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:	24/07/20233
Roll No:	9572
Team Members:	Sanika Rozario Crystal Fernandes

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:

Case Study 1—Requirements Specification Document

1 Abstract

This is the requirements document for the case study that will be used throughout the book. The system to be developed is for scheduling the volunteers to certain NGOs, based on the input about location and proximity, event times, and time preferences of the different NGOS involved within this system. A day prior to the event, the volunteer schedules should be rolled out, indicating which volunteers fit in which slots of the event. Volunteers express interests and availability, avoiding scheduling conflicts. Different conditions have to be satisfied by the final schedule.

2 Introduction

2.1 Purpose

- The purpose of the NGO Volunteer Scheduling System is to facilitate efficient and optimized matching of volunteers to NGOs for events.
- This document outlines the external requirements for the system, including location and proximity considerations, event times, and time preferences of the NGOs and volunteers.
- By providing a user-friendly interface and advanced matching algorithms, the system aims to streamline the process of assigning volunteers to specific events, ensuring maximum volunteer participation, and enhancing the overall impact of the NGOs' activities.
- Additionally, the interfaces of the system enable NGOs and volunteers to easily interact with the platform, manage schedules, and receive timely notifications and reminders for event preparations.

2.2 Scope

- Develop a web-based platform for scheduling volunteers to specific NGOs' events.
- Efficiently match volunteers based on location, proximity, event times, and time preferences.
- Provide a user-friendly interface for NGOs and volunteers to manage profiles and preferences.
- Allow NGOs to create events and view matched volunteers.
- Enable volunteers to express interests, availability, and apply for events.
- Implement an advanced matching algorithm for optimized volunteer assignments.
- Generate schedules for each event indicating assigned volunteers and time slots.
- Send notifications and reminders to volunteers and NGOs for upcoming events.
- Allow volunteers to provide feedback and ratings for the events they participate in.
- Include an admin panel for managing user accounts and event details.
- Ensure data security and privacy measures to protect user information.
- The document will act as the authoritative reference for the system's requirements and will be used by the development team as a basis for implementation and validation.

2.3 Definitions, Acronyms, Abbreviations

NGO - Non-Governmental Organization

2.4 References

[1] Rida Naaz, Hafsa Nayeem, Syed Mohammad Athar and Shahid Bhat, "ATHWAS: THE NGO APP", Vol 8 Issue 11, November 2021

[2] Kunal Pardeshi, Tejal Shigwan, Rohit Kulkarni, Asmita Bhogade, Karan Mashal and Kaushal Komawar, "Development of Social Application to Enhance NGO and Society Collaboration", Vol 4 Issue 10, October 2015

[3] Ms. Snehal Chaudhari, Ms. Sneha Dighe, Ms. Rucha Desai, Ms. Sofiya Mulla and Mrs. Yugchhaya Dhote,"An Online Platform for Connecting NGO", Vol 6 Issue 4, April 2017

[4] Muhammad Fazril Bin Mohd Amin, "Volunteer Management System", August 2012

[5] Saraswathi Seemakuthi1, Venkat Alekhya.Siriki2 , Dr. E.Laxmi Lydia3, “A Review on Various Scheduling Algorithms”, International Journal of Scientific & Engineering Research, Volume 6, Issue 12, December-2015

[6] Aain Fathima, Dr. Smita Kavatekar, Dr. Ganesh D, "A Study on a Mobile Application for Charitable Donations" , JAIN (Deemed-To-Be University), Bengaluru

[7] Boreum Choi and Mingyung Kim (2016). A Study of the Factors Affecting Mobile Donation Application Use. International Journal of Human-Computer Interaction 32(12)

[8] Kakumani Lavanya Lathaa, Kotte Prabhakar, “NON-GOVERNMENT ORGANIZATIONS: PROBLEMS & REMEDIES IN INDIA”, Pondicherry University Puducherry, India

2.5 Developer’s Responsibilities

- Designing and developing the NGO Volunteer Scheduling System as per the specified requirements.
- Installing the software on the client's hardware or hosting environment.
- Conducting user training for NGOs, volunteers, and administrators if needed.
- Maintaining the system for one year after installation, addressing issues promptly.
- Implementing data security measures to protect user information.
- Thoroughly testing the system for functionality and quality.
- Ensuring compliance with the SRS document's specifications.

3 General Description

3.1 Product Functions Overview

- The NGO Volunteer Scheduling System efficiently matches volunteers to specific NGOs' events based on their preferences, availability, and NGOs' requirements.
- Volunteers and NGOs can register on the platform and create profiles with relevant information.
- Volunteers can express their preferences for specific NGOs or event types, while specifying their availability to avoid scheduling conflicts.
- NGOs can create events and provide event details, including time, location, and required skills.
- An advanced matching algorithm pairs volunteers with suitable NGOs, considering preferences, location, and availability.
- Schedules are generated for each event, indicating assigned volunteers for specific time slots.
- The system sends notifications and reminders to volunteers and NGOs for upcoming events and schedule changes.
- Volunteers can provide feedback and ratings for the events they participate in.

- The platform includes an admin panel for managing user accounts and event details.
- Data security and privacy measures are implemented to protect user information.

3.2 User Characteristics

- The main users of the NGO Volunteer Scheduling System are department secretaries, who possess basic computer literacy and can use programs like editors and text processors.
- The system aims to provide a user-friendly interface and reporting features to optimize volunteer engagement, event coverage, and overall efficiency in managing volunteers for NGOs' activities.
- The system is designed to accommodate users with varying levels of technological proficiency, ensuring easy interaction and navigation for all users involved in the scheduling process.
- Department secretaries and other users will have access to the platform's functionalities, enabling them to manage profiles, preferences, and event assignments efficiently.
- The system prioritizes user experience and aims to streamline the scheduling process, minimizing the need for extensive training or technical expertise for effective utilization.

3.3 General Constraints

- Operating System: The System will be compatible with various operating systems, including:
    - Windows (Windows 10, Windows 8, Windows 7, etc.)
    - Android (for the mobile app developed in Android Studio using Java)
- Data Accommodation: The system will utilize Firebase as the database to store and manage data efficiently. Firebase provides scalable cloud storage, and the system can accommodate a considerable amount of volunteer, NGO, and event-related data.
- Network Connectivity: The system should handle only online scenarios effectively. Firebase's real-time database capabilities will facilitate seamless synchronization between the app and the cloud database.
- Mobile Device Compatibility: The mobile app developed in Android Studio using Java should be compatible with a wide range of Android devices, ensuring accessibility for volunteers and NGOs using different smartphones and tablets.
- Data Protection and Privacy: Firebase's built-in security features, such as authentication and data rules, will be utilized to protect sensitive data.
- Scalability: The system should be designed with scalability in mind to accommodate potential growth in the number of volunteers, NGOs, and events over time. Firebase's scalable infrastructure will support the system's expansion as the user base increases.
- Backup and Recovery: Regular data backups and a recovery mechanism will be implemented to ensure the safety and integrity of the system's data in case of unexpected data loss or corruption.
- Device Storage: The mobile app should be designed to minimize storage usage on users' Android devices while providing necessary offline functionality.

3.4 General Assumptions and Dependencies

- Assumptions:
    1. Users have basic computer literacy and are familiar with using web-based and mobile applications.
    2. Users have access to compatible devices, including desktop computers or laptops for web access and Android smartphones or tablets for the mobile

    app.
  3. Volunteers and NGOs will provide accurate and up-to-date information during the registration process to ensure effective matching and scheduling.
  4. Users will have a stable internet connection for accessing the web platform and mobile app.
  5. The mobile app's performance will depend on the capabilities of the users' Android devices, including processing power and available memory.
- Dependencies:
  1. The successful development and deployment of the NGO Volunteer Scheduling System depend on the availability and reliability of Firebase services for data storage and synchronization.
  2. The web platform's accessibility depends on the users having a compatible web browser and internet access.
  3. The mobile app's functionality relies on the users' Android devices meeting the minimum system requirements for running the app.
  4. The system's performance may be influenced by the quality and speed of the users' internet connection, affecting data synchronization and real-time updates.

4 Specific Requirements

4.1 Inputs and Outputs

- External entities: Volunteers, NGOs, and Admin (if applicable).
- Data flows: Representing the inputs and outputs between the entities and the system.
- Processes: Representing the operations or algorithms that handle the data and generate schedules.
- Data stores (optional): Representing any data storage points, such as a database, if the system maintains historical information

Specific Requirements:

Inputs:
1. Volunteer Registration Information: Volunteers will input their personal details, location, skills, interests, and availability during the registration process.
2. NGO Registration Information: NGOs will provide their organizational details, mission, location, and event requirements during the registration process.
3. Volunteer Preferences: Volunteers can express their preferences for specific NGOs or types of events they are interested in participating in.
4. Volunteer Availability: Volunteers will update their availability for different events and time slots to avoid scheduling conflicts.
5. NGO Event Details: NGOs will input event information, including event date, time, location, required skills, and the number of volunteers needed.
6. Existing Data (Optional): The system may have access to existing data such as past volunteer participation, NGO-event history, and volunteer ratings.

Outputs:
1. Volunteer-NGO Match: The system will provide a schedule that matches volunteers to specific NGOs' events, taking into account their preferences and availability.
2. Event Schedule: For each event, the system will generate a schedule indicating which volunteers are assigned to specific time slots or tasks.
3. Volunteer Notifications: Volunteers will receive notifications about their assigned events, event details, and any schedule changes.

4. NGO Notifications: NGOs will receive notifications with volunteer assignment details and reminders for their upcoming events.
5. Volunteer Ratings and Feedback: The system may allow volunteers to provide feedback and ratings for the events they participate in.
6. Admin Reports (Optional): The admin panel may generate reports and analytics on volunteer engagement, event success, and other relevant metrics.

By capturing relevant inputs and generating accurate outputs, the NGO Volunteer Scheduling System aims to efficiently manage the scheduling process, optimize volunteer assignments, and enhance the overall experience for volunteers and NGOs alike.

Error messages. At the minimum, the following error messages are to be given:

1. Invalid Input: The system will display an error message if any required field in the Volunteer or NGO registration information is missing or filled with invalid data.
2. Conflicting Availability: When a volunteer tries to update their availability for an event and there is a scheduling conflict with another event they are already assigned to, the system will show a conflict notification.
3. No Matching Volunteers: If there are no volunteers available or interested in a particular event based on their preferences and availability, the system will notify the NGO about the unavailability of suitable volunteers.
4. Schedule Overlap: The system will alert volunteers and NGOs if there is a time slot overlap in their assigned events, allowing them to resolve the conflict.
5. Event Capacity Reached: When the number of volunteers needed for an event is already met, the system will inform other volunteers that the event is full.
6. Event Cancellation: In case of unforeseen circumstances or low volunteer participation, the system may send notifications to volunteers and NGOs about event cancellations.
7. Incorrect Volunteer Assignment: If there is any discrepancy in the volunteer-NGO match or event schedule, the system will prompt the admin to review and correct the assignment.
8. Notification Delivery Failure: If there is an issue with sending notifications, the system will display a notification delivery failure message to volunteers and NGOs.

Name	**Donation:** Volunteers can donate to charity
Input	The users bank account number
Output	Display a message whether the transaction was successful or unsuccessful
Description	1. User will enter bank account 2. User fill the amount to donate 3. Confirm donation by submit button

Name	**Search:** Volunteers can search and filter charities
Input	The search bar and/or filters
Output	Display all the NGOs that fit the criteria

Description	1. User will select the appropriate filters 2. Confirm filters by submit button 3. User will see the NGOs

Name	**Notification:** Volunteers can get notifications
Input	-
Output	Display a message the NGO wishes to send

Name	**Join event:** Volunteers can directly register by clicking join button
Description	1. User will select the NGO event they wish to join 2. User will click on Join button

Name	**Reviews:** Volunteers can leave reviews for NGOs and can find out more about a certain NGO
Description	1. Users can select the rating they want to give or select the NGO to find out more about it

Error Messages		
Invalid Input: Missing or Invalid Data	Text, Numeric	Conflicting Availability
Conflicting Availability: Scheduling Conflict	Text, Numeric	No Matching Volunteers
No Matching Volunteers: No Suitable Volunteers Found	Text, Numeric	Schedule Overlap
Event Capacity Reached: Maximum Volunteers Reached	Text, Numeric	Event Cancellation
Event Cancellation: Unforeseen Circumstances	Text, Numeric	Incorrect Volunteer Assignment
Incorrect Volunteer Assignment: Mismatch in Schedule	Text, Numeric	Notification Delivery Failure

4.2 Functional Requirements

Time and Room Allocation:
- The system will determine the time and room number for each event to avoid scheduling multiple events at the same time in the same room.
- Classroom capacity will be checked to ensure it can accommodate the expected number of volunteers for each event.

Preference for Post-Graduate Events:
- The system will prioritize post-graduate events over undergraduate events when scheduling.
- Post-graduate events will be scheduled in the order they appear in the input file, and the highest possible priority of an instructor will be considered.
- Events without specified priorities will be assigned time slots as available, and incorrect priorities will be discarded.

No Overlapping Post-Graduate Events:
- The system will ensure that no two post-graduate events are scheduled at the same time.

Unschedulable Events:
- The system will generate a list of all events that could not be scheduled due to unsatisfiable constraints.
- The reasons for unschedulability, such as conflicting time slots or unavailable rooms, will be provided.

Validating Input Data:
- The system will check the validity of data in input files against specified criteria.
- Invalid data items will be ignored, and error messages will be displayed for improper input data.

Event Notifications for NGOs:
- NGOs will receive notifications with event details, including scheduled time and room number.

Constraint Notifications for NGOs:
- NGOs will receive notifications about any constraints that were not satisfied for their events along with possible solutions.

NGO Event Scheduling Preferences:
- NGOs can specify their preferences for event scheduling, such as preferred time slots or specific rooms.

Admin Reports:
- The admin panel may generate reports and analytics on event scheduling, volunteer engagement, event success, and other relevant metrics.

4.3 External Interface Requirements

- User Command:A single user command to initiate the system.File names specified in the command line or prompted if missing.
- Input Files: Volunteers, NGOs, and event data provided via input files.
- Command-Line Prompt: Prompt for input file names if not specified.
- Error Handling: Graceful handling of missing or invalid input files.
- Output Display: Results and schedules presented in a clear format.
- Notifications: Display of event details and constraint information to NGOs.
- Interaction and Navigation: User-friendly interface with easy navigation.
- Intuitive Design: Designed for efficient user interaction.

4.4 Performance Constraints

- Input File Size: The system should handle input files containing data for up to 1000 volunteers and 100 NGOs efficiently.
- Processing Time: The system must generate event schedules and volunteer assignments within a reasonable time frame, preferably under a few seconds.

- Response Time: The user interface should provide prompt responses to user commands and interactions.
- Database Access: Database queries and updates should be optimized for quick data retrieval and storage.
- Resource Utilization: The system should utilize system resources (e.g., CPU and memory) efficiently to avoid excessive resource consumption.
- Scalability: The system should be designed to accommodate potential future increases in the number of volunteers and NGOs without significant performance degradation.
- Network Connectivity: If the system relies on internet connectivity for sending notifications, it should handle potential network delays gracefully.
- Concurrent Users: The system should support multiple concurrent users without compromising performance.

4.5 Design Constraints

Software Constraints

- Operating System Compatibility: The NGO Volunteer Scheduling System should be compatible with multiple operating systems, including Windows, macOS, and Linux, to accommodate a diverse user base.
- Mobile Application Development: The mobile app version of the system should be developed using Android Studio and Java, ensuring compatibility with Android devices.
- Database Management: The system should integrate with Firebase, a cloud-based database management system, to efficiently store and manage volunteer and NGO data.
- User Interface Design: The user interface should follow modern design principles, providing an intuitive and user-friendly experience for volunteers, NGOs, and administrators.

Hardware Constraints

- Server Infrastructure: Adequate server resources should be provisioned to handle the system's database operations, user authentication, and event scheduling tasks efficiently.
- Internet Connectivity: The system's server should have a stable internet connection to ensure seamless communication with mobile devices and web browsers.
- Mobile Device Compatibility: The mobile app should be designed to work on a wide range of Android devices with varying screen sizes, resolutions, and hardware specifications.
- Printer Compatibility: If the system includes printing functionalities for reports and analytics, it should support common printers accessible to the users.
- Mobile Device Storage: The mobile app should consider the storage limitations of users' devices, optimising data storage and minimising the app's footprint.

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) 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 for the following reasons:

1. Clarity and Precision: An SRS provides clear and precise documentation of the project's requirements. It ensures that all stakeholders have a common understanding of the software's functionalities, features, and constraints, reducing misunderstandings and ambiguities.
2. Reduced Development Errors: A comprehensive SRS minimises the chances of errors and omissions during development. It acts as a reference point for developers, helping them build the software according to the specified requirements.
3. Resource Optimization: With a detailed SRS, project resources are utilised more efficiently. Developers can focus on the specific features outlined in the document, reducing unnecessary development efforts.
4. Change Management:The SRS acts as a baseline for managing changes during the development process. Any modifications or additions to the requirements are tracked and properly documented, ensuring that the project stays on track.
5. Client-Developer Alignment: A well-defined SRS fosters better communication between clients and developers. Clients can review the requirements and provide feedback early in the development process, leading to a better alignment of expectations.
6. Time and Cost Estimation: An SRS helps in accurate time and cost estimation. Development teams can assess the complexity of the project based on the detailed requirements, leading to more accurate project planning.
7. Quality Assurance and Testing: The SRS serves as a foundation for quality assurance and testing activities. Testers can use the document to verify that the developed software meets the specified requirements.
8. Project Success: A well-structured SRS significantly impacts project success. It sets the direction for the entire development process, ensuring that the final product meets the client's needs and expectations.
9. Risk Mitigation: By detailing the project's scope and requirements, potential risks can be identified and mitigated early in the development process, reducing the likelihood of project failure.
10. Maintenance and Enhancements:An SRS serves as a reference for future maintenance and enhancements. It aids in understanding the software's original intent, making it easier to extend or modify the system.

b) Analysis of a given SRS document to identify ambiguities or inconsistencies and propose improvements:To analyze a given SRS document and identify ambiguities or inconsistencies, follow these steps:

1. Read Carefully:Read the SRS document thoroughly to understand the project's requirements and scope.
2. Check for Ambiguous Language: Look for vague or unclear statements that could lead to different interpretations. Ambiguous language can create confusion during development.
3. Look for Contradictions: Identify any conflicting statements or requirements that may cause inconsistencies in the software's functionality.

4. Validate Completeness:Ensure that all necessary requirements are included in the document. Look for any missing features or functionalities.
5. Verify Verifiability: Check if the requirements are verifiable and testable. Ambiguous or unverifiable requirements can lead to difficulties in validation.
6. Review Consistency: Check if similar functionalities are described consistently throughout the document. Inconsistency may lead to misunderstandings during development.
7. Clarify Terminology: Identify technical terms or acronyms that might not be well-defined in the document. Provide clear definitions to avoid confusion.
8. Check for Dependencies: Look for dependencies between requirements. Ensure that all interrelated functionalities are well-documented and understood.
9. Verify Feasibility:Assess the feasibility of implementing the specified requirements. Unrealistic or impractical requirements may need to be refined or modified.
10. Seek Feedback:Share the SRS document with relevant stakeholders and development teams to get their feedback. Address any concerns and incorporate improvements based on their input.

c) Comparison of different techniques for requirement elicitation:
1.Interviews: Direct interactions with stakeholders to gather requirements by asking questions and seeking clarifications.
2.Surveys: Collecting data through questionnaires to gather a large number of responses from stakeholders.
3.Workshops:Conducting collaborative sessions with stakeholders to discuss and brainstorm requirements.
4.Prototyping: Creating a basic version of the software to elicit feedback and refine requirements.
5.Observation:Observing users in their work environment to understand their needs and pain points.
6.Document Analysis: Reviewing existing documents, such as manuals or reports, to extract requirements.
7. Focus Groups:Organizing group discussions with stakeholders to capture diverse perspectives.
8.Questionnaires:Distributing standardised sets of questions to stakeholders to collect specific information.
9. Brainstorming: Encouraging open discussions and idea generation among stakeholders.