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:	27 - 07 - 2023
Roll No:	9595
Team Members:	Atharva Dalvi , Reanne Dcosta , Nicole Falcao

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:

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 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.

PetKonnect—Requirements Specification Document

1 Abstract

The stray pet adoption website is a purpose-driven online platform designed to bridge the gap between homeless animals and loving forever homes. This website serves as a central hub for animal shelters, rescue organizations and potential adopters to come together and make a positive impact on the lives of stray pets. Through a user-friendly interface, the website offers an extensive database of stray animals' profiles, complete with images and detailed information on their age, breed, size, gender, health status, and personality traits. Users can search and filter through these profiles to find the perfect companion that aligns with their preferences and living conditions.

2 Introduction

2.1 Purpose

The purpose of the stray pet adoption website is to provide a compassionate and efficient online platform that connects homeless animals with caring adopters. By offering comprehensive pet listings, user-friendly adoption applications and transparent communication channels, the website aims to reduce the number of stray animals and increase pet adoption rates.Ultimately, the website's purpose is to create a positive impact on the lives of animals, finding them safe and loving forever homes while fostering a compassionate community dedicated to animal welfare.

2.2 Scope

The stray pet adoption website has a broad scope to connect potential adopters with homeless animals, facilitate adoptions and support shelters and rescue organizations. It will provide a user-friendly interface, pet listings, adoption applications, messaging, events, resources, and fundraising opportunities.

2.3 Definitions, Acronyms, Abbreviations

Not applicable.

2.4 References

[1] Raenu Kolandaisamy, Kasthuri Subaramaniam, Indraah Kolandaisamy, Lin Siew Li, “Stray Animal Mobile App”, Regional Conference on Sciences, Technology and Social Sciences, December 2016.2.5 Developer’s Responsibilities:

[2] Xu Le, Pan Younghwan, “Research on the Design of Service Process for Adoption of Stray Animals”, 人間工学 Vol.57, Supplement 2 ,2021.

[3] Kevin Horecka, Sue Neal, “Critical Problems for Research in Animal Sheltering", 1 April 2022.

[4] Peter Sandoe, Janne B.H. Jensen, Frank Jensen, Soren Saxmose Nielsen, “Shelters Reflect but Cannot Solve Underlying Problems with Relinquished and Stray Animals-A Retrospective Study of Dogs and Cats Entering and Leaving Shelters in Denmark from 2004 to 2017 ”, October 5, 2019.

[5] Dr. Manju Mittal, “Impact of Stray Animals on Public Health and Safety in Punjab”, 2019 JETIR June 2019, Volume 6, Issue 6.

2.5 Developer’s Responsibilities:

1. Front-End Development:

- Design and implement the user interface (UI) for the website.
- Ensure responsive and user-friendly design across different devices.
- Implement pet listings, search filters, and adoption application forms.

2. Back-End Development:

- Develop the server-side logic to handle user requests and data processing.
- Create databases to store pet profiles, user information, and adoption applications.
- Implement APIs for communication between the front-end and back-end

3 General Description

3.1 Website Functions Overview

The Website will function as a platform wherein general information about Pets and Stray animals in our country will be available and users can adopt a stray animal as a pet absolutely for free i.e it will connect people looking to adopt a stray as a pet and people trying to help strays to find a home. Along with NGOs, the Website will also allow users or Local people to register stray animals,especially dogs around them, so that they can help them connect the animals to potential pet parents. Various NGOs and local people seeking to help Stray animals can connect and register the animal on our Website with the little or more information they have about the same

3.2 User Characteristics

The main users of this system will be Local people looking to adopt a pet , and locals who can register stray animals on the website

3.3 General Constraints

The system is accessible using internet and a web browser

3.4 General Assumptions and Dependencies

Not applicable.

4 Specific Requirements

1. User Registration and Login:

- Allow users to create accounts with email or social media login options.
- Provide account verification and password reset mechanisms for security.

2. Pet Listings and Profiles:

- Implement a user-friendly interface to display stray pet profiles.
- Include pet details such as images, age, breed, gender, size, health status, and behavior traits.
- Allow shelters to update and manage pet listings easily.

3. Search and Filter:

- Enable users to search for pets based on various criteria (e.g., location, age, breed).
- Implement advanced filtering options to help users find specific types of pets.

4. Adoption Application:

- Offer an online form for potential adopters to apply for pet adoption.
- Include questions about the applicant's living situation, previous pet ownership experience, and commitment to pet care.

4.2 Functional Requirements

1. `Pet Matching and Adoption

- Determine suitable matches between prospective adopters and available stray pets based on preferences, such as species, age, size, and temperament.
- Ensure that no more than one adoption application is processed for the same pet simultaneously.
- Prioritize senior pets and those with longer shelter stays for adoption over newer arrivals.
-

2. Home Suitability and Capacity

- Evaluate the suitability of the adopter's home environment for the specific pet, considering factors like living space, yard size and compatibility with other pets.
- Ensure that the adopter's living space is adequate for the expected needs and behaviors

3. Avoiding Adoption Conflicts:

- Ensure that no two adopters are approved for the same pet simultaneously.
- Prevent scheduling conflicts by not allowing multiple adopters to schedule interactions with the same pet at the same time.

4. Unschedulable Pets and Reasons:

- Generate a list of pets that cannot be adopted due to specific constraints, such as compatibility issues, unsuitable living environments or inadequate resources.
- Provide clear reasons for the non-schedulability of each pet, ensuring transparency in the adoption process.

5. Data Validation and Error Handling:

- Validate input data from adopters to ensure accurate and complete information is provided.
- Verify that adopters meet minimum requirements, such as legal age and responsible pet ownership history.
- Handle errors in data submission and provide informative messages to guide adopters through the correction process.

6. Adoption Application Status Tracking:

- Provide adopters with real-time updates on the status of their adoption applications, including approval, pending review, or rejection.

7. Community Engagement and Support:

- Foster a community of adopters, volunteers, and pet lovers through discussion forums, social media integration, and educational resources.
- Offer dedicated support channels for adopters to address inquiries, concerns, and provide guidance throughout the adoption process.

4.3 External Interface Requirements

1. User Interface:

- The website should have a clean, intuitive, and user-friendly interface to enhance the user experience for potential adopters and shelter staff.
- It should be responsive and compatible with various devices (e.g., desktops, tablets, smartphones).
- It will have two user interfaces one for Adopters and the other for guardians so that locals can register as well as adopt pets.

4.4 Performance Constraints

1. Response Time and User Experience:

- The website should ensure fast loading times and responsiveness, with pages loading within 2 seconds to provide a seamless user experience.
- User interactions, such as browsing pet listings, submitting adoption applications, and scheduling virtual meet-ups, should be smooth and without noticeable delays.

2. Scalability and Concurrent Users:

- The system should be able to handle a high number of concurrent users,

especially during peak adoption periods or events, without experiencing significant performance degradation.
- The platform should scale effectively to accommodate potential increases in user traffic and adoption activities.

3. Real-time Updates and Notifications:

   Real-time updates on adoption application statuses and pet availability should be provided to users, ensuring that information remains accurate and up-to-date.

4. Security and Data Privacy:

- The platform should ensure secure data transmission and storage, maintaining the privacy of user information and adoption-related data.
- Any user interactions, such as adoption applications and scheduling, should be encrypted and protected from potential security breaches.

4.5 Design Constraints

1. Software Constraints:

- The system can run on all operating systems.

2. Hardware Constraints:

- Not applicable

3. Acceptance Criteria :

- The system should accurately match adopters with suitable pets based on preferences.
- Adopters must be able to browse listings, submit applications, and schedule interactions smoothly.
- Unschedulable pets must generate clear reasons for unschedulability.
- Timely notifications for application statuses and pet availability are crucial.
- User information must be securely handled, and error handling should provide informative feedback.

POSTLAB:

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

A well-defined Software Requirement Specification (SRS) is crucial in the software development lifecycle because it serves as the foundation for the entire project. It outlines what the software should do, how it should function, and what features it should have.

The SRS helps in the following ways:

- Clarity: It ensures that everyone involved in the project, including developers, clients, and stakeholders, have a clear understanding of what needs to be built.
- Scope: It defines the scope of the project, making it easier to manage expectations and avoid unnecessary changes later on.
- Communication: It serves as a common reference point for all parties, enabling effective communication and reducing misunderstandings.
- Planning: It allows for better planning and allocation of resources, including time, budget, and team members.
- Validation: It provides a basis for validating the software against the specified requirements, ensuring that it meets the intended goals.

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

Software Requirement Specification (SRS) - Stray Pet Adoption System

1. Introduction:

The Stray Pet Adoption System aims to create an online platform to facilitate the

adoption of stray animals. The system will connect shelters, animal rescuers, and

potential pet owners, making the adoption 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 Pets:

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

● Improvement: Define the specific search filters, such as species, breed, age, location, and size, for better clarity.

2.3 Pet Details and Profiles:

● Inconsistency: "Each pet profile will include the pet's age and birthdate."

● Improvement: Clarify whether the age and birthdate should both be displayed or provide a single field for the pet's birthdate.

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

Requirement elicitation is a crucial phase in the software development process, where the goal is to gather and understand user needs, expectations, and specifications. Various techniques can be employed for requirement elicitation, including interviews,surveys, and use case modeling. Each technique has its strengths and weaknesses, and their effectiveness in gathering user needs depends on the context, project, and stakeholders involved.

1. Interviews:

● Description: Interviews involve direct, one-on-one conversations between the software developers and stakeholders, such as users, clients, and subject matter experts.

Advantages:

● Provides in-depth information by allowing open-ended discussions.

● Enables clarifications and follow-up questions for better understanding.

● Allows for building rapport and trust with stakeholders.

Disadvantages:

● Time-consuming, especially for large groups of stakeholders.

● May be influenced by biases or misinterpretations of interviewers.

● May not represent the views of all potential users.

2. Surveys:

● Description: Surveys are questionnaires or forms distributed to a group of stakeholders to gather their opinions, preferences, and requirements.

Advantages:

● Can reach a larger number of stakeholders efficiently.

● Provides quantitative data for analysis.

Disadvantages:

● Limited to the scope and format of the questions asked.

● May lack the depth and context provided by interviews.

3. Use Case Modeling:

● Description: Use case modeling involves creating scenarios that describe how users interact with the system to achieve specific goals. Use cases define interactions between actors (users or external systems) and the system.

Advantages:

● Provides a visual representation of user-system interactions.

● Helps identify system functionalities and user roles.

Effectiveness in Gathering User Needs:

● Interviews: Interviews are highly effective for gathering in-depth and nuanced information. They allow for personalized interactions, which can reveal hidden or unanticipated requirements. However, they can be time-intensive and might not scale well for large user groups.

● Surveys: Surveys are efficient for collecting data from a larger audience, providing insights into common preferences and patterns. They are particularly useful for collecting quantitative data.