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 (SRS) as per IEEE Format
Date of Performance:	
Roll No:	9634
Team Members:	Rianna,Aditya,Alex,Chris

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:

Lab Experiment 01

Experiment Name: Software Requirement Specification (SRS) as per IEEE Format

Objective: The objective of this lab experiment is to guide students in creating a Software Requirement Specification (SRS) document following the IEEE (Institute of Electrical and Electronics Engineers) standard format. The IEEE format ensures a structured and consistent approach to capturing software requirements, facilitating effective communication among stakeholders and streamlining the software development process.

Introduction: Software Requirement Specification (SRS) is a formal document that precisely defines the functional and non-functional requirements of a software project. The IEEE standard format provides a systematic framework for organizing the SRS, making it comprehensive, clear, and easily understandable by all parties involved in the project.

Lab Experiment Overview:

1. Introduction to IEEE Standard: The lab session begins with an overview of the IEEE standard format for SRS. Students are introduced to the various sections and components of the SRS as per the standard.
2. Selecting a Sample Project: Students are provided with a sample software project or case study for which they will create the SRS. The project should be of moderate complexity to cover essential elements of the IEEE format.
3. Requirement Elicitation and Analysis: Students conduct requirement elicitation sessions with the project stakeholders to gather relevant information. They analyze the collected requirements to ensure they are complete, unambiguous, and feasible.
4. Structuring the SRS: Using the IEEE standard guidelines, students organize the SRS document into sections such as Introduction, Overall Description, Specific Requirements, Appendices, and other relevant subsections.
5. Writing the SRS Document: In this phase, students write the SRS document, ensuring it is well structured, coherent, and adheres to the IEEE format. They include necessary diagrams, use cases, and requirements descriptions.
6. Peer Review and Feedback: Students exchange their SRS documents with their peers for review and feedback. This review session allows them to receive constructive criticism and suggestions for improvement.
7. Finalization and Submission: After incorporating the feedback received during the review session, students finalize the SRS document and submit it for assessment.

Learning Outcomes: By the end of this lab experiment, students are expected to:

- Understand the IEEE standard format for creating an SRS document.
- Develop proficiency in requirement elicitation, analysis, and documentation techniques.
- Acquire the skills to structure an SRS document following the IEEE guidelines.

- Demonstrate the ability to use diagrams, use cases, and textual descriptions to define software requirements.
- Enhance communication and collaboration skills through peer reviews and feedback sessions.

Pre-Lab Preparations: Before the lab session, students should review the IEEE standard for SRS documentation, familiarize themselves with the various sections and guidelines, and understand the importance of clear and unambiguous requirements.

Materials and Resources:

- IEEE standard for SRS documentation
- Sample software project or case study for creating the SRS
- Computers with word processing software for document preparation
- Review feedback forms for peer assessment

Conclusion: The Software Requirement Specification (SRS) lab experiment in accordance with the IEEE standard format equips students with essential skills in documenting software requirements systematically. Following the IEEE guidelines ensures that the SRS document is well-organized, comprehensive, and aligned with industry standards, facilitating seamless communication between stakeholders and software developers. Through practical hands-on experience in creating an SRS as per the IEEE format, students gain a deeper understanding of the significance of precise requirement definition in the success of software projects. Mastering the IEEE standard for SRS documents prepares students to be effective software engineers, capable of delivering high-quality software solutions that meet client expectations and industry best practices.

Hospido-An app that helps you during emergencies

1 Abstract

The outbreak of the novel coronavirus (COVID-19) has posed unprecedented challenges to healthcare systems worldwide and the largest impact was seen on the medical and healthcare community. In densely populated urban areas like Mumbai the need for efficient access to medical services and information has become paramount.Most of the doctors' usual working hours increased more than 2x during this time. A deep dive into

Hospido in shows how doctors made extraordinary contributions to meet the urgent

health needs of patients and adapted to their growing needs. Hospido is an application that makes healthcare services available to patients in need at any time. We also provide hospitals with their specialities and the respective doctors. Hospido allows you to alert top hospitals in the area about your immediate arrival through the click of a single button.It also gives online appointments to patients with respective specialists in required fields. Hospido also displays information of many hospitals like services offered, number of hospital beds, doctors on duty, etc. This app aims to bridge the gap between medical facilities and patients, ensuring quick and effective responses to healthcare needs while prioritizing safety and convenience.

2 Introduction

2.1 Purpose

1) **Convenience:** To create an application that provides hospital options to individuals in Mumbai .

2)**Self Booking**:Patients can directly find a hospital and book appointments for themselves through our app.

3)**In emergencies:** Users are provided with an emergency button to use during an emergency. This button when clicked sends a text alert to close family and friends whose numbers are saved by the user.

2.2 Scope

1. **Convenient Appointment Booking**: The app allows patients to schedule appointments with

doctors, specialists, and other healthcare providers conveniently. It can offer a user-friendly interface where patients can browse through available time slots and select the one that suits them best.

2. **Reduced Waiting Times**: By streamlining the appointment process, the app can help reduce patient waiting times. This benefits both patients, who spend less time in waiting rooms, and healthcare providers, who can manage their schedules more efficiently.
3. **Notifications and Reminders**: The app can send automated reminders to patients about their upcoming appointments, reducing the likelihood of no-shows and ensuring that patients are well-prepared for their visits.

2.3 Definitions, Acronyms, Abbreviations

Not applicable.

2.4 References

Not applicable.

2.5 Developer's Responsibilities

1. **User Authentication and Authorization**: The app should have robust user authentication mechanisms to prevent unauthorized access to patient information. Two-factor authentication (2FA) can be considered for added security.
2. **Testing and Quality Assurance**: Developers are responsible for thorough testing of the app to identify and fix any bugs or vulnerabilities. Quality assurance processes should be in place to ensure the app functions as intended and provides a seamless user experience.
3. **Accessibility**: The app should be designed with accessibility in mind, ensuring that it can be used by people with disabilities. This includes support for screen readers, adjustable font sizes, and other accessibility features.
4. **User Interface and Experience**: The developers should create an intuitive and user-friendly interface that makes it easy for patients to schedule appointments, view their medical information, and access relevant resources.
5. **Integration with Hospital Systems**: If the app is integrated with the hospital's electronic health records (EHR) system or other existing hospital systems, developers must ensure smooth and secure data exchange between the app and these systems.

3 General Description

3.1 Product Functions Overview

Emergency and Urgent Care Options: In case of emergencies or urgent medical needs, the app can provide options for immediate attention or guidance.

Appointment Rescheduling and Cancellations: Patients should be able to reschedule or cancel appointments through the app, subject to the hospital's policies.

Doctor Search and Selection: Patients can search for doctors or specialists based on their specialty, location, availability, patient reviews, and other relevant criteria. They can then select their preferred healthcare provider for the appointment.

3.2 User Characteristics

1. **Patients**: The app's primary user group comprises patients seeking medical care, appointments, and health-related information. Patients can be of various ages, backgrounds, and medical needs.
2. **Healthcare Providers**: Healthcare providers include doctors, specialists, nurses, and other medical professionals. They use the app to manage their appointment schedules, view patient information, and communicate with their patients.
3. **Administrative Staff**: The hospital's administrative staff, such as receptionists and scheduling coordinators, may also use the app to assist patients with appointment bookings and manage the healthcare provider's schedules.

3.3 General Constraints

The system should run on phones with Android with software OreO.

3.4 General Assumptions and Dependencies

Not applicable.

4 Specific Requirements

4.1 Inputs and Outputs

Not applicable

4.2 Functional Requirements

Not applicable

4.3 External Interface Requirements

Medical history of the user is required to provide help during an emergency. Contact information of users and their close peers are required to contact in case of an emergency.

4.4 Performance Constraints

Not applicable

4.5 Design Constraints

Software Constraints :

The system is to run under the Android operating system.

Hardware Constraints :

An Android phone is required to run this software

Acceptance Criteria

Before accepting the system, the developer must demonstrate that the system works on the course data for the last 4 semesters. The developer will have to show through test cases that all conditions are satisfied.

Postlab:

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

Ans:A well-defined Software Requirements Specification (SRS) is essential for the success of any software development project. The SRS serves as a blueprint for the project, and it ensures that all members working on a project are on the same page about what the software is supposed to do.

A good SRS will include the following: A description of the system's functional and nonfunctional requirements. A description of the system's environment, including the hardware and software that the system will interact with. A description of the system's interfaces, both with users and with other systems. The SRS is a living document, and it should be updated as the project progresses. However, a well-defined SRS at the start of the project can help to avoid many problems later on.
Here are some of the benefits of having a well-defined SRS:

- Improved communication and understanding between stakeholders. The SRS helps to ensure that everyone involved in the project has a clear understanding of what the software is supposed to do. This can help to avoid misunderstandings and delays.
- Reduced rework. A well-defined SRS can help to identify and resolve issues early in the development process. This can reduce the need for rework later on, which can save time and money.
- Improved quality of the software. A well-defined SRS can help to ensure that the software meets the needs of the users and the business. This can lead to a higher-quality product that is more likely to be successful.
- Increased stakeholder satisfaction. Stakeholders are more likely to be satisfied with the project if they have a clear understanding of what the software is supposed to do. This can lead to increased buy-in from stakeholders, which can help to ensure the success of the project.

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

Ans: Software Requirement Specification (SRS) - Hospido

1. Introduction: Hospido aims to create an online platform to facilitate the ease of users\patients in case of any emergencies in Mumbai which makes the medical process efficient and user-friendly.

2. Functional Requirements:

2.1 User Registration and Login:

● Ambiguity: "Users can register using their email or social media accounts."

● Improvement: Specify the required user details for registration, such as full name, email address, password, and optional social media account integration.

2.2 Search and Browse Hospital:

● Ambiguity: "Users can search for hospitals based on various filters."

● Improvement: Define the specific search filters, such as location, no. of beds available, age and cost for better clarity.

2.3 Hospital Details and Profiles:

● Inconsistency: "Each hospital profile will include the hospital’s location and speciality."

● Improvement: This information can be displayed in a clear and concise manner for the ease of a specific user.

c)Compare and contrast different techniques for requirement elicitation, such as interviews, surveys, and use case modelling, and determine their effectiveness in gathering user needs.
Ans:
This is a detailed comparison of different techniques such as interviews, surveys and use case modeling and their effectiveness in gathering user needs:
Interviews:

- Advantages: Interviews allow for a more in-depth understanding of the needs of the user. The user can provide feedback on the interviewer's understanding.
- Disadvantages: Interviews can be time-consuming and expensive, and they can be difficult to schedule.

Surveys:

- Advantages: Surveys are a cost-effective way to gather user requirements from a large group of people. They can be distributed online or in person, and they can be completed quickly and easily by the respondents.
- Disadvantages: Surveys can be difficult to design, and they may not provide enough detail about the user's requirements.

Use case modeling:

- Advantages: Use case modeling is a visual way to represent the user's requirements. It can help to identify the different types of users.
- Disadvantages: Use case modeling can be difficult to understand for non-technical users. It can also be time-consuming to create a use case model, and it may not be as detailed as an interview or survey.

Effectiveness in gathering user needs
The effectiveness of different requirement elicitation techniques in gathering user needs depends on a number of factors, including the type of software being developed, the size of the user population. In general, interviews are the most effective technique for gathering detailed user requirements. Surveys are a good option for gathering requirements from a large group of people, and use case modeling is a good way to visualize the user's requirements.
The best way to choose the right technique for requirement elicitation is to consider the specific needs of the project. If you need to gather detailed user requirements, then interviews are the best option. If you need to gather requirements from a large group of people, then surveys are a good choice. And if you need to visualize the user's requirements, then use case modeling is a good option.