Software Requirements Specification

for

<Project>

Version <X.X>

Prepared by

Group Name: <*place your group name here*>

|  |  |  |
| --- | --- | --- |
| <name> | <student #> | <e-mail> |
| <name> | <student #> | <e-mail> |
| <name> | <student #> | <e-mail> |
| <name> | <student #> | <e-mail> |
| <name> | <student #> | <e-mail> |

|  |  |
| --- | --- |
| Instructor: | <*place your instructor’s name here>* |
| Course: | <place your course name here> |
| Lab Section: | *<place your lab section here>* |
| Teaching Assistant: | *<place your TA’s name here>* |
| Date: | <place the date of submission here> |

Contents

Contents ii

Revisions iii

1 Introduction 1

1.1 Project Purpose and Scope, and Objectives 1

1.2 Roles and responsibilities 1

1.3 Technical Assumptions and Constraints 1

1.4 Naming Conventions 1

2 Requirements 2

2.1 Functional Requirements 2

2.2 Non Functional Requirements 2

3 Other Requirements 3

4 System Architecture and Architectural Design 4

4.1 Logical View 4

4.2 Deployment View 4

4.3 Use Case View 4

5 Design and Implementation 5

6 Other Supporting Information 6

7 References 7

Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
| --- | --- | --- | --- |
| Draft Type and Number | Full Name | Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded. | 00/00/00 |

# 

*<In this template you will find text bounded by the “<>” symbols. This text appears in italics and is intended to guide you through the template and provide explanations regarding the different sections in this document. There are two types of comments in this document. These comments that are in black are intended specifically for that course. These comments that are in blue are more general and apply to any SRS. Please, make sure to delete all of the comments before submitting the document.*

*The explanations provided below, do not cover all of the material, but merely, the general nature of the information you would usually find in SRS documents. It is based on the IEEE requirements and was adapted specifically for the needs of Software Engineering courses. Most of the sections in this template are required sections, i.e. you must include them in your version of the document. Failure to do so will result in marks deductions. Optional sections will be explicitly marked as optional.*

# Introduction

*<TO DO: Please provide a brief introduction to your project and a brief overview of what the reader will find in this section. Also give some Historical detail: past to today usage area, current works in the literature>*

## Project Purpose and Scope, and Objectives

<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem. Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals.

TO DO: Write 1-2 paragraphs describing the purpose and scope of this document as explained above. Make sure to describe the benefits associated with the product >

## Roles and responsibilities

<Define organizational structure and roles.

TO DO: Please provide a list of team members and their responsibilities.>

## Technical Assumptions and Constraints

<List your tech constraint. These may include user interface style guides, standards, system requirements specifications, use case documents, or a vision and scope document.

TO DO: describe supported operating system, platform, coding language, extra tool…etc.>

## Naming Conventions

<List your naming conventions. These may include user interface style guides, standards, or coding style.

< TO DO: Describe any standards or typographical conventions that were followed when writing this SRS, such as naming that have special significance. Sometimes, it is useful to divide this section to several sections, e.g., Formatting Conventions, Naming Conventions (CamelCase …), etc.>

# Requirements

## Functional Requirements

1. Authentication Login

There will be an authentication login on the drone. Commands can only be sent when logged in with a username and password. Not everyone can do whatever they want.

1. Verified Commands

Commands will be verified according to authorizations. Only administrators can access very private data. Only group member's accounts can be admins and access this data.

1. Drone's Actions Location

If the drone's actions are constantly connected to the internet, it will be kept in the database on the memory card, if not on the server on the internet.

1. Reporting Information

As the metric information of the drone will be stored for later use, reporting them. If user commands to see specific information from this report drone will be able to give feedbacks Ex: How many smiling people drone have seen today?

1. Saving Paths

All paths the drone travels are kept in memory. In other words, the drone will have memorized the roads it has traveled. In this way, the drone does not need to be controlled and does not get lost while flying on the paths it has memorized.

## Non-Functional Requirements

### Performance Requirements

The system must be interactive, and the delays involved must be less. So, in every action-response of the system, there are no immediate delays.

In case of opening databases, sorting questions and evaluation there are no delays, and the operation is performed in less than 2 seconds for opening, sorting, computing, posting most of the files.

Also, when connecting to the server the delay is based editing on the distance of the 2 systems and the configuration between them so there is high probability that there will be or not a successful connection in less than 20 seconds for sake of good communication.

### Safety and Security Requirements

The drone should not rise above a certain height. This height limit should be considered assuming it is specified in the drone model specifications.

The security of the web interface will be strengthened and the information of users whose login information is saved will be protected.

Thanks to the rejection of special commands that did not come from admin accounts, access to the information in the drone memory will be protected.

### Software Quality Attributes

1. Adaptability

Drone’ s software and hardware will be adaptable for updates. If new functionality needs to be added, then program code will be updated. If drone needs new device to add hardware for new task, then new equipment will be added.

1. Availability

Except the times drone’s charge finished it will be able to get commands and give feedbacks in current time.

1. Testability

If new equipment added to hardware or new function added to software drone will be tested to make sure it works right.

1. Usability

Drone usage will be simplified for users to command more easier, and the web interface will be understandable to everyone.

# Other Requirements

<This section is **Optional.** Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

# System Architecture and Architectural Design

<Architectural design is concerned with understanding how a software system should be organized and designing the overall structure of that system.

*TODO: Describe the system architecture, how the application interacts with other component. Also provide a short description about software development approach*>

Ex:

The Model View Controller (MVC) design pattern is used….. Parts…..

Or

N-tier architecture is used…..

## Logical View

<The logical view is concerned with the functionality that the system provides to end-users. UML Diagrams used to represent the logical view include Class diagram, Activity diagram, State Diagram

TODO: Choose one of them, draw logical view of your system and give brief description …

## Deployment View

<*Describe the physical network and hardware configurations on which the software will be deployed. This includes at least the various physical nodes (computers, CPUs), the interaction between (sub)systems and the connections between these nodes (bus, LAN, point-to-point, messaging, etc.).*

TODO: Draw deployment view of your system and give brief description …>

## Use Case View

<A use case defines a goal-oriented set of interactions between external actors and the system under consideration.

TO DO: Provide a use case diagram which will encapsulate the entire system and all possible actors. Do not include detailed use case descriptions (these will be needed when you will be working on the Test Plan), but make sure to include a short description of what every use-case is, who are the actors in your diagram.>

### Use Case Scenarios

<A use case scenario is made up of a number of simple, discrete steps that are designated as being

performed by either the “System” or a “User”..

TO DO: Provide a brief user story explaining who is using the system and what they are trying to accomplish for each use case>

# Design and Implementation

< This section provides guidelines and information about your project

*TODO: User Interfaces, Necessary code blocks(not all of them)…>*

# Other Supporting Information

< They may include:

(a) Sample I/O formats, descriptions of cost analysis studies, results of user surveys

(b) Supporting or background information that can help the readers of the SRS

(c) A description of the problems to be solved by the software

(d) Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements

….>

# References

< Identify each document by title, report number - if applicable - date, and publishing organization.

Specify the sources from which the references can be obtained.>