Department of Computer Engineering

Academic Term: First Term 2023-24

Class: T.E /Computer Sem – V / Software Engineering

Practical No:	**1**
Title:	**Software Requirement Specification**
Date of Performance:	
Roll No:	**9567, 9552**
Team Members:	**Shruti Patil, Mrunal Kotambkar**

Rubrics for Evaluation:

Sr. No	Performance Indicator	Excellent	Good	Below Average	Total Score
1	On time Completion & Submission (01)	01 (On Time)	NA	00 (Not on Time)	
2	Theory Understanding(02)	02(Correct)	NA	01 (Tried)	
3	Content Quality (03)	03(All used)	02 (Partial)	01(rarely followed)	
4	Post Lab Questions (04)	04(done well)	3 (Partially Correct)	2(submitted)	

Signature of the Teacher:

Department of Computer Engineering

Academic Term: First Term 2022-23

Class: T.E /Computer Sem – V / Software Engineering

Signature of the Teacher:

Software Requirements Specification

for

E-Waste Reduction System

Fr Conceicao Rodrigues College of Engineering Computer Department

1st August 2023

1 Introduction

1.1. Purpose

The purpose of this project is to address the issue of electronic waste and its environmental impact by implementing strategies to extend the lifespan of electronic devices, promoting responsible disposal through donation and selling, and enhancing recycling infrastructure. The primary goal is to minimise electronic waste generation and promote sustainable practices in managing electronic devices. By achieving this, the project aims to contribute to the reduction of electronic waste, conserve valuable resources, and support a more environmentally-friendly approach to electronic consumption and disposal.

1.2. Document Convention

This document follows MLA Format. Bold-faced text has been used to emphasise section and subsection headings. Italicized text is used to label and recognize diagrams.

1.3. Intended Audience and Reading Suggestions

This document is to be read by the developer and the concerned staff. They might review the document to learn about the project and to understand the requirements.

Overall Description – Marketing staff have to become accustomed to the various product features in order to effectively advertise the product.

System features – Testers need an understanding of the system features to develop meaningful test cases and give useful feedback to the developers.

External Interface Requirements – The hardware developers need to know the requirements of the device they need to build. The marketing staff also needs to understand the external interface requirements to sell the product.

Nonfunctional and Functional Requirements – The hardware developers.

1.4. Product Scope

The product scope of e-waste reduction refers to the range of products, services, and initiatives aimed at minimising the generation of electronic waste (e-waste) and managing it more sustainably. E-waste includes discarded electronic devices such as computers, mobile phones, tablets, televisions, and other electronic equipment.By addressing these aspects within the product scope of e-waste reduction, companies, governments, and consumers can work

together to mitigate the environmental and health hazards associated with e-waste and promote a more sustainable electronic industry.

1.5. References

1. E-waste in India, Research Unit, RajyaSabhaSecretraiat, June, 2011.

2. E-waste : A hidden threat to Global environment and Health, Deepti Mittal.

3. M. C. Vats and S. K. Singh "Status of e-waste in India-A review" Transportation vol. 3 no. 10 2014.

4.Mainstreaming the Informal Sector in E-Waste Management – GTZ Advisory Services.

5. Managing e-waste in India – A review, Gulsan Sirkek, Gaurav Gupta.

6. Kim Peters, City farmer, Canada's office of urban agriculture. Community-based waste management for environmental management and income Generation in Low-income Areas: A case study of Nairobi, Kenya.

2 Overall Description

2.1. Product Perspective

- Customers will have a wide range of knowledge, lessons, and tips on upkeep, maintenance, and upgrading electronic gadgets. This will help to lengthen device lifespans and lessen the need for early replacements.
- Users will be able to list their working gadgets on these platforms and connect with potential buyers. This makes it easier to reuse electronics and keeps them from going into the waste stream too soon.
- Users will be able to list their working gadgets on these platforms and connect with potential buyers. This makes it easier to reuse electronics and keeps them from going into the waste stream too soon.

2.2. Product Functions

- Project that connects individuals and businesses with nearby recycling centres that accept e-waste
- individuals can donate as well as can sell their old but functional electronic devices for reuse.
- Project that showcases creative ideas and do-it-yourself (DIY) projects for repurposing or upcycling old electronic devices.

2.3. User classes and Characteristics

- Users will be able to post requests of e waste collection when generated.
- Tech giants will be able to see those requests and can connect with the customers.
- System engineers will be available to sort the queries of the users online as well as offline.
- System workers will be collecting the e waste and passing it on to the tech giants.

2.4. Operating Environment

Hardware Requirements:

1. A PC with 16GB RAM, 512 GB SSD and graphics processor Nvidia RTX 3050
2. Processor Intel i5 or i7 with 11 or 12 Generation.

Software Requirements:

1. Front-End: HTML CSS JavaScript ReactJs
2. Back-End: NodeJs ExpressJs
3. Database: MongoDB
4. Tech stack: MERN stack

2.5. Design and Implementable Constraints

- Ensuring the authenticity of buyers and sellers poses a constraint due to potential security risks.
- The system should be compatible with various platforms (Unix, Linux, Mac, Windows) and rely on PHP, JavaScript, and MySQL technologies.
- The system will run on a local host, separate from the internet, requiring robust data security measures.
- Accuracy of provided knowledge, maintenance tips, and DIY projects is crucial for user engagement and credibility.
- Adherence to e-waste disposal regulations is essential, necessitating proper documentation of electronic waste transactions.

2.6. User Documents

- The users will need documentation in order to properly understand the system and troubleshoot any potential issues that may develop in the future.
- For users to comprehend and utilise the software system successfully, user documentation for an SRS (Software Requirements Specification) document about e-waste reduction is necessary.
- Clear directions, explanations, and advice on how to use the system's many features and functionalities should be provided in this documentation.

2.7. Assumptions and Dependencies

- It is expected that all user needs have been satisfied and that the method used to design the system will be able to assist in resolving any issues that may arise.
- It is believed that the system's users will be familiar with computers and able to operate it with ease.
- The electronic waste management system will offer capabilities like the capacity to report improper garbage disposal, connect electronic waste collection businesses with renters, and connect electronic waste collection companies with recycling firms.

3 External Interface Requirements

3.1. User Interfaces

The user interface is designed to be as user-friendly as possible in every aspect. The used fonts and buttons are designed to load on web pages quickly and easily. It won't take long for the page to load because the pages will be kept light in space.

The starting page will be a centralised dashboard where users can log in and access their account information, device listings, recycling activity, and educational resources. Provides an overview of the user's contributions to reducing electronic waste and their engagement with sustainable practices.

3.2. Hardware Interfaces

- Processor: - Pentium I or above.
- RAM: - 128 MB or above.
- HD: - 20 GB or above.
- NIC: - For each party

3.3. Software Interfaces

Following are the needed requirements:

1.Operating System: Unix, Linux, Mac, Windows etc
2.Development tool: JavaScript ReactJs
3.Application: React application
4.Database: MongoDB

3.4. Communication Interfaces

The MERN stack relies on different mechanisms to handle communication between pages and to manage user sessions like RESTful APIs, JSON Web Tokens , React Hooks.

4 System Features

4.1 Donate sell purchase

Users can donate, sell and buy second hand devices from this website. This will help to reduce e waste as non-usable electronic devices one will be used by another individual. Lifespan of devices will increase and this will reduce e waste. Users will also generate a small amount of money from selling old devices.

4.2: Waste pickup

Users can upload waste pickup requests and system workers will reach out to users for collection of the e waste. The collected e waste will be handed over to system engineers & they will divide it into 2 types as repairable and non-repairable. The one which is repairable will be modified and sold on the website under service 1 and one which is non repairable would be transferred to tech giants and recycling companies for recycling and reuse.

4.3: E Waste Counselling

Users can put a counselling request on a website for their old devices and an online meet with experts will be arranged. If desired outcomes are not obtained then an offline meet will be scheduled with users and system engineers will visit users place and review the device. If the device is useful as per the need of user, then necessary modifications would be suggested and made by the system engineers else if the device doesn't satisfy users need system engineer would purchase the device from user and if modifications are possible then it would be made and sold on website to satisfy needs of other users else it would be transferred to recycling companies for recycling.

4.4: Troubleshooting

Video solutions by users and experts across the world would be posted for frequently asked questions. It will act as an open-source platform where everyone can share their ideas regarding e waste management and others can access it.

5 Other Non-Functional Requirements

5.1. Performance Requirements

- The system should provide quick responses to user actions, ensuring a smooth and seamless experience when users donate, sell, purchase, or request waste pickup.
- The system should be able to handle a growing number of users, devices, and transactions without significant degradation in performance.
- The system should support concurrent uploads of waste pickup requests and device listings, maintaining efficient performance during peak usage.
- The system should intelligently allocate server resources based on user attributes like location, ensuring low latency and fast response times for users worldwide.

5.2. Safety Requirements

- The system should employ strong user authentication methods to ensure only authorised users can access and perform actions within the system.
- User data, including personal information, payment details, and counselling sessions, must be securely stored and transmitted using encryption to protect user privacy and prevent unauthorised access.

5.3. Software Quality Attributes

- Environmental Impact: Considering the project's focus on sustainability, the system's design and operation should aim to minimise its own environmental impact, such as energy consumption and resource usage.
- Data Integrity: The system should ensure the accuracy and consistency of data across different components and interactions.
- Maintainability: The application is to be designed so that it is easily maintained. Also it should allow incorporating new requirements in any module of the system.
- Compatibility: To serve a varied user base, the user interfaces should be compatible with a range of hardware, browsers, and operating systems.

5.4. Business Rules

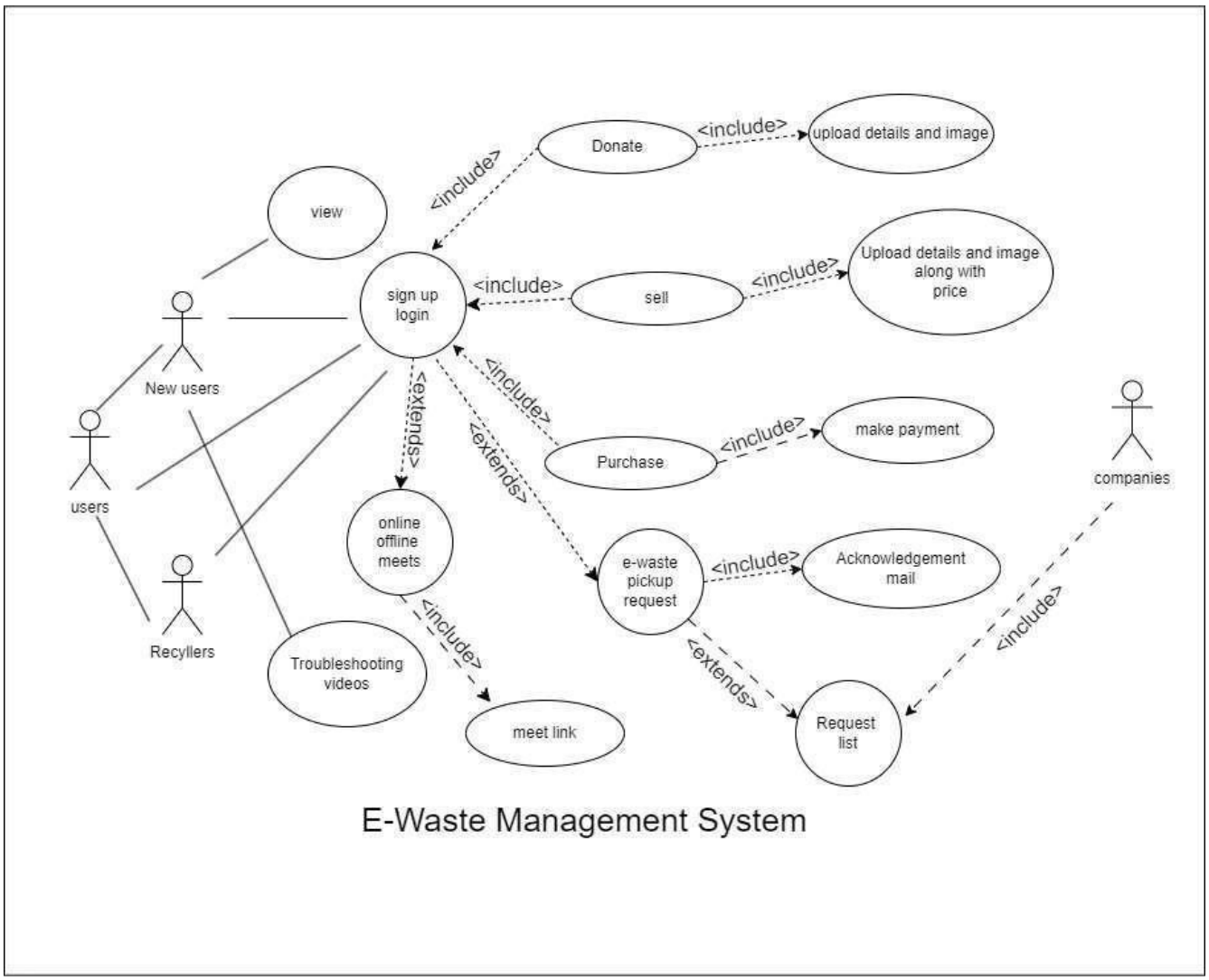
- Users must provide accurate and honest information about the condition and specifications of the devices they list for donation or sale.
- User data, including personal information, payment details, and communication, must be encrypted and securely stored.
- The platform must abide by all applicable environmental laws and guidelines for disposing of electronic waste.

Appendix A: Glossary

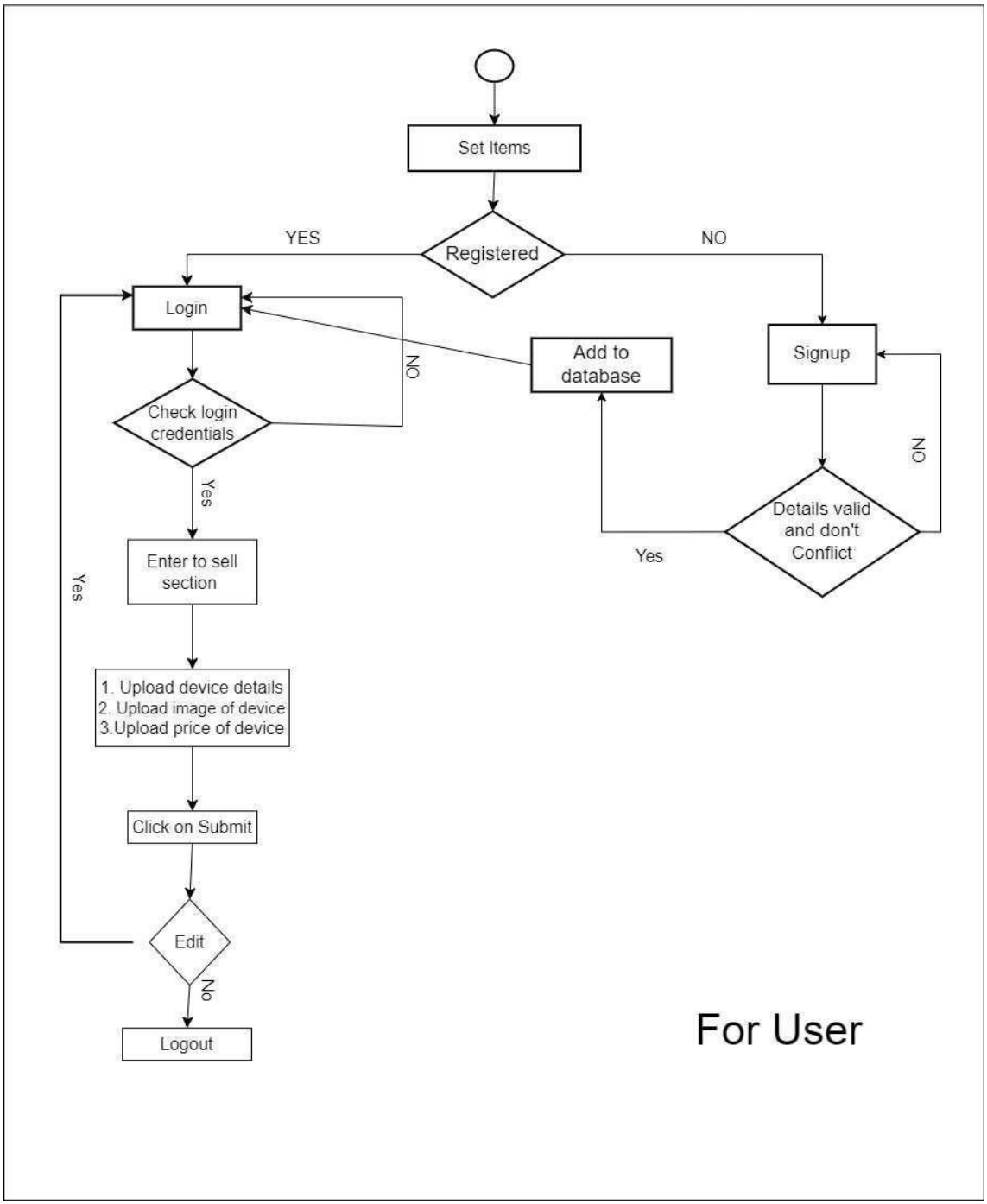
SQL	Structured Query Language
GUI	Graphical User interface
CSS	Cascading Style Sheets
JAVASCRIPT	Interactive,Dynamic,Versatile web language
EXPRESS JS	Minimal, Flexible, Robust Web Framework
RESTful APIs	Representational State Transfer

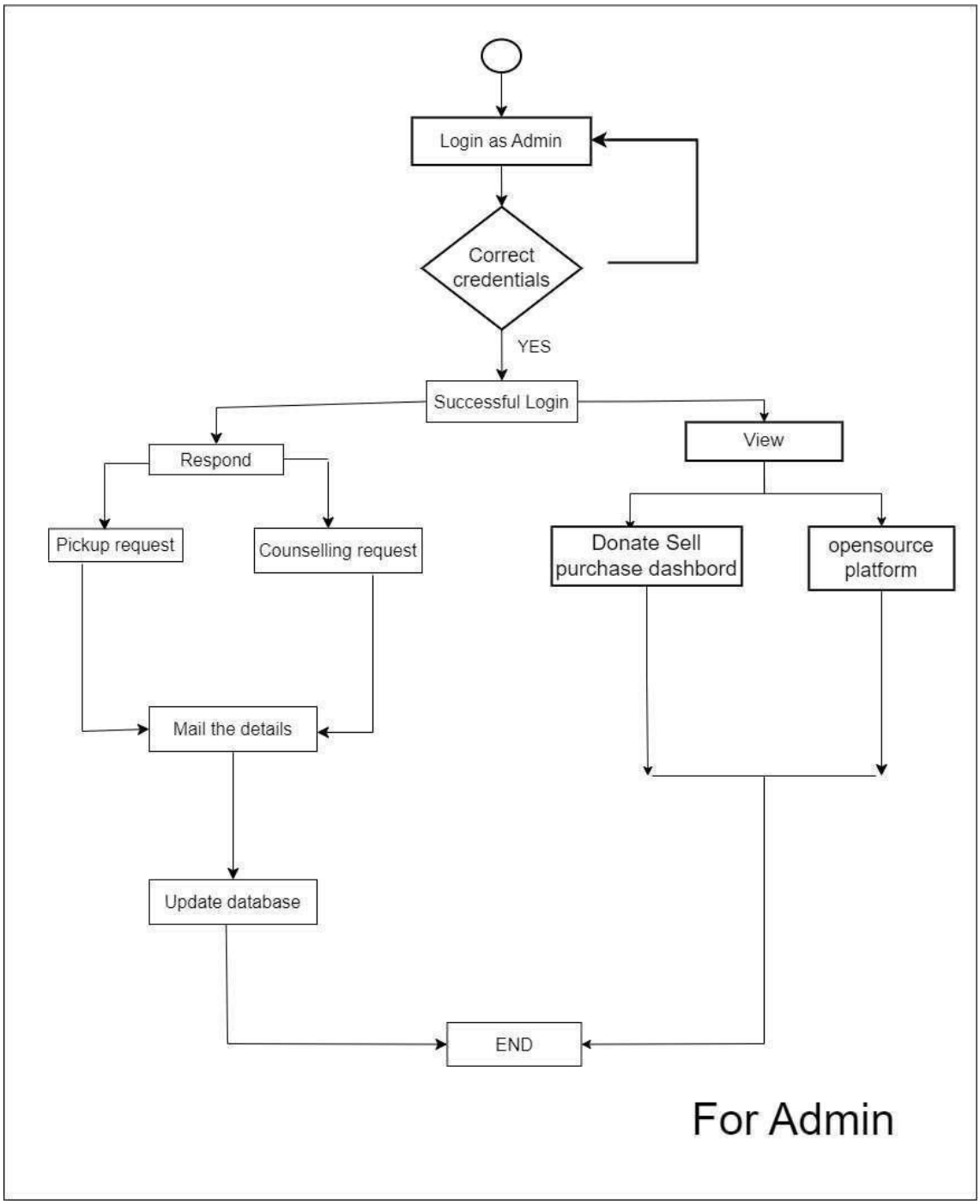
Appendix B: Analysis Models

B1: ***Use Case Diagram***



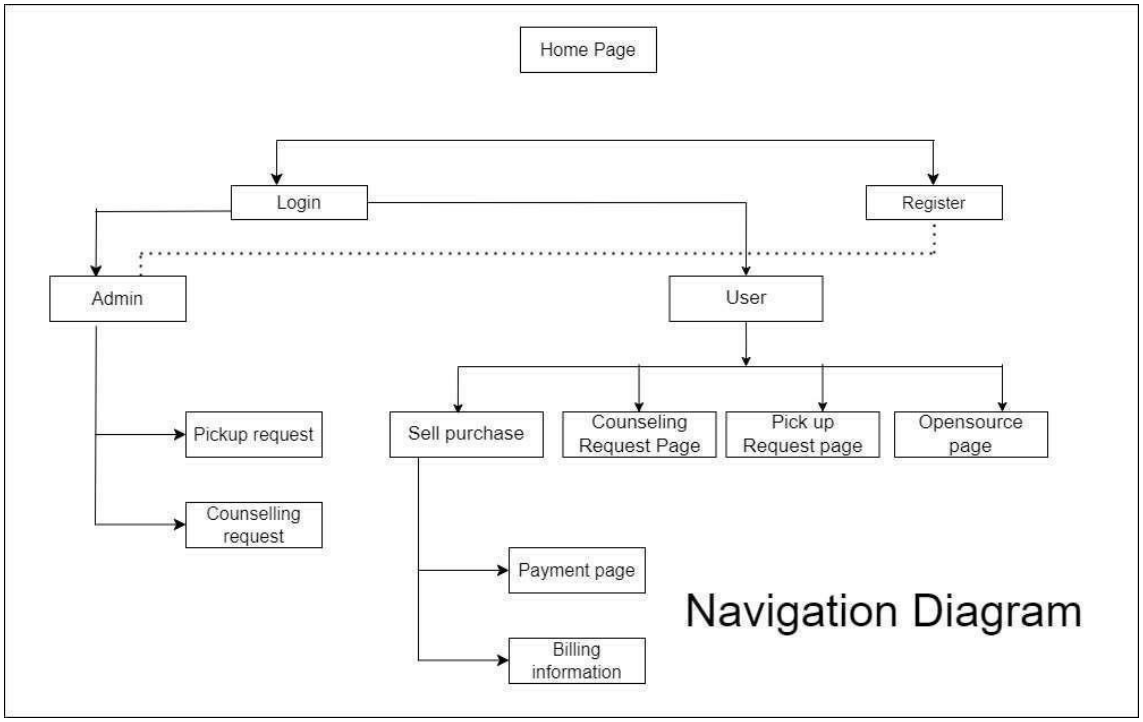
B2: ***Activity Diagram***




Login as Admin
Correct credentials
YES
Successful Login
Respond
View
Pickup request
Counselling request
Donate Sell purchase dashbord
opensource platform
Mail the details
Update database
END
For Admin

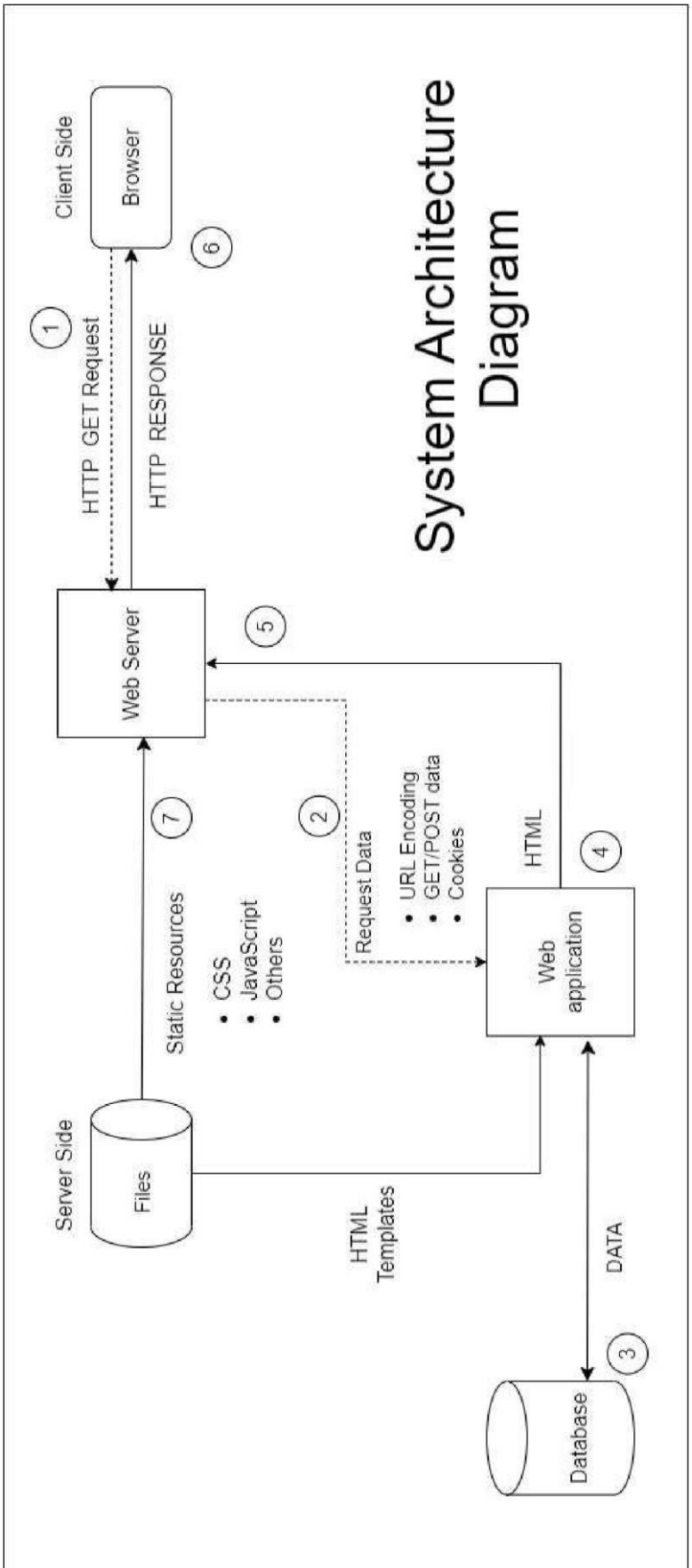
B :

3 ***Navigation Diagram***



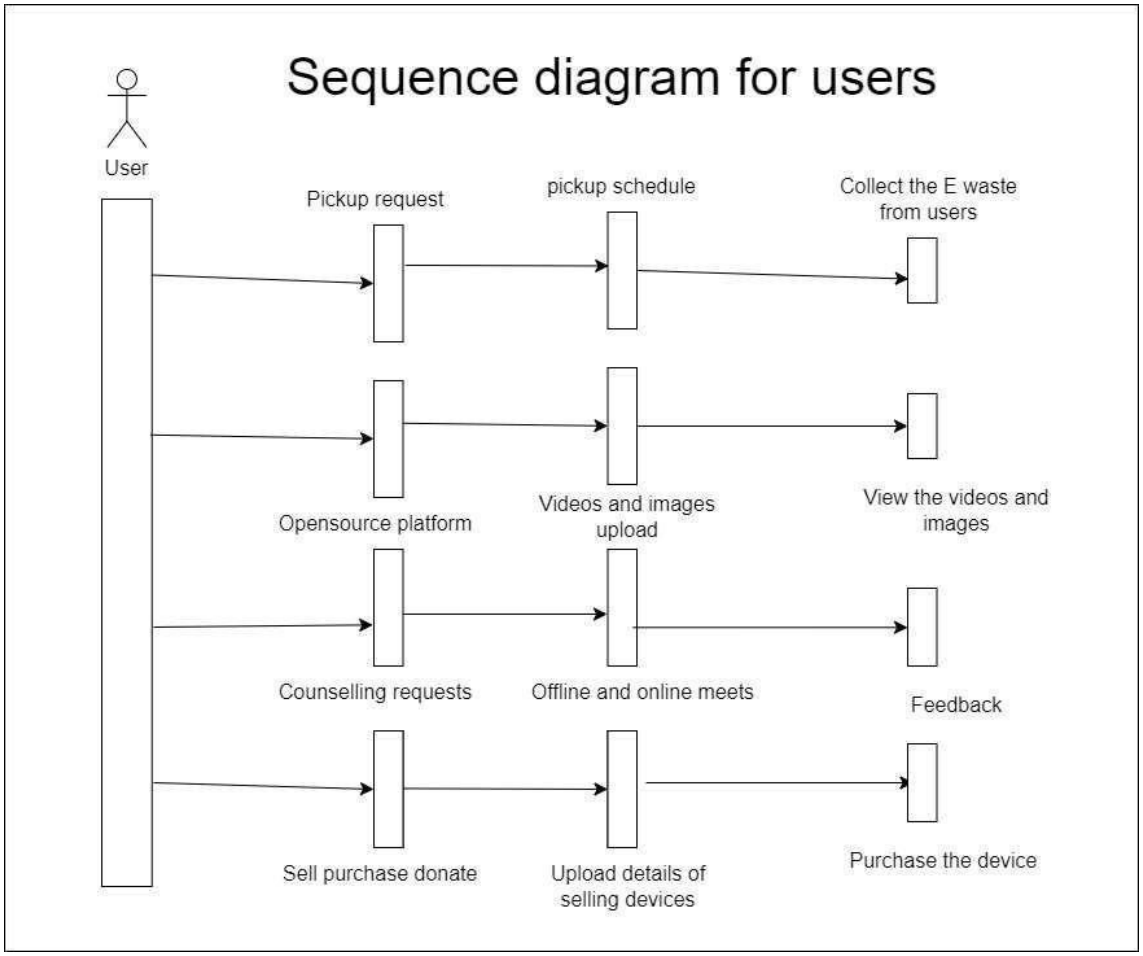
B :

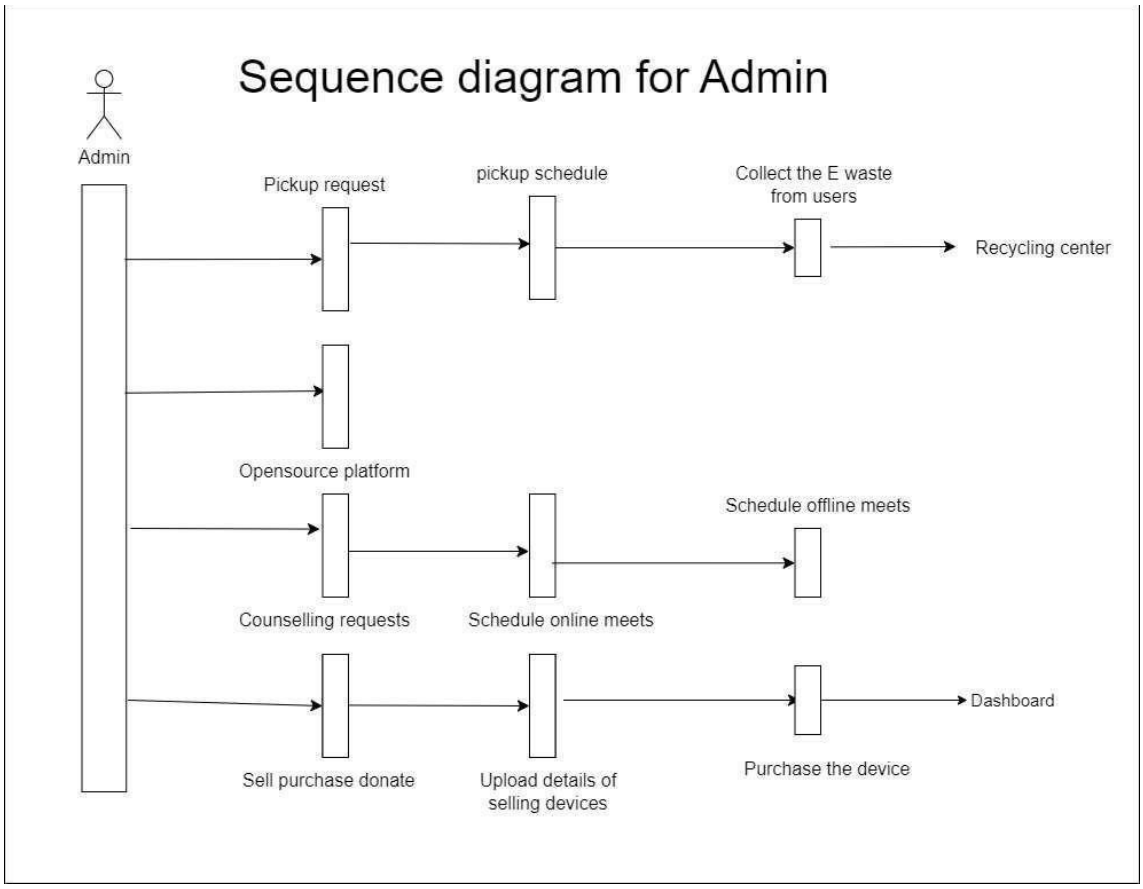
4 ***Overall System Architecture Diagram***



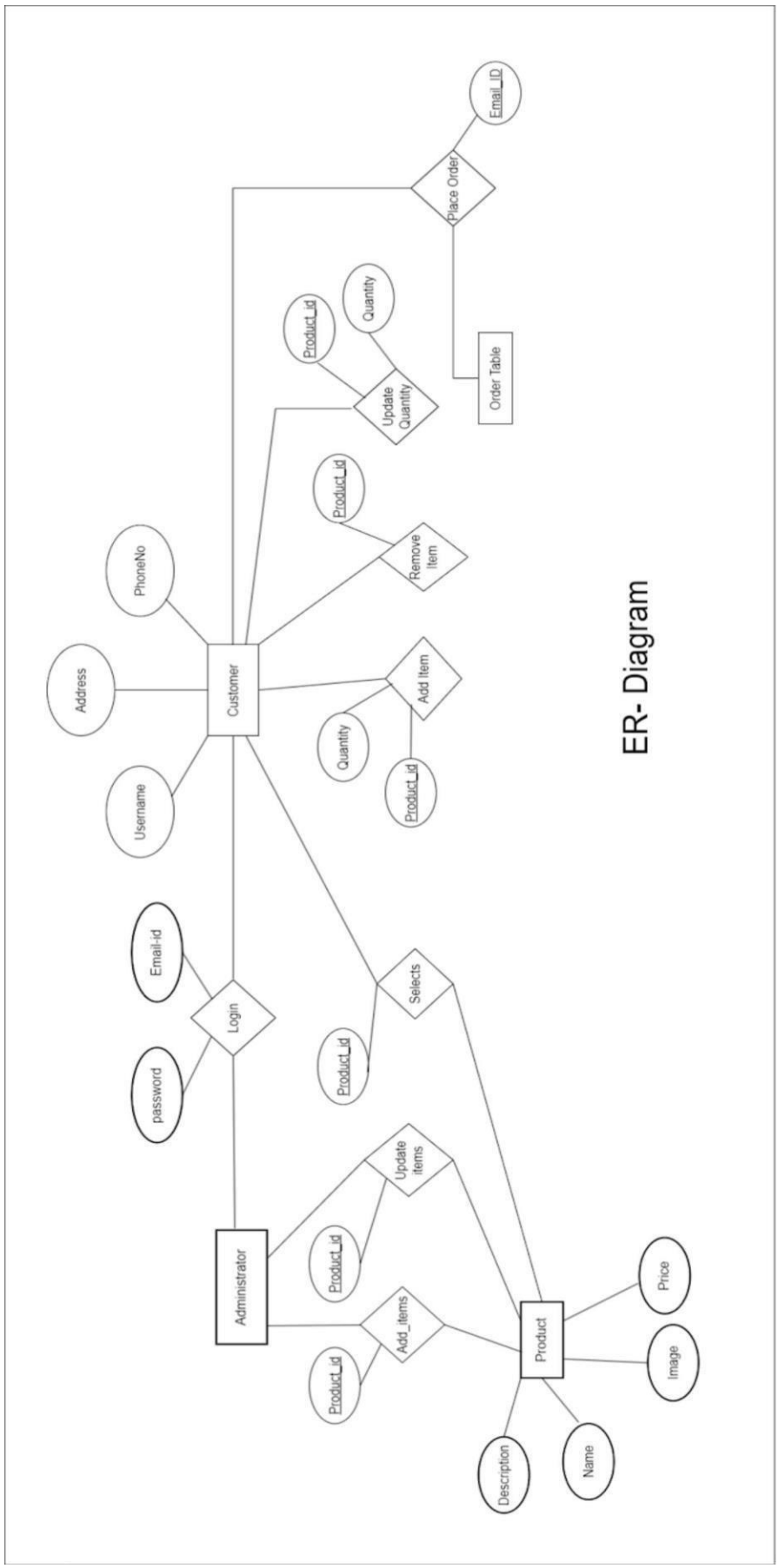
B :

5 ***Sequence Diagram***




Sequence diagram for Admin
Admin
Pickup request
pickup schedule
Collect the E waste from users
Recycling center
Opensource platform
Schedule offline meets
Counselling requests
Schedule online meets
Dashboard
Sell purchase donate
Upload details of selling devices
Purchase the device

B6: ER-Diagrams



POSTLAB:

Evaluate the importance of a well-defined Software Requirement Specification (SRS) in the softwaredevelopment lifecycle and its impact on project success.

A well-defined Software Requirement Specification (SRS) is a critical document in the software development lifecycle, and its importance cannot be overstated. It serves as a blueprint for the entire software project, providing a clear and detailed description of what needs to be developed. The significance of a well-defined SRS and its impact on project success are as follows:

Clarity and Understanding: An SRS document outlines the functional and non-functional requirements of the software. It ensures that all stakeholders, including developers, designers, testers, and clients, have a clear understanding of what the software is expected to do. This clarity reduces misunderstandings and misinterpretations, which can lead to costly changes and delays in the later stages of development.

Alignment with Client Expectations: The SRS is often developed with input from the client or end-users. This involvement ensures that the software aligns with their expectations and needs. When client expectations are met, the chances of project success and client satisfaction increase significantly.

Risk Mitigation: A well-defined SRS can help identify potential risks and challenges early in the development process. By thoroughly documenting requirements, developers can anticipate potential issues and plan for mitigation strategies, reducing the likelihood of costly surprises later in the project.

Project Planning and Estimation: The SRS provides a basis for project planning and estimation. Project managers can use the detailed requirements to estimate costs, timelines, and resource needs accurately. This helps in setting realistic project expectations and managing resources effectively.

Scope Control: An SRS clearly defines the scope of the project. Any changes or additional requirements that arise during development can be evaluated against the SRS to determine their impact on the project. This scope control prevents "scope creep," where the project continually expands without clear boundaries, which can lead to project delays and cost overruns.

Communication and Collaboration: The SRS serves as a communication tool between different stakeholders. Developers, designers, testers, and project

managers can refer to the SRS to ensure that they are on the same page. This fosters collaboration and teamwork, which is essential for project success.

Quality Assurance: With a well-defined SRS, testing teams can create comprehensive test cases and ensure that the software meets the specified requirements. This leads to a higher-quality end product with fewer defects.

Documentation for Future Reference: The SRS also serves as a valuable reference document throughout the project and after completion. It helps in maintaining and updating the software and can be used for future enhancements or audits.

Legal and Compliance Requirements: In some industries, a clear and well-documented SRS is essential for legal and compliance reasons. It can serve as evidence that the software meets the necessary regulatory standards.

Customer Satisfaction and Success: Ultimately, a well-defined SRS contributes to customer satisfaction. When the software meets or exceeds the specified requirements, clients are more likely to consider the project successful, leading to positive references, repeat business, and a good reputation in the industry.

Analyse a given SRS document to identify any ambiguities or inconsistencies and propose improvements to enhance its clarity and completeness.

Ambiguities:

Vague Terminology: Look for vague terms or phrases that lack clear definitions. For example, if the document states, "efficient e-waste disposal," it should specify what is meant by "efficient."

Unclear Requirements: Check if any requirements are ambiguous, making it difficult to understand what the software should do. Clarify these requirements by specifying inputs, expected outputs, and any constraints.

Inconsistencies:

Conflicting Requirements: Ensure there are no conflicting requirements within the document. For instance, if one section states that data should be permanently deleted, while another suggests data retention for compliance, there's an inconsistency.

Non-Functional Requirements: Examine non-functional requirements like performance, security, and scalability. Ensure these requirements are consistent with the project's overall objectives and constraints.

Completeness:

Missing Requirements: Verify that all necessary requirements are included in the SRS. Ensure that no crucial functionalities, features, or constraints have been omitted.

Use Case Coverage: Confirm that the document covers a comprehensive set of use cases or user scenarios related to e-waste management.

Clarity and Readability:

Complex Language: Simplify complex language and technical jargon to make the document more accessible to all stakeholders.

Logical Structure: Ensure that the document is logically organized with clear sections and headings.

Consistency with Project Objectives:

Assess whether the requirements align with the broader goals of e-waste management, such as sustainability, environmental compliance, or legal regulations. Requirements should not contradict or undermine these goals.

Traceability and Dependencies:

Check if there's traceability between requirements, meaning you can identify dependencies and relationships between different requirements. This helps in understanding the impact of changes.

Testability:

Ensure that the requirements are testable, meaning they can be verified and validated through testing processes. Ambiguous or non-testable requirements can lead to issues during the testing phase.

Scope and Boundary:

Clearly define the scope of the software and set boundaries to prevent scope creep. Identify what the software is responsible for and what is outside its scope.

Assumptions and Constraints:

Document any assumptions made during the requirements gathering process and the constraints that might impact the development and implementation of the software.

User Feedback: Consider seeking input and feedback from potential users and stakeholders to ensure that the SRS reflects their needs and expectations accurately.

Compare and contrast different techniques for requirement elicitation, such as interviews, surveys, and use case modelling, and determine their effectiveness in gathering user needs.

Requirement elicitation is a crucial step in the software development process, and various techniques are employed to gather user needs and define system requirements. Each technique has its strengths and weaknesses, and their effectiveness in gathering user needs can vary based on the project context and objectives. Below, I will compare and contrast three common requirement elicitation techniques: interviews, surveys, and use case modeling.

1. Interviews:

Description: Interviews involve one-on-one or group discussions between stakeholders (e.g., end-users, clients, domain experts) and the project team to collect requirements.

Advantages:

High level of interaction and direct communication allows for in-depth exploration of user needs.

Facilitates clarification of vague or ambiguous requirements in real-time.

Allows for rapport building and trust between the development team and stakeholders.

Challenges:

Resource-intensive, especially when dealing with a large number of stakeholders.

May suffer from interviewer bias or stakeholder reticence, leading to incomplete information.

2. Surveys:

Description: Surveys involve distributing questionnaires or forms to a large number of stakeholders to gather requirements and preferences.

Advantages:

Efficient for collecting data from a large and diverse audience.

Anonymity may encourage honest feedback from users who are uncomfortable in interviews.

Allows for quantitative analysis and statistical insights.

Challenges:

Limited to predefined questions, which may not capture all user needs.

Lack of real-time clarification or follow-up on responses.

Lower depth of understanding compared to interviews.

3. Use Case Modeling:

Description: Use case modeling focuses on creating scenarios or user stories that describe how the system will be used, emphasizing interactions between users and the system.

Advantages:

Provides a visual representation of system functionality and user interactions.

Encourages a user-centric perspective by focusing on how users will use the system.

Helps in identifying various paths, exceptions, and alternate flows in system usage.

Challenges:

May not capture all user needs, especially those outside the scope of specific use cases.

Requires skill and experience to create comprehensive use cases.

Less suitable for systems with complex and abstract requirements.

Effectiveness in Gathering User Needs:

Interviews: Interviews are highly effective in gathering user needs because they allow direct, interactive communication with stakeholders. They are particularly valuable when the requirements are complex, ill-defined, or subject to change. However, they can be resource-intensive and may not be suitable for large-scale data collection.

Surveys: Surveys are effective for gathering user needs when you need to collect data from a large, geographically dispersed audience. They are useful for obtaining quantitative data and opinions. However, they may lack the depth and context provided by interviews and use case modeling.

Use Case Modeling: Use case modeling is effective in capturing user needs related to system functionality and interactions. It provides a clear, visual representation of user requirements, which is valuable for system design. However, it may not address more abstract or non-functional requirements effectively.

In practice, a combination of these techniques is often used to achieve a comprehensive understanding of user needs. The choice of technique or combination thereof should be driven by project-specific factors, such as project size, complexity, budget, and the nature of the requirements.