***Software Engineering***

***Software Requirements Specification***

***(SRS) Document***

**<Schooly>**

**<September 13, 2022>**

**<1.0.0>**

**<By: Andrew van Es, Evan Corsar, Nishant Sharma>**

**<Honor Code>**

Table of Contents

[1 Introduction 4](#_Toc114559575)

[1.1 Purpose 4](#_Toc114559576)

[1.2 Document conventions 4](#_Toc114559577)

[1.3 Definitions, Acronyms, and Abbreviations 4](#_Toc114559578)

[1.4 Intended audience 4](#_Toc114559579)

[1.4.1 **General Description** 5](#_Toc114559580)

[1.4.2 **Functional Requirements** 5](#_Toc114559581)

[1.4.3 **Technical Requirements** 5](#_Toc114559582)

[1.4.4 **Non-Functional Requirements** 5](#_Toc114559583)

[1.5 Project Scope 5](#_Toc114559584)

[1.6 Technology Challenges 5](#_Toc114559585)

[1.7 References 5](#_Toc114559586)

[2 General Description 5](#_Toc114559587)

[2.1 Product perspective 5](#_Toc114559588)

[2.2 Product features 6](#_Toc114559589)

[2.3 User class and characteristics 6](#_Toc114559590)

[2.4 Operating environment 6](#_Toc114559591)

[2.5 Constraints 6](#_Toc114559592)

[2.6 Assumptions and dependencies 7](#_Toc114559593)

[3 Functional Requirements 7](#_Toc114559594)

[3.1 Primary 7](#_Toc114559595)

[3.2 Secondary 8](#_Toc114559596)

[4 Technical Requirements 8](#_Toc114559597)

[4.1 Operating System & Compatibility 8](#_Toc114559598)

[4.2 Interface requirements 8](#_Toc114559599)

[4.2.1 User Interfaces 8](#_Toc114559600)

[4.2.2 Hardware Interfaces 8](#_Toc114559601)

[4.2.3 Communications Interfaces 9](#_Toc114559602)

[4.2.4 Software Interfaces 9](#_Toc114559603)

[5 Non-Functional Requirements 9](#_Toc114559604)

[5.1 Performance requirements 9](#_Toc114559605)

[5.2 Safety requirements 9](#_Toc114559606)

[5.3 Security requirements 10](#_Toc114559607)

[5.4 Software quality attributes 10](#_Toc114559608)

[5.4.1 **Availability** 10](#_Toc114559609)

[5.4.2 **Correctness** 10](#_Toc114559610)

[5.4.3 **Maintainability** 10](#_Toc114559611)

[5.4.4 **Reusability** 10](#_Toc114559612)

[5.4.5 **Portability** 10](#_Toc114559613)

[5.5 Process Requirements 10](#_Toc114559614)

[5.5.1 Development Process Used 10](#_Toc114559615)

[5.5.2 Time Constraints 11](#_Toc114559616)

[5.5.3 Cost and Delivery Date 11](#_Toc114559617)

[5.6 Other requirements 11](#_Toc114559618)

# Introduction

## Purpose

Schooly is designed to help students and instructors stay organized with their various classes. Whether that be to keep track of assignments due or stay on top of grading assignments, Schooly allows students and instructors to stay ahead with their respective courses.

## **Document conventions**

The purpose of this Software Requirements Document (SRD) is to describe the various client-views and developer-view required for Schooly. Client-orientated requirements describe the various views that a student, instructor, and administrator may observe. Developer-orientated requirements describe the system from a technical perspective. Requirements include in-depth descriptions of functionality, data and data storage, and other implementations of features.

## Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Term** | **Definition. Acronym, Abbreviation** |
| DBMS | An abbreviation for Database Management System. |
| MS | An abbreviation for Microsoft. Microsoft is a large software company which produces the software that will be used to implement ATPS. |
| UI | An abbreviation for User Interface. |

## **Intended audience**

Describe which part of the SRS document is intended for which reader. Include a list of all stakeholders of the project, developers, project managers, and users for better clarity.

### **General Description**

This section is intended for all readers to understand the basics and non-technical specifications of the project.

### **Functional Requirements**

This section is intended for developers, project managers, and other more advanced stakeholders to understand the complex and more technical terms and requirements for this project.

### **Technical Requirements**

This section is like the Functional Requirements. This is more intended for the developers and teams working on this project to understand the needs and issues that may occur during the development.

### **Non-Functional Requirements**

This section is intended for all users that need an understanding of time constraints, future plans for this project, and other legal requirements.

## **Project Scope**

Specify how the software goals align with the overall business goals and outline the benefits of the project to the business.

Schooly is designed to keep students and teachers on track and organized. Allowing students and teachers access to to-do lists, view assignments, and other various information for productivity will allow both parties to be well organized and decrease stress-levels all together.

## Technology Challenges

## References

# General Description

## **Product perspective**

Describe the context and origin of the product.

During our time at high school, we did not have access to the apps like Canvas that allowed us to view the list of assignments, our grades and other school-related assignments. We think that if we had access to a similar kind of app, we would not have missed the due dates or could have planned accordingly for upcoming tests. Therefore, we thought that designing an app that will help students keep track of their coursework can really help them succeed.

## **Product features**

A high-level summary of the functions the software would perform and the features to be included.

Schooly is designed for students and teachers alike. This is intended to keep students and teachers on track and organized throughout the semesters. Allowing students to view **to-do** **lists**, **assigned works, view their grades**, and other **school-related assignments.** Similarly, teachers will be able to view **students**, and **courses**, **grades for all students**, and **assign work.** Additionally, there will be a third party, an administrator, that will be allowed to **overwrite grades, manage teachers' assignments, and view all students' grades.**

## **User class and characteristics**

A categorization and profiling of the users the software are intended for and their classification into different user classes

Our application expects users to understand the basics of web apps but will be intuitive enough for those who have little experience with computers altogether. Our web app will be easy to follow and clearly labeled for each action a user may need to make. The important information will be displayed clearly at a glance for the user to access quickly.

## **Operating environment**

Specification of the environment the software is being designed to operate in.

This web application is being designed for any educational environment ranging from Elementary Schools to College Campuses. The application could also be used within the home to help students finish their school assignments on time.

## Constraints

Any limiting factors that would pose a challenge to the development of the software. These include both design as well as implementation constraints.

This application is meant for students and teachers alike, which could pose a problem as the computer literacy skills of some of the users could lead to some issues if the interfaces of the application are too confusing. In turn, the application should be easy to both navigate and simple to use.

## Assumptions and dependencies

A list of all assumptions that you have made regarding the software product and the environment along with any external dependencies which may affect the project

As of now, we are planning to use the Spring framework which is an excellent open-source application framework that provides infrastructure support for developing a Java application.

Vaadin is one of the dependencies that we might use but we are still at an early stage, so we need to explore more. Since we do not have much knowledge in JS, HTML, and CSS, this is an excellent framework that can assist us in writing the UI.

Thymeleaf is another dependency that we might use that will help us design our HTML files.

# Functional Requirements

Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.

## Primary

All the requirements within the system or sub-system in order to determine the output that the software is expected to give in relation to the given input. These consist of the design requirements, graphics requirements, operating system requirements and constraints if any.

* FR0: The system will allow the admin to create or delete instructors and students.
* FR1: The system will allow the admin to lookup for the lists of instructors and students along their associated courses.
* FR2: The system will allow a course instructor to create or delete assignments and tests.
* FR3: The system will allow students to enroll into a course.
* FR4: The system will allow students to view the assignments and upcoming test dates for a course.
* FR5: The system will allow students to mark an assignment as complete or important or incomplete.

## Secondary

Some functions that are used to support the primary requirements.

* FR0: The system will store created users’ information in a database
* FR1: The system will store created assignments and grades in a database

# Technical Requirements

## Operating System & Compatibility

* Compatible with most modern browsers (Mozilla, Chrome, Microsoft Edge, Safari).
* Compatible with most modern operating systems that can run the above browsers.

## Interface requirements

### User Interfaces

The UI will allow the user to sign in and out. It will allow users to click through various tabs to view different information and screens. For students, there will be grades, to-do, and course tabs; for Instructors, there will be similar tabs except for to-do being replaced with a tab to create assignments. The ‘grades’ tab for instructors will be to view all grades of all students. Administrators will have access to all grades of all students, edit and manage instructors and courses, and be allowed to remove or add students to a course.

The logic behind the interactions between the users and the software. This includes the sample screen layout, buttons and functions that would appear on every screen, messages to be displayed on each screen and the style guides to be used.

### Hardware Interfaces

* Any Microsoft Windows OS that can run a current browser software
* Any Apple OS that can run a current browser software
* A stable connection to the internet
* HTTP will be used to access the web-application

All the hardware-software interactions with the list of supported devices on which the software is intended to run on, the network requirements along with the list of communication protocols to be used.

### Communications Interfaces

Determination of all the communication standards to be utilized by the software as a part of the project

### Software Interfaces

* **Spring Framework** will be used for the backend framework in developing the Java web application.
* **Vaadin** is a dependency that will be used as a framework to assist in designing the UI.
* **Thymeleaf** is a resource used for frontend development.
* **MySQL** will likely be used as the DBMS.

The interaction of the software to be developed with other software components such as frontend and the backend framework to the used, the database management system and libraries describing the need and the purpose behind each of them.

# Non-Functional Requirements

Constraints on the services or functions offered by the system (e.g., timing constraints, constraints on the development process, standards, etc.). Often apply to the system rather than individual features or services.

## Performance requirements

* NFR1(R): The system will consume no more than 100MB of memory and storage.
* NFR2(R): The novice user (instructor, administrator, or student) should be able to traverse the web application easily and with few problems. They will be able to become comfortable in less than a week of use.
* NFR3(R): The expert user will be able to quickly check grades/grade assignments and feel very comfortable with the layout.

## Safety requirements

* The implementation of regular system Backups to prevent loss/manipulation of data.
* Implementation of features that allow users to undo certain types of actions (Within Reason).
* Implementation of Administrative overwriting privileges.

*List out any safeguards that need to be incorporated as a measure against any possible harm the use of the software application may cause.*

## Security requirements

* NFR5(R): Administrators and instructors will be required to have a strong password
* NFR6(R): Multi-factor authentication will be required for all users to ensure the security of grades, assignments, and other valuable data.

## Software quality attributes

### **Availability**

This web-application will be available for all schooling system that may need it. Students, teachers, and administrators will have access to their respective views and information.

### **Correctness**

User testing and review will be undergone to ensure correctness of this web application.

### **Maintainability**

As part of user testing, bugs that could be fatal will be corrected efficiently. In cases that the web-app may fail, we will have a maintainer monitoring the system and servers.

### **Reusability**

Schooly will be able to be reused throughout various schools and university settings.

### **Portability**

To continue to ensure correctness, this web-application will be ported to only laptops / personal computers. There will be room to expand into the mobile / tablet realm at a later date.

Detailing on the additional qualities that need to be incorporated within the software like maintainability, adaptability, flexibility, usability, reliability, portability etc.

## Process Requirements

### Development Process Used

Incremental Development Model (Agile)

### Time Constraints

Time is a limited factor that is determined by the final product requiring to be done by November 15 - November 22.

### Cost and Delivery Date

Cost estimate is $0 and a delivery date of November 15 - 22, 2022.

## Other requirements

* NFR4(R): The system will conform to FERPA guidelines to maintain student privacy.
* NFR7(R): Update ability to be ported to mobile devices.

All SRS/SRD should be:

● **Correct:** A method of analysis that ensures that the software meets the requirements identified.

● **Unambiguous:** There is only one interpretation of what the software will be used for, and it is communicated in a common language.

● **Complete:** There is a representation for all requirements for functionality, performance, design constraints, attributes, or external interfaces.

● **Consistent:** Must be in agreement with other documentation, including a systems requirements specification and other documents.

● **Ranked for Importance and/or Stability:** Since all requirements are not of equal weight, you should employ a method to appropriately rank requirements.

● **Verifiable:** Use measurable elements and defined terminology to avoid ambiguity.

● **Modifiable:** A well-defined organizational structure of the SRS document that avoids redundancies can allow easy adaptation.

● **Traceable:** Ability to trace back to the origin of development and move forward to the documents produced from the SRS.

● **Legible and Professionally Presented**: Must use a consistent font and style. Must have proper formatting of tables and charts. Must be grammatically correct. Use active tense and concise sentences.