Software Engineering

Unit 3:
Software Requirement Analysis and Specification

Compiled By: Sagar Rana Magar

Course Outline

- System and Software Requirements
- Type of Software Requirements: Functional and Non-Functional Requirements
- Domain Requirements, User Requirements
- Elicitation-and Analysis of Requirements:
  - Overview of Techniques
  - View Points,
  - Interviewing,
  - Scenarios,
  - Use-Case.
  - Ethnography,
  - Requirement Validation,
  - Requirement Specification,
  - Feasibility.

Requirement

- The requirement for a system are the description of what the system should do, the service or services that it provides and the constraints on its operation.
- 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.

Requirement Process Activities

1. **Requirements Elicitation**
   - Collect requirements from the stakeholders, customers, users, developers, etc.
2. **Requirements Analysis**
   - Study, analyze and model the problem to be solved
3. **Requirements Specification**
   - Define and document the requirements in and organized and precise manner
4. **Requirements Validation**
   - Ensure that the requirements are correct, complete, and consistent, and they satisfy the customer needs
5. **Requirements Management**
   - Trace/track requirements and manage requirements change throughout the software life-cycle

Requirement Elicitation

- Elicitation is the gathering and discovery of requirements from stakeholders and other sources.
- Elicitation is the first step of requirements development.
- A variety of techniques can be used such as interviews, document analysis, focus groups, etc.

Requirement Analysis

- The process of studying and refining system, hardware, or software requirements.
- Analysis involves
  - reaching a richer and more precise understanding of each requirement, and
  - representing sets of requirements in multiple, complementary ways.
- It is conceptual modeling of a solution to help understand the problem
- Architectural design and requirements allocation
  - Functional and non-functional grouping
  - Allocation of requirements to subsystem components
- Requirements negotiation and conflict resolution
  - Incompatible features, incompatible features and constraints, conflict between scope and resources, etc.

Requirement Specification

- A document that specifies the requirements for a system or component.
- Specification involves representing and storing the collected requirements knowledge in a persistent and well-organized fashion that facilitates effective communication and change management.
- Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards.
- Use cases, user stories, functional requirements, and visual analysis models are popular choices for requirements specification.

How do you know you have the right requirements?

Requirement Validation

- Validation involves techniques to confirm that the correct set of requirements has been specified to build a solution that satisfies the project's business objectives.
- Requirements validation:
  - Certify that the requirements meet the stakeholders' intentions
  - Ensure that the requirements define the right system
  - Ensure a common understanding across the project team and among the stakeholders
- Requirements Validation Techniques
  - Requirements reviews,
  - Prototyping to elicit and validate requirements
  - Review requirements analysis models
  - Prove properties of formal analysis models
  - Plan how each requirement will be verified in acceptance tests

Requirement Maintenance

- The process of eliciting, documenting, organizing, and tracking changing requirements and communicating this information across the project team to ensure that iterative and unanticipated changes are maintained throughout the project lifecycle.
- Requirements change during projects and there are often many of them.
- Management of this change becomes paramount to ensuring that the correct software is built for the stakeholders.

Types of Requirements

- Functional Requirements
- Non-Functional Requirements
- Domain Requirements
- User Requirements

Types of Requirements

- Requirements are commonly classified as (IEEE std 830, 1998):
  - functional:
    - A requirement that specifies an action that a system MUST be able to perform, without considering physical constraints
  - Non-functional
    - A requirements that specifies system PROPERTIES, such as environment and implementation constraints, performance, dependencies, maintainability, extensibility and reliability.
    - Often classified as:
      - Performance requirements
      - External interface requirements
      - Design constraints
      - Quality attributes

Functional Requirements

- Any Requirement Which Specifies What The System Should Do.
  - Something the system must do.
- These are the requirements that the end user specifically demands as basic facilities that the system should offer.
- All these functionalities need to be necessarily incorporated into the system as a part of the contract.
- If the system does not meet a functional requirement it will fail.
- This is because it will not be able to achieve something it must do to operate properly.

Non-Functional Requirements

- Any Requirement That Specifies How The System Performs A Certain Function.
- A non-functional requirements is a measure of how well the system must do what it does.
- They specify criteria that judge the operation of a system, rather than specific behaviors.
- A non-functional requirement is essential to ensure the usability and effectiveness of the entire software system.
- for example: “Modified data in a database should be updated for all users accessing it within 2 seconds.”

Aspect: Non functional requirements

- Non functional requirements examples help to better understand what these are. Here are some examples:
  - Speed
    - how fast the system performs certain activities.
  - Availability
    - for how much of the time the system is available
    - e.g. does it operate overnight, or every day of the year, or not.
  - Capacity
    - what the limits are of what the system is able to handle.
  - Reliability
    - how dependable the system is.
  - Usability
    - how easy the system is to use for the customer or end user.

Examples: Non functional requirements

- non functional requirements examples include:
  - The time taken for a specific page to load.
  - The speed within which certain requests must be processed.
  - The level of availability the system should have.
  - Which functions can be performed at different times, and when maintenance will be carried out.
  - How many users the system can handle concurrently.

Requirements

Video Call Software

- Functional Requirements
  - Add Participants
  - Drop Participants
  - Count Participants
  - Message Participants
  - Mute Participants
  - Invite Participants
- Non-Functional Requirements
  - Video/Audio Quality
  - Reliability
  - Ease of Use
  - Cost
  - Localization
  - Availability

Requirements

Web Application

- Functional Requirements
  - Authentication of user whenever he/she logs into the system.
  - System shutdown in case of a cyber attack.
  - A Verification email is sent to user whenever he/she registers for the first time on some software system.
- Non-Functional Requirements
  - Emails should be sent with a latency of no greater than 12 hours from such an activity.
  - The processing of each request should be done within 10 seconds
  - The site should load in 3 seconds when the number of simultaneous users are > 10000

Functional Vs Non-Functional

	Functional requirements	Nonfunctional requirements
Objective	Describe what the product does	Describe how the product works
End result	Define product features	Define product properties
Focus	Focus on user requirements	Focus on user expectations
Documentation	Captured in use case	Captured as a quality attribute
Essentiality	They are mandatory	They are not mandatory, but desirable
Origin type	Usually defined by user	Usually defined by developers or other tech experts
Testing	Component, API, UI testing, etc. Tested before nonfunctional testing	Performance, usability, security testing, etc. Tested after functional testing
Types	External interface, authentication, authorization levels, business rules, etc.	Usability, reliability, scalability, performance, etc.

Domain requirements

- Domain requirements are the requirements which are characteristic of a particular category or domain of projects.
- The basic functions that a system of a specific domain must necessarily exhibit come under this category.
- For instance, in an academic software that maintains records of a school or college, the functionality of being able to access the list of faculty and list of students of each grade is a domain requirement.
- These requirements are therefore identified from that domain model and are not user specific.

User requirements

- often referred to as user needs,
  - describe what the user does with the system, such as what activities that users must be able to perform.
- You should write user requirements in natural language supplied by simple tables, forms, and intuitive diagrams.
- The requirement document shouldn't include details of the system design, and you shouldn't use any software jargon or formal notations.
- User requirements are generally signed off by the user and used as the primary input for creating system requirements.
- This is why user requirements are generally considered separately from system requirements.
- The business analyst carefully analyzes user requirements and carefully constructs and documents a set of high quality system requirements ensuring that that the requirements meet certain quality characteristics.

Elicitation and Analysis of Requirements

Elicitation and Analysis of Requirements

- Requirements elicitation and analysis is a process of interacting with customers and end-users to find out about
  - the domain requirements,
  - what services the system should provide, and
  - the other constraints.
- Specifically, requirement elicitation is requirement gathering process where gathering, researching, defining, structuring, and clarifying a software's requirements is done.
- Whereas, the requirements Analysis is about filtering the collected requirements to produce a list of actual requirements together with dependencies between them.

Requirement Elicitation

- Requirement elicitation is collecting the requirements of a system from users, customers or other stakeholders.
- It is the process of bringing out all the details required for the successful production of a software/system with catering the demand of the end-users or stakeholders in the planned efforts or in the schedule.
- Requirement Elicitation methods:
  - Interviews
  - Brainstorming Sessions
  - Use Case Approach
  - Quality Function Deployment
  - Facilitated Application Specification Technique (FAST)

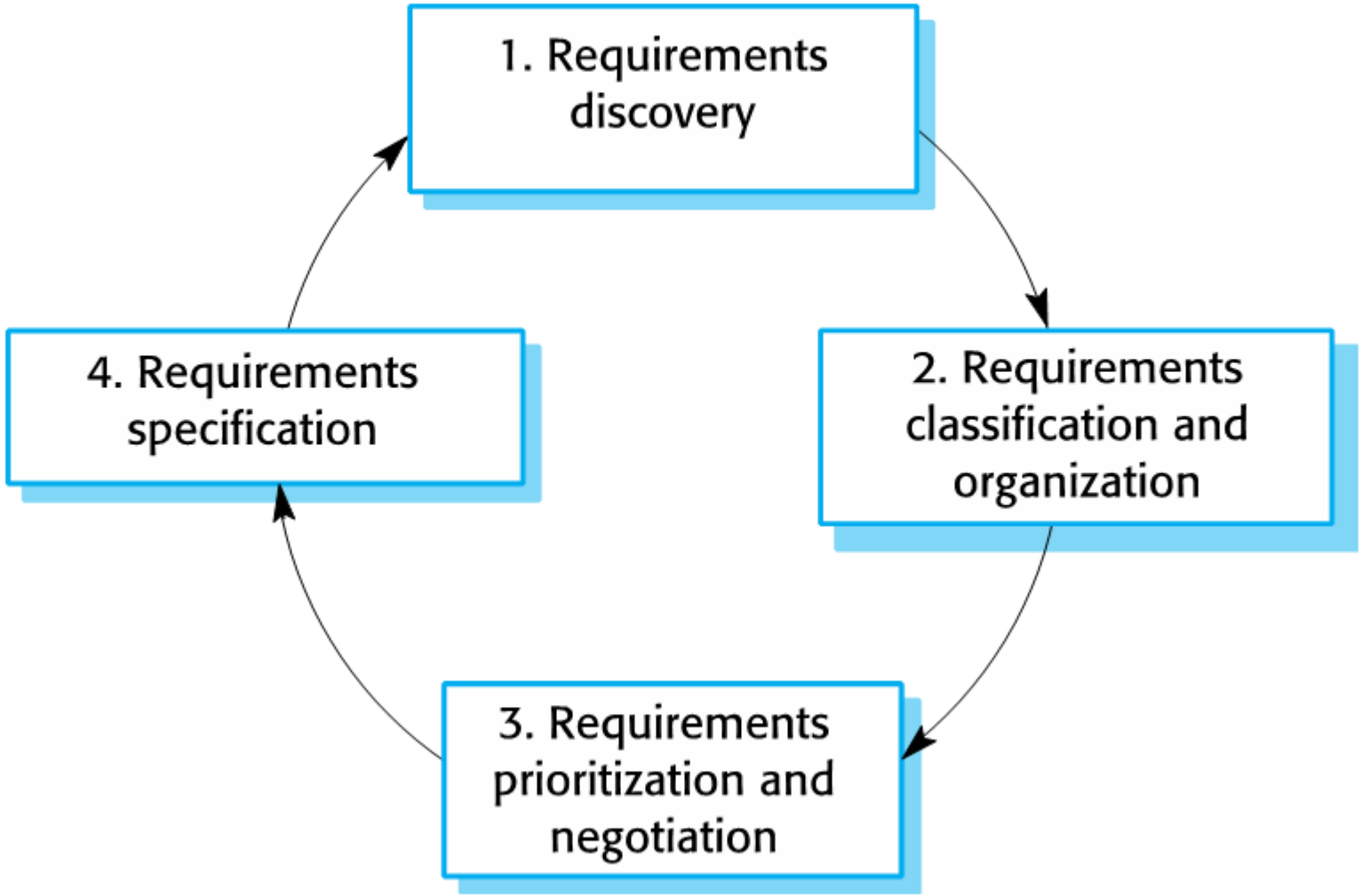
Problems in Requirement Elicitation

- Understanding large and complex system requirements is difficult.
- Undefined system boundaries.
- Customers/Stakeholders are not clear about their needs.
- Changing requirements is another issue.
- Identifying critical requirements.

Requirement Analysis

- **Software requirement analysis** simply means complete study, analyzing, describing software requirements to make consistent and unambiguous requirements.
- Requirement analysis is significant and essential activity after elicitation.
- Requirements analysis is critical to the success or failure of a systems or software project.
- Requirement analysis helps organizations to determine the actual needs of stakeholders or customers or client.
- It also enables the development team to communicate with stakeholders in a language they understand (like charts, models, flow-charts,) instead of pages of text.

Requirement Elicitation & Analysis Process



Requirement Elicitation & Analysis Process

1. **Requirements discovery**
   - Interacting with stakeholders to discover their requirements. Domain requirements are also discovered at this stage.
2. **Requirements classification and organisation**
   - Groups related requirements and organises them into coherent clusters.
3. **Prioritisation and negotiation**
   - Prioritising requirements and resolving requirements conflicts.
4. **Requirements specification**
   - Requirements are documented and input into the next round of the spiral.

Requirement Elicitation & Analysis Techniques

Requirement Elicitation & Analysis Techniques

1. Interviewing
2. View Points
3. Scenarios
4. Use-Case
5. Ethnography

Interviewing

- This is the most common technique used for requirement elicitation.
- In this technique, the interviewer directs the question to stakeholders to obtain information.
- One to one interview is the most commonly used technique.
- Types:
  - If the interviewer has a predefined set of questions then it's called **a structured interview.**
  - If the interviewer is not having any particular format or any specific questions then it's called an **unstructured interview**.
- Objective of conducting an interview is to understand the customer's expectations from the software.

Interviewing: Pros and Cons

- **Benefits:**
  - Interactive discussion with stakeholders.
  - The immediate follow-up to ensure the interviewer's understanding.
  - Encourage participation and build relationships by establishing rapport with the stakeholder.
- **Drawbacks:**
  - Time is required to plan and conduct interviews.
  - Commitment is required from all the participants.
  - Sometimes training is required to conduct effective interviews.

Brainstorming

- This technique is used to generate new ideas and find a solution for a specific issue.
- A highly trained facilitator is required to handle group bias and group conflicts.
- Every idea is documented so that everyone can see it.
- Finally, a document is prepared which consists of the list of requirements and their priority if possible.
- This session is generally conducted around the table discussion.
- All participants should be given an equal amount of time to express their ideas.

Brainstorming: Pros and Cons

- **Benefits**:
  - Creative thinking is the result of the brainstorming session.
  - Plenty of ideas in a short time.
  - Promotes equal participation.
- **Drawbacks:**
  - Participants can be involved in debating ideas.
  - There can be multiple duplicate ideas.

Scenarios

- Scenarios and user stories are real-life examples of how a system can be used.
- Stories and scenarios are a description of how a system may be used for a particular task.
- Because they are based on a practical situation, stakeholders can relate to them and can comment on their situation with respect to the story.

Viewpoints

- A viewpoint is a way of organizing the requirements for a software system, based on some perspective such as an end-user perspective or a manager's perspective.
- A key strength of viewpoint-oriented analysis is that
  - it recognizes the existence of multiple perspectives and provides a framework for discovering conflicts in the requirements proposed by different stakeholders.
- To understand the requirements for a system, we need to
  - understand a number things:
  - The services the system provides . . .
  - The application domain of the system
  - Non-functional constraints on the system and its development process
  - The environment where the system is to be installed and organizational issues affecting the system's operation.

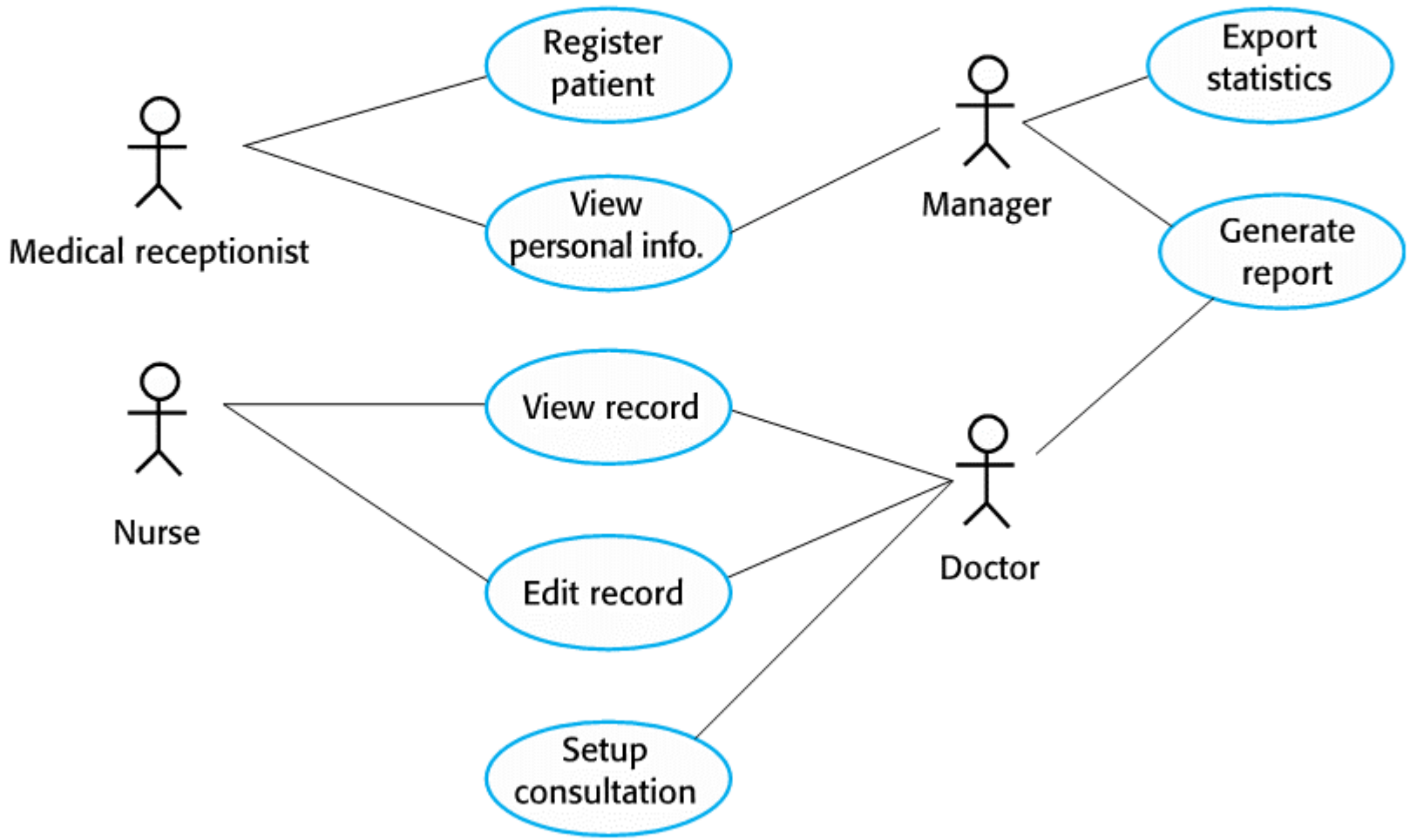
Ethnography

- Ethnography is a qualitative research method used to study people, system and environment.
- Ethnography is an observational technique that can be
  - used to understand operational processes and
  - help derive requirements for software to support these processes.
- People do not have to explain or articulate their work.
- Ethnography can provide an in-depth understanding of the socio-technological realities surrounding everyday software development practice,
  - i.e., it can help to uncover not only what practitioners do, but also why they do it.
- Ethnography is effective for understanding existing processes but cannot identify new features that should be added to a system.

Use Case Approach

- This technique combines text and pictures to provide a better understanding of the requirements.
- The use cases describe the ‘what’, of a system and not ‘how’.
- Hence, they only give a functional view of the system.
- The components of the use case design includes three major things –
  - Actor,
  - Use cases,
  - use case diagram.

USE CASE Diagram



Requirements Validation

Requirement Validation

- Concerned with demonstrating that the requirements define the system that the customer really wants.
- Requirements error costs are high so validation is very important
  - Fixing a requirements error after delivery may cost up to 100 times the cost of fixing an implementation error.
- Requirement Validation techniques
  - Requirements reviews
    - Systematic manual analysis of the requirements.
  - Prototyping
    - Using an executable model of the system to check requirements. Covered in Chapter 2.
  - Test-case generation
    - Developing tests for requirements to check testability.

Requirement Checking

- Validity
  - Does the system provide the functions which best support the customer's needs?
- Consistency
  - Are there any requirements conflicts?
- Completeness
  - Are all functions required by the customer included?
- Realism
  - Can the requirements be implemented given available budget and technology
- Verifiability
  - Can the requirements be checked?

Requirement Specification

Software Requirement Document

- A software requirements document is also known as software requirements specifications.
- It is a document that describes the intended use-case, features, and challenges of a software application.
- The requirements document should include the overview, the proposed methods and procedures, a summary of improvements, a summary of impacts, security, privacy, internal control considerations, cost considerations, and alternatives.
- Software requirement documents provide an important map of the product being built, the features that will be included.

What You Should Include in Your Software Requirements Document?

1. Introduction
   - Purpose, Intended Audience, Intended Use, Scope, Definition and Acronyms

2. Overall Description
   - Describe what you are building and for who.
   - User Needs, Assumptions and Dependencies

3. System Features and Requirements
   - Functional Requirements, External Interface Requirements, System Feature, Non-functional Requirements

SRS: Introduction

1. Purpose
   - Set the expectations for the outcome of the product.
2. Intended Audience
   - Who is the software for? Who is the end-user? Will the software be used internally at a company or externally?
3. Intended Use
   - What is the software for? What problem is it solving?
4. Scope
   - Explain the scope of the software. What are the main goals and objectives? How do they relate to the company's goals?
5. Definitions and Acronyms
   - Provide an overview of any definitions the reader should understand before reading on.

SRS: Overall Description

- Describe what you are building and for who.

1. User Needs
   - Explain the user needs for this software.
2. Assumptions and Dependencies
   - What assumptions are you making that could cause an error in your approach?
   - Is the project reliant on any other factors that could affect the development of the software?

SRS: System Features and Requirements

1. Functional Requirements
   - Take time to define the functional requirements that are essential for the software to build.
2. External Interface Requirements
   - Are there any UX and UI requirements that you must keep in mind as you build?
3. System Features
   - What features are required for the software to even work.
4. Nonfunctional Requirements
   - Are there any non-functional requirements that you need to address (i.e. budget, team, etc.)

Feasibility

Feasibility

- Study yourself
- Class Discussion and Presentation