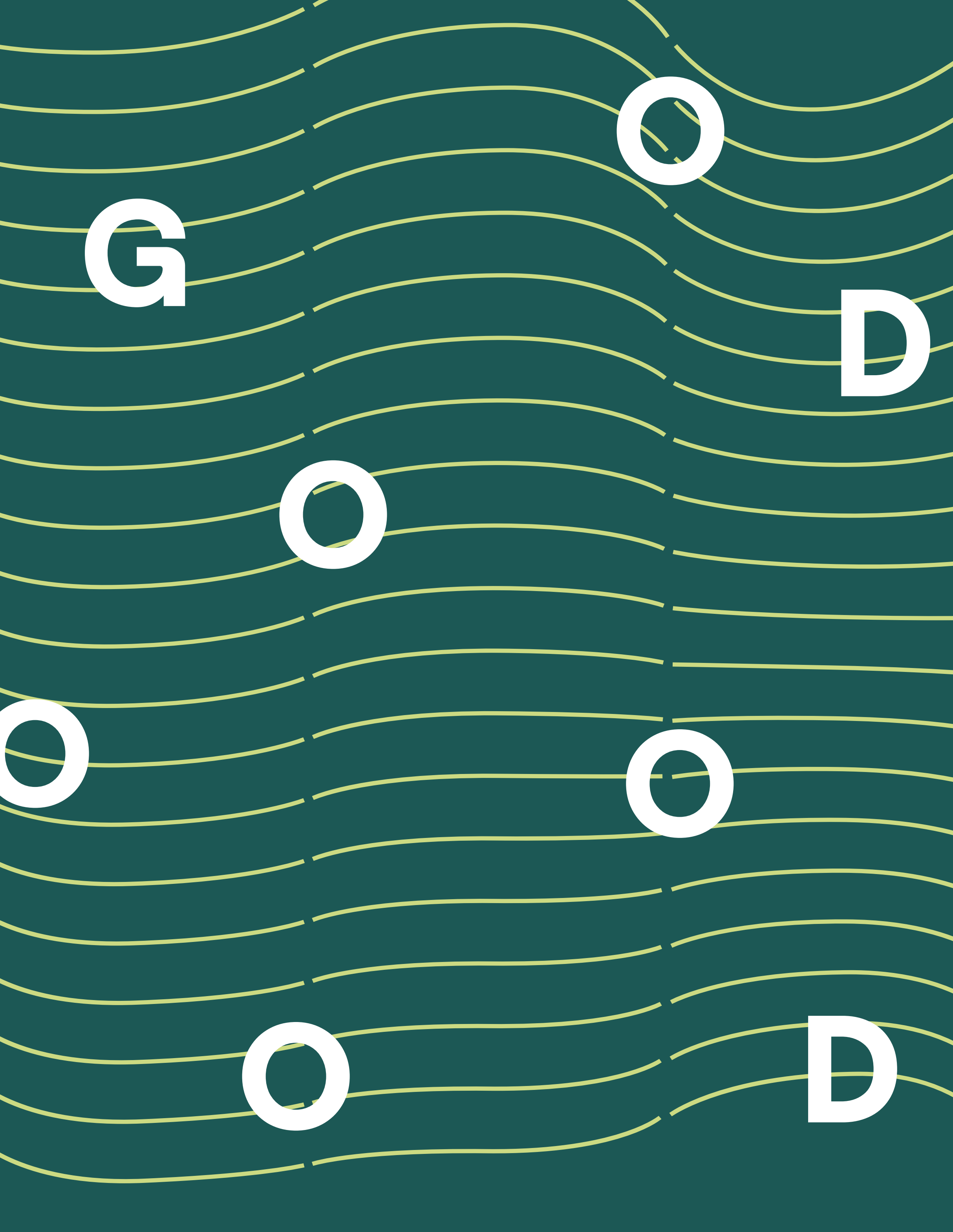
O
G
D
O
O
O
O
D

Requirement Gathering and Analysis

- Is a critical phase in the Systems Development Life Cycle (SDLC).
- It involves collecting and documenting the necessary requirements that a system must meet to ensure the final product fulfills the needs of its users and stakeholders.
- This phase ensures that the project team has a clear understanding of what the system should do and how it should perform.

A. Techniques for Requirement Collection

There are several techniques used to gather requirements from stakeholders:

1. **Interviews:**
   1. ○ **Purpose:** Interviews involve direct interaction with stakeholders to gather detailed information.
   2. ○ **Types:**
      1. ■ **Structured Interviews:** Predefined questions are asked, ensuring consistency across interviews.
      2. ■ **Unstructured Interviews:** More open-ended, allowing stakeholders to freely express their needs and concerns.
   3. ○ **Advantages:** Provides in-depth information, allows clarification of details, and helps build relationships with stakeholders.
   4. ○ **Disadvantages:** Time-consuming, may require multiple sessions, and depends on the interviewee's availability and willingness to share information.
2. **Surveys and Questionnaires:**
   1. ○ **Purpose:** Collect information from a large group of people using a standardized set of questions.
   2. ○ **Types:**
      1. ■ **Open-Ended Surveys:** Respondents can provide detailed answers in their own words.
      2. ■ **Closed-Ended Surveys:** Respondents choose from predefined options.
   3. ○ **Advantages:** Can reach a wide audience quickly, cost-effective, and

    1. provides quantitative data.
    2. ○ **Disadvantages:** Limited depth of responses, possible low response rates, and potential for misinterpretation of questions.

1. **Observations:**
    1. ○ **Purpose:** Observe users in their natural environment to understand how they interact with current systems and processes.
    2. ○ **Types:**
        1. ■ **Participant Observation:** The observer actively engages in the process being studied.
        2. ■ **Non-Participant Observation:** The observer remains detached and does not interfere with the process.
    3. ○ **Advantages:** Provides real-world insights into user behaviors and processes, identifies hidden requirements.
    4. ○ **Disadvantages:** Can be intrusive, time-consuming, and may not capture all aspects of the system's use.
2. **Workshops:**
    1. ○ **Purpose:** Collaborative sessions where stakeholders and the project team work together to define requirements.
    2. ○ **Advantages:** Encourages stakeholder engagement, facilitates consensus building, and provides immediate feedback.
    3. ○ **Disadvantages:** Requires careful planning and skilled facilitation, may be challenging to coordinate schedules.
3. **Document Analysis:**
    1. ○ **Purpose:** Review existing documentation related to the system or business processes to extract requirements.
    2. ○ **Advantages:** Provides historical context, useful for understanding legacy systems, and complements other techniques.
    3. ○ **Disadvantages:** Documents may be outdated or incomplete, and requires interpretation by the analyst.

1. **Software Review:**
    1. ○ **Purpose:** Examine existing software systems to gather insights about requirements, features, and potential improvements.
    2. ○ **Key Activities:**
        1. ■ Evaluate the features and functionalities of current software.
        2. ■ Collect user feedback on their experiences, issues, and suggestions.
        3. ■ Benchmark against similar systems or industry standards.
        4. ■ Review existing documentation, including user manuals and technical specs.
    3. ○ **Advantages:** Provides a detailed understanding of the current system's strengths and weaknesses, involves users directly, and helps identify both functional and non-functional requirements.
    4. ○ **Disadvantages:** Time-consuming, requires significant effort to collect and analyze feedback, and may be biased if only certain user groups are considered.
    5. ○ **Software Tools:**
        1. ■ **Jira:** Useful for tracking user feedback, issues, and suggestions.
        2. ■ **Confluence:** Helps in documenting the findings from software reviews and user feedback.
        3. ■ **SurveyMonkey:** Could be used to gather structured feedback from users through surveys.
        4. ■ **Google Analytics:** Provides insights into user behavior and interactions with web-based systems.
        5. ■ **Bugzilla:** Tracks and manages bug reports and user issues, helping identify recurring problems.

B. Modeling Techniques

Once requirements are gathered, they need to be documented and modeled to provide a clear and structured understanding of the system.

1. **Use Cases:**

   1. ○ **Purpose:** Describe how users (actors) interact with the system to achieve specific goals.
   2. ○ **Components:**
      1. ■ **Actors:** Entities (users or systems) that interact with the system.
      2. ■ **Use Case:** A specific interaction or function performed by the system.
      3. ■ **Scenario:** A sequence of steps within a use case.
   3. ○ **Advantages:** Helps in identifying functional requirements, easy to understand for stakeholders, and useful for creating test cases.
   4. ○ **Example:** A use case for a banking system might describe how a user logs in and checks their account balance.
   5. ○ **Software Tools:**
      1. ■ **Visual Paradigm:** A versatile tool for creating use case diagrams and other UML diagrams.
      2. ■ **Lucidchart:** Easy-to-use online diagramming tool.
      3. ■ **Microsoft Visio:** Widely used for creating a variety of diagrams, including use cases.

1. **Activity Diagrams:**
   1. ○ **Purpose:** Visualize the flow of activities and control in a system.
   2. ○ **Components:**
      1. ■ **Activities:** Actions or tasks performed.
      2. ■ **Transitions:** Flow between activities.
      3. ■ **Decision Points:** Branches in the flow based on conditions.
      4. ■ **Swimlanes:** Divisions representing different actors or systems.
   3. ○ **Advantages:** Clarifies complex workflows, identifies bottlenecks, and supports parallel processing.
   4. ○ **Example:** An activity diagram for an e-commerce checkout process might include activities such as selecting items, entering shipping information, and processing payment.
   5. ○ **Software Tools:**

        1. ■ **Lucidchart:** Ideal for creating activity diagrams and other flowcharts.
        2. ■ **Draw.io:** Free online tool for creating diagrams.
        3. ■ **IBM Rational Software Architect:** Advanced tool for modeling and design.

1. **Entity-Relationship (ER) Diagrams:**
    1. ○ **Purpose:** Model the data structure of a system by showing entities and their relationships.
    2. ○ **Components:**
        1. ■ **Entities:** Objects or concepts, typically represented by rectangles.
        2. ■ **Attributes:** Characteristics of entities, represented by ovals.
        3. ■ **Relationships:** Connections between entities, represented by diamonds or lines.
    3. ○ **Advantages:** Provides a clear visualization of the data model, useful for database design, and identifies data dependencies.
    4. ○ **Example:** An ER diagram for a school management system might include entities like Student, Course, and Enrollment, with relationships indicating which students are enrolled in which courses.
    5. ○ **Software Tools:**
        1. ■ **MySQL Workbench:** Excellent for designing and visualizing ER diagrams for databases.
        2. ■ **Oracle SQL Developer Data Modeler:** Comprehensive tool for data modeling.
        3. ■ **ER/Studio:** Professional tool for creating detailed ER diagrams.