***Software Engineering***

***Software Requirements Specification***

***(SRS) Document***

**[Paper Path]**

**[9/26/2023]**

**[Version 1]**

**By: [Christopher Hochrein, Philip Sijerkovic, Amin Abdelrahim]**

**[Honor Code]**

**Table of Contents**

[1. Introduction 3](#_Toc126197756)

[1.1. Purpose 3](#_Toc126197757)

[1.2. Document Conventions 3](#_Toc126197758)

[1.3. Definitions, Acronyms, and Abbreviations 3](#_Toc126197759)

[1.4. Intended Audience 4](#_Toc126197760)

[1.5. Project Scope 4](#_Toc126197761)

[1.6. Technology Challenges 4](#_Toc126197762)

[1.7. References 4](#_Toc126197763)

[2. General Description 4](#_Toc126197764)

[2.1. Product Perspective 4](#_Toc126197765)

[2.2. Product Features 4](#_Toc126197766)

[2.3. User Class and Characteristics 4](#_Toc126197767)

[2.4. Operating Environment 4](#_Toc126197768)

[2.5. Constraints 5](#_Toc126197769)

[2.6. Assumptions and Dependencies 5](#_Toc126197770)

[3. Functional Requirements 5](#_Toc126197771)

[3.1. Primary 5](#_Toc126197772)

[3.2. Secondary 5](#_Toc126197773)

[4. Technical Requirements 5](#_Toc126197774)

[4.1. Operating System and Compatibility 5](#_Toc126197779)

[4.2. Interface Requirements 5](#_Toc126197780)

[4.2.1. User Interfaces 5](#_Toc126197781)

[4.2.2. Hardware Interfaces 6](#_Toc126197782)

[4.2.3. Communications Interfaces 6](#_Toc126197783)

[4.2.4. Software Interfaces 6](#_Toc126197784)

[5. Non-Functional Requirements 6](#_Toc126197785)

[5.1. Performance Requirements 6](#_Toc126197786)

[5.2. Safety Requirements 6](#_Toc126197787)

[5.3. Security Requirements 6](#_Toc126197788)

[5.4. Software Quality Attributes 6](#_Toc126197789)

[5.4.1. Availability 6](#_Toc126197790)

[5.4.2. Correctness 6](#_Toc126197791)

[5.4.3. Maintainability 6](#_Toc126197792)

[5.4.4. Reusability 6](#_Toc126197793)

[5.4.5. Portability 6](#_Toc126197794)

[5.5. Process Requirements 6](#_Toc126197795)

[5.5.1. Development Process Used 6](#_Toc126197796)

[5.5.2. Time Constraints 6](#_Toc126197797)

[5.5.3. Cost and Delivery Date 7](#_Toc126197798)

[5.6. Other Requirements 7](#_Toc126197799)

[5.7. Use-Case Model Diagram 7](#_Toc126197800)

[5.8. Use-Case Model Descriptions 7](#_Toc126197801)

[5.8.1. Actor: Editor (Responsible Team Member) 7](#_Toc126197802)

[5.8.2. Actor: Writer (Responsible Team Member) 7](#_Toc126197803)

[5.8.3. Actor: Photographer (Chris) 8](#_Toc126197804)

[5.9. Use-Case Model Scenarios 8](#_Toc126197805)

[5.9.1. Actor: Editor (Responsible Team Member) 8](#_Toc126197806)

[5.9.2. Actor: Writer (Responsible Team Member) 9](#_Toc126197807)

[5.9.3. Actor: Photographer (Chris) 9](#_Toc126197808)

# Introduction

## Purpose

The goal of Paper Path is to make it easy for a newspaper team to complete projects by enabling effective communication between project members and allowing management of project materials.

## Document Conventions

The purpose of this Software Requirements Document (SRD) is to describe the requirements for the software, and how it will look for end-users. This includes a description of the different actors and their roles. This document will also list out the system requirements to run our application.

The purpose of this Software Requirements Document (SRD) is to describe the client-view and developer-view requirements for the Automated Police Ticketing System (APTS). Client-oriented requirements describe the system from the client’s perspective. These requirements include a description of the different types of users served by the system. Developer-oriented requirements describe the system from a software developer’s perspective. These requirements include a detailed description of functional, data, performance, and other important requirements.

## Definitions, Acronyms, and Abbreviations

[Include any specialized terminology dictated by the application area or the product area.

For example:]

|  |  |
| --- | --- |
| Java | A programming language originally developed by James Gosling at Sun Microsystems. We will be using this language to build the Restaurant Manager. |
| MySQL | Open-source relational database management system. |
| .HTML | Hypertext Markup Language. This is the code that will be used to structure and design the web application and its content. |
| SpringBoot | An open-source Java-based framework used to create a micro Service. This will be used to create and run our application. |
| MVC | Model-View-Controller. This is the architectural pattern that will be used to implement our system. |
| Spring Web | Will be used to build our web application by using Spring MVC. This is one of the dependencies of our system. |
| Thymeleaf | A modern server-side Java template engine for our web environment. This is one of the dependencies of our system. |
| NetBeans | An integrated development environment (IDE) for Java. This is where our system will be created. |
| API | Application Programming Interface. This will be used to implement a function within the software where the current date and time is displayed on the homepage. |

## 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.]

The intended audience are writers, editors, and photographers.

## Project Scope

The goal of this software is to make it easy for any team behind a newspaper to complete drafts in a expeditious manner. This is in line with the goals of a newspaper business as speedy delivery of content is vital for an industry that moves as quickly as newspaper production.

Using this software will:

* Allow easy management of multiple versions of drafts and photos related to said drafts.
* Enable effective communication between editors and writers.
* Make it easier for writers and editors to communicate with photographers and get photos related to their current work.

## Technology Challenge

*Leave 1.6 blank for now.*

## References

*Leave 1.7 blank for now.*

# General Description

## Product Perspective

Paper Path was originally created out of the desire to make it easier to work on group projects together.

## Product Features

The application will allow employees to create different accounts depending on their role in the team. Writers are able to upload drafts for editors to view and edit. Once editors have finished editing a draft, they can then send it back to writers for revision, or they can mark the draft as completed. Writers and editors can also make requests for photos that will be uploaded to a bulletin that all photographers in the system can see and access. Photographers can select and respond to requests by uploading photos, which can then be accessed by the parties that made the requests.

## User Class and Characteristics

Our application is designed for use by a newspaper writing team. Users should know the basics of using a computer, but the system should be simple and intuitive. The system should allow them to fulfill tasks they were already doing in a more streamlined and accessible way.

## Operating Environment

The application will be designed to operate via the web on desktop operating systems.

## Constraints

*Leave 2.5 blank for now.*

## 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

No decision has been made on dependencies as of yet.

The software will be dependent on Spring Web and Thymeleaf in order to create and execute the MVC architecture that will be developed within NetBeans. The application will also use the World Time API (http://worldtimeapi.org/) that will display the current date and time on the home dashboard for everyone to see.

# Functional Requirements

## Primary

* FR0: The system will allow users to make accounts and log in.
* FR1: The system will allow writers to send drafts to editors.
* FR2: The system will allow editors to edit drafts and add notes for changes to be made. These can then be sent back to writers to produce the next draft.
* FR3: The system will allow writers and editors to post requests for photos to a bulletin viewable by photographers. These requests can contain a written description of the photo they are looking for.
* FR4: The system will allow writers and editors to access photos that are submitted in response to these requests and allow them to close requests.
* FR5: The system will allow photographers to view photo requests on the bulletin, choose requests to accept, and then upload a photo in response to a specific request.

## Secondary

* Authorization scheme so that only relevant parties can access documents and photo requests.
* Account creation and password protection so that each employee can be recognized within the system.

# Technical Requirements



## Operating System and Compatibility

[The environments that will be needed to operate the system]

The application will be compatible with desktop operating systems.

## Interface Requirements

### User Interfaces

Each user will have a separate interface, different users will be able to send different files to each other to collaborate on their projects. There will be a login feature for a user’s profile. On your profile you will be able to add your name, and a description about yourself.

### Hardware Interfaces

The web app will run on any device that can access the internet and access/interact with web pages.

### Communications Interfaces

The operating device must be able to connect to the internet. HTTP will be used as the network communication protocol.

### Software Interfaces

The frontend will be built using HTML and CSS, connected to the backend with Spring Boot and Java. The backend will be built with Java.

# Non-Functional Requirements

## Performance Requirements

[TBD]

## Safety Requirements

N/A

## Security Requirements

* NFR#(R): User account data will only be accessible by said user.

## Software Quality Attributes

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

### Availability

[Details]

### Correctness

[Details]

### Maintainability

[Details]

### Reusability

[Details]

### Portability

[Details]

## Process Requirements

### Development Process Used

Software Process Model

### Time Constraint

### Must be completed by December 6th

### Cost and Delivery Date

No cost. Delivery date is December 6th.

## Other Requirements

TBD

## Use-Case Model Diagram

A diagram of a diagram

Description automatically generated

## Use-Case Model Descriptions

### Actor: Editor (Responsible Team Member)

* **Use-Case Name:** Login
* **Use-Case Name**: Access drafts
* **Use-Case Name**: Send drafts to writers
* **Use-Case Name**: Create photo requests
* **Use-Case Name:** View photo requests and access photos

### Actor: Writer Amin A

* **Use-Case Name:** Login
* **Use-Case Name**: Access drafts
* **Use-Case Name**: Send drafts to editors
* **Use-Case Name**: Create photo requests
* **Use-Case Name**: View photo requests and access photos
* **Use-Case Name**: [Brief Use-Case Description]

### Actor: Photographer (Chris)

* **Use-Case Name**: Login
* **Use-Case Name**: View photo requests and access photos
* **Use-Case Name**: Upload Photos

## Use-Case Model Scenarios

### Actor: Editor (Philip)

* **Use-Case Name**: Login
  + **Initial Assumption**: User has a registered account with credentials which are saved in the system.
  + **Normal**: User will enter a username and password to be logged in
  + **What Can Go Wrong:** User could enter the wrong credentials
  + **Other Activities**: None
  + **System State on Completion**: User is logged in
* **Use-Case Name**: Access drafts
  + **Initial Assumption**: User reaches a page with a grid list of all drafts/completed work saved in system
  + **Normal**: User can preview drafts in alphabetical/chronological order, then select a draft to continue working on it,
  + **What Can Go Wrong:** User could accidentally delete a draft. User could access drafts they don’t have authorization for.
  + **Other Activities**: Delete a draft, make a copy of draft
  + **System State on Completion**: User can view and edit all authorized drafts
* **Use-Case Name**: Send drafts to writer
  + **Initial Assumption**: User has a draft to be sent to the writer.
  + **Normal**: User selects a desired draft and recipient.
  + **What Can Go Wrong**: Draft gets sent prematurely. Writer fails to receive draft.
  + **Other Activities**: View list of previously sent drafts
  + **System State on Completion**: Draft has been sent to writer
* **Use-Case Name:** Create photo requests
  + **Initial Assumption:** User has an account with the permission to post to the bulletin board
  + **Normal:** User will write a description of the photo they want and upload the request to a bulletin board.
  + **What Can Go Wrong:** User could have errors in their writing of the request
  + **Other Activities:** Users can edit and delete photo requests they have uploaded
  + **System State on Completion:** The photo request is posted on the bulletin board and viewable by photographers in the system
* **Use-Case Name**: View photo requests and access photos
  + **Initial Assumption**: User has uploaded a photo request and/or there has been a photo uploaded in response to a photo request.
  + **Normal**: User selects a photo request to view from the UI
  + **What Can Go Wrong:** User could close a request prematurely
  + **Other Activities**: Close a photo request
  + **System State on Completion**: User is able to view a photo request and content associated with it

### Actor: Writer (Amin)

* **Use-Case Name**: Login
  + **Initial Assumption**: User has a registered account with credentials which are saved in the system.
  + **Normal**: User will enter a username and password to be logged in
  + **What Can Go Wrong:** User could enter the wrong credentials
  + **Other Activities**: None
  + **System State on Completion**: User is logged in
* **Use-Case Name**: Access drafts
  + **Initial Assumption**: User reaches a page with a grid list of all drafts/completed work saved in system
  + **Normal**: User can preview drafts in alphabetical/chronological order, then select a draft to continue working on it. User can see edits/suggestions made by editor (if applicable)
  + **What Can Go Wrong:** User could accidently delete a draft. User could access other writers’ drafts.
  + **Other Activities**: delete a draft, make a copy of draft
  + **System State on Completion**: User can view and edit all authorized drafts
* **Use-Case Name**: Send drafts to editor
  + **Initial Assumption**: User has a draft to be sent to the editor.
  + **Normal**: User selects a desired draft and recipient.
  + **What Can Go Wrong**: Draft gets sent prematurely. Editor fails to receive draft.
  + **Other Activities**: View list of previously sent drafts
  + **System State on Completion**: Draft has been sent to editor
* **Use-Case Name:** Create photo requests
  + **Initial Assumption:** User has an account with the permission to post to the bulletin board
  + **Normal:** User will write a description of the photo they want and upload the request to a bulletin board.
  + **What Can Go Wrong:** User could have errors in their writing of the request
  + **Other Activities:** Users can edit and delete photo requests they have uploaded
  + **System State on Completion:** The photo request is posted on the bulletin board and viewable by photographers in the system
* **Use-Case Name**: View photo requests and access photos
  + **Initial Assumption**: User has uploaded a photo request and/or there has been a photo uploaded in response to a photo request.
  + **Normal**: User selects a photo request to view from the UI
  + **What Can Go Wrong:** User could close a request prematurely
  + **Other Activities**: Close a photo request
  + **System State on Completion**: User is able to view a photo request and content associated with it

### Actor: Photographer (Chris)

* **Use-Case Name**: Login
  + **Initial Assumption**: User has a registered account with credentials which are saved in the system.
  + **Normal**: User will enter a username and password to be logged in
  + **What Can Go Wrong:** User could enter the wrong credentials
  + **Other Activities**: None
  + **System State on Completion**: User is logged in
* **Use-Case Name**: View photo requests and access photos
  + **Initial Assumption**: User has uploaded a photo request and/or there has been a photo uploaded in response to a photo request.
  + **Normal**: User selects a photo request to view from the UI
  + **What Can Go Wrong:** User could close a request prematurely
  + **Other Activities**: Close a photo request
  + **System State on Completion**: User is able to view a photo request and content associated with it
* **Use-Case Name:** Upload photos
  + **Initial Assumption:** User has a photo to be uploaded and a request to upload it to
  + **Normal:** User accesses request, attaches file, and then uploads
  + **What Can Go Wrong:** user could upload wrong photo
  + **Other Activities:** N/A
  + **System State on Completion:** Photo has been uploaded to request, requester can view photo