

Software Requirements Specification

for

Schedule Helper (name wip)

Version <0.0>

Prepared by

Group Name: Test Team

|  |  |  |
| --- | --- | --- |
| Rawad Bader | 011678766 | rawad.bader@wsu.edu |
| Steven Arbuckle | <student #> | <e-mail> |
| Caleb Hooper | 11684910 | caleb.hooper@wsu.edu |
| <name> | <student #> | <e-mail> |
| <name> | <student #> | <e-mail> |

|  |  |
| --- | --- |
|  |  |
| Date: | <place the date of submission here> |
|  |  |
|  |  |
|  |  |

Contents

Revisions iii

1 Introduction 1

1.1 Document Purpose 1

1.2 Product Scope 1

1.3 Intended Audience and Document Overview 1

1.4 Definitions, Acronyms and Abbreviations 1

1.5 Document Conventions 1

1.6 References and Acknowledgments 2

2 Overall Description 3

2.1 Product Perspective 3

2.2 Product Functionality 3

2.3 Users and Characteristics 3

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 4

3 Specific Requirements 5

3.1 External Interface Requirements 5

3.2 Functional Requirements 6

3.3 Behaviour Requirements 6

4 Other Non-functional Requirements 7

4.1 Performance Requirements 7

4.2 Safety and Security Requirements 7

4.3 Software Quality Attributes 7

5 Other Requirements 8

Appendix A – Data Dictionary 9

Appendix B - Group Log 10

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 provide explanations and guide you through the document. There are two types of comments in this document. The comments that are in black are intended specifically for the course. The 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**.>*

# Introduction

*<TO DO: Please provide a brief introduction to your project and a brief overview of what the reader will find in this section.>*

This document is the Software Requirements Specification (SRS) for the Web Application to help people with their schedule.

This document was developed by our team for the Fundamentals of Software Engineering

course offered by the Department of Computer Science at the Washington State University Vancouver.

This document is based on IEEE Std 830-1998 [1].

This project is a tool to help in managing people schedule and published to other team members. It is more useful in a school project team and it organize people free time for meeting to work in a specific martial. In this case sharing document & data related to project the team working on it.

## Document Purpose

The purpose of this document is to present a detailed description of the Web scheduler system that helps the member of team to manage their time and availability for work or any other subject they are working on. It will explain the purpose and features of the system, the interfaces of the system, what the system will do to manage the register entries from a different people, and make sure there is empty space for each request. This development tools will make tasks easy for people trying to fit manage their schedule for study groups or meetings for school related project.

## Product Scope

This software system will be a Web scheduler system will be running on a local editor system that requires login administers and proved the code for the people only will be included in the project or work in the same job. This system will have error check system to make sure the people do not pick the similar days in the schedule, which would otherwise have to be performed manually. By adding the error checking the amount of work and effort will be minimized which make setting the schedule efficient then setting it manually.

More specifically, this system is designed to allow a student to manage and communicate with a group to manage their time accordingly and make their schedule available to view by every member of the team. The application as a mentioned before it let the person what are the spots free to add their schedule. The platform of this web application will be designed mainly in JavaScript.

## Intended Audience and Document Overview

Our audience for this document will be our Dr. Xinghui Zhao.

## Definitions, Acronyms and Abbreviations

Admin Administrator

HTML Hyper Text Markup Language

JS JavaScript

PM Project Manager

RSC Research, Development, Test and Evaluation Directorate.

SLT Software Lifecycle Tools

SRS Software Requirements Specification

TDD ﻿Test-Driven Development

UML Unified Modeling Language

## Document Conventions

<In general, this document follows the IEEE formatting requirements. Use Arial font size 11, or 12 throughout the document for text. Use italics for comments. Document text should be single spaced and maintain the 1” margins found in this template. For Section and Subsection titles please follow the template.

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

## References and Acknowledgments

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document.

TO DO: Use the standard IEEE citation guide (attached) for this section.>

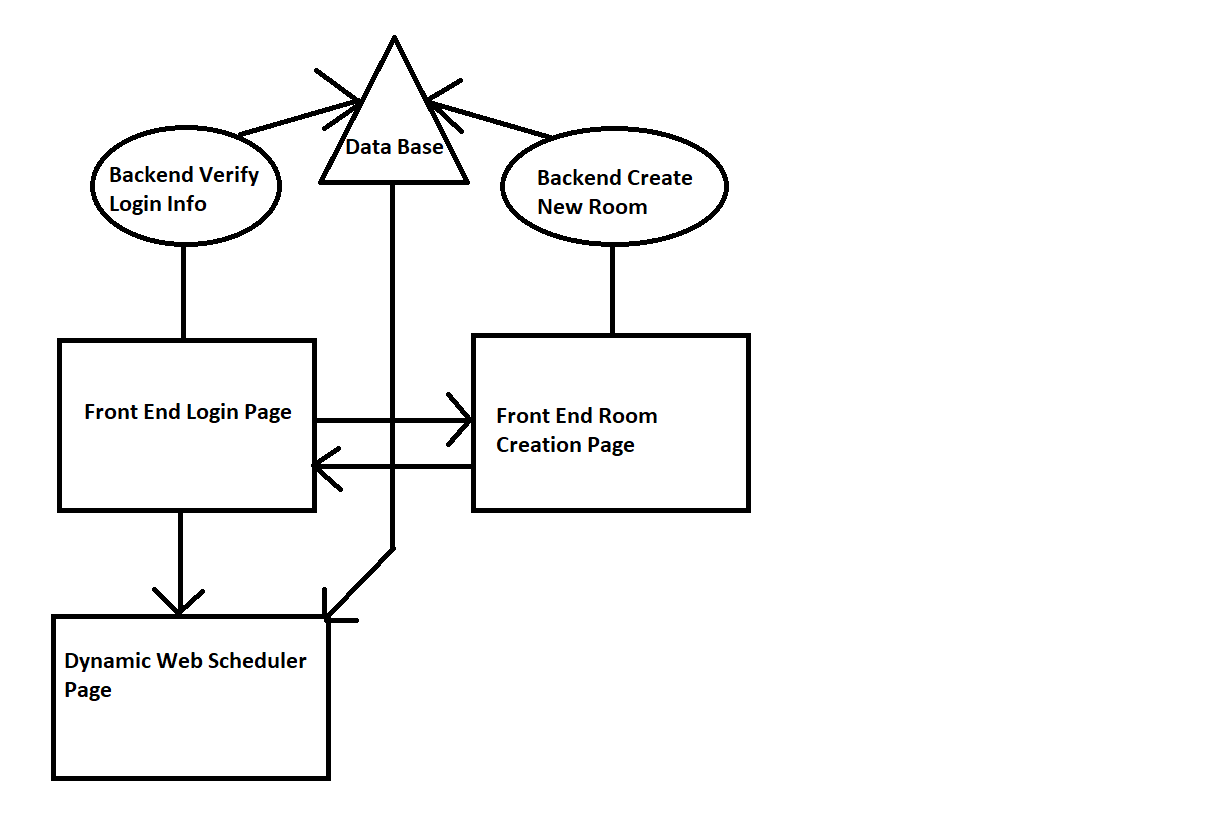
[1] IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications. This document is published by and available from The Institute of Electrical and Electronics Engineers (IEEE).

# Overall Description

## Product Perspective

The scheduler helper will be a self-contained product made from scratch. It will require the following pieces before its operational.

* Front End Login UI
* Front End Room Creation UI
* Dynamic Web Page for Actual Scheduler
* Backend or Logging In
* Backend for Creating A Room
* Database for User Info



## Product Functionality

* Submit your schedule With Excel **style** grid UI
* Display time overlaps with other members of the room and times where no one has overlaps
* Group chat log for everyone in a room.
* Login page for logging into a room
* Room creation page where you can create a new room

## Users and Characteristics

The product will be geared to anyone with a group of friends that may want to get together and do something as a group, but an additional unintended audience could be corporate office workers organizing events or gatherings.

**People with Friend Groups (MOST IMPORTANT TO STATISFY):**

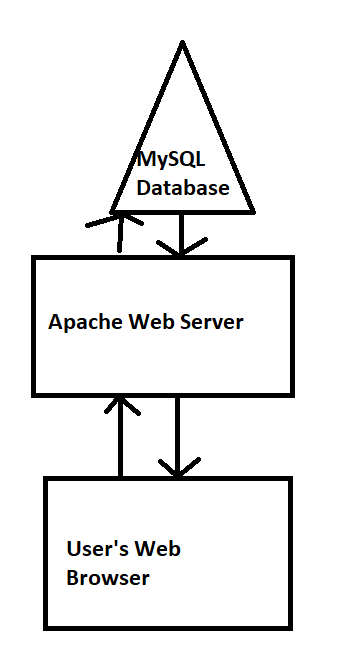
* Low to moderate technical skills
  + Product will need to be user fridley (Simple UI)
* Low on Free Time
  + If user is using a scheduler to find free time with friends, they likely have very little to begin with, so we need to reduce the amount of time spend on the scheduler as much as possible.
* Carefree Environment
  + Users will be more carefree so program could take on a looser theme or tone like other chat programs like (Discord)

**Corporate Office Worker:**

* Moderate to Advance Technical Skills
  + Product Should still be as simple as possible but user base would have fundamentals on how programs are usually laid out and work.
* Professional Environment
  + Cooperate environments are often very professional so we should refrain from covering our program with unprofessional themes or jokes and keep program a having a simple professional tone. For example (Slack).

## Operating Environment

Program will be targeted for web browser on both PC and Smart Phone Devices. The product will involve the following systems.

* MySQL database (User Info)
* Apache Web Server (Web Site Hosting)
* Web Bowser (How User Interact with Program)

## Design and Implementation Constraints

The project will be created using HTML/CSS for front end web development and we will create dynamic webpages using PHP and JavaScript. We will use MySQL as a database for storing user data and we will access the database using PHP. Hardware requirements will most likely not be a limitation on this product as the stuff we are doing is relatively low resource intensive. Security will be a feature we implement but as there is little to no important information we will be storing on our servers security is not a priority. We will create the software and provide source for project and after completion its up to users and community to maintain and add new features to the project.

## User Documentation

The program should be designed user friendly enough as to there is no need or very little need for guidance and direction. Provided documentation and help could be simple an minimum for example tool tips explaining what sections of the site do or a quick brief overview video displayed when creating a new room.

## Assumptions and Dependencies

We will be programing all parts of the website from scratch and not using old code but we will be using open source web servers and database servers. Since we are only using opensource products there will be no legal problems. We will assume that users have either a phone or a PC to access the internet which will mean we need to optimize our front-end UI design to work on all kinds of these devices. For example, the layout of the page needs to work and make intuitive sense no matter what size screen you are using on the product.

# Specific Requirements

## External Interface Requirements

### User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., Cancel) that will appear on every screen, error message display standards, and so on. Define the software components for which a user interface is needed.

TO DO: The least you can do for this section is to describe in words the different User Interfaces and the different screens that will be available to the user. Optional: You may also provide an initial Graphical User Interface design (does not have to be final).>

### Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware. You are not required to specify what protocols you will be using to communicate with the hardware, but it will be usually included in this part as well.

TO DO: Please provide a short description of the different hardware interfaces. If you will be using some special libraries to communicate with your software mention them here. In case you have more than one hardware interface divide this section into subsections.>

### Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems (Windows? Linux? Etc…), tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.

TO DO: The previous part illustrates some of the information you would usually include in this part of the SRS document. To make things simpler, you are only required to describe the specific interface with the operating system.>

### Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.

TO DO: Do not go into too much detail, but provide 1-2 paragraphs were you will outline the major communication standards. For example, if you decide to use encryption there is no need to specify the exact encryption standards, but rather, specify the fact that the data will be encrypted and name what standards you consider using. >

## Functional Requirements

*< Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. This section is the direct continuation of section 2.2 where you have specified the general functional requirements. Here, you should list in detail the different product functions with specific explanations regarding every function.*

*TO DO: Break the functional requirements to several functional areas and divide this section into subsections accordingly. Provide a detailed list of all product operations related to these functional areas.*

## Behaviour Requirements

### 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 shows 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.>

# Other Non-functional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.

TODO: Provide relevant performance requirements based on the information you collected from the client. For example you can say “1. Any transaction will not take more than 10 seconds, etc…>

## Safety and Security Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied. Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements.

TODO:

* Provide relevant safety requirements based on your interview with the client or, on your expectation for the product.
* Describe briefly what level of security is expected from this product by your client and provide a bulleted (or numbered) list of the major security requirements.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.

TODO: Use subsections (e.g., 4.3.1 Reliability, 4.3.2 Portability, etc…) provide requirements related to the different software quality attributes. Base the information you include in these subsections on the material you have learned in the class. Make sure, that you do not just write “This software shall be maintainable…” Indicate how you plan to achieve it, etc.>

# 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.>

Appendix A – Data Dictionary

*<Data dictionary is used to track all the different variables, states and functional requirements that you described in your document. Make sure to include the complete list of all constants, state variables (and their possible states), inputs and outputs in a table. In the table, include the description of these items as well as all related operations and requirements.>*

Appendix B - Group Log

<Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist the Teaching Assistant to determine the effort put forth to produce this document>