Software Requirements Specification

for

Gusty.bike

Prepared by

Jessica Spranger
Anna Malyevac
John Pruchnic

2/5/2019

1. Introduction

1.1 Purpose

This document describes the requirements for the client's new website. It serves as a guide for the development team, and as a means to minimize ambiguity between the developers and the client.

1.2 Scope

This project is to develop a website for the client, a professor at the University of Mary Washington, Gusty Cooper. The client's students will use the site to submit code answers to posted problems, and anyone will be able to find the site and see the client's blog. The website is maintained by the client following completion.

1.3 References

The client's previous websites may be referenced to further understand his background.

http://gusty.bike/

http://gustycooper.github.io/gustycooper.github.io/

http://gustycooper.org/

1.4 Overview of the remainder of the document

This document is structured such that a general understanding of the product may be understood by reading Section 2, and a more thorough understanding of the product may be garnered by reading the requirements in Section 3. Things to avoid are laid out in Section 4. Assumptions made by the requirements team are listed in Section 5. In Section 6, terms used throughout this document are defined.

2. Project Description

2.1 System overview

The website is hosted on a cloud service. It has two main features: a blog written by the client, and a method for the client to host and receive submissions for a programming contest. The client can add, remove, and edit blog posts as well as define and edit the programming competition requirements. The client has used Mooshak for hosting programming contests before, but any alternative tool including DMOJ or DOMjudge may be used if deemed better. The client can create and remove login credentials for users as needed. Users may see the client's blog whether or not they have an account. Users with accounts may login to participate in contests.

2.2 Client characteristics

The client, Gusty Cooper, is a professor at the University of Mary Washington. The client needs a website that will aid him in grading code submissions from students, and function as a place to host his personal blog. As an experienced computer scientist, the client desires that the site be built using a modern technology stack. He has been disappointed in the past by tools he found unintuitive in frameworks like Wordpress.

2.3 User characteristics

There are three types of users for the website: administrator, logged in user, and anonymous user.

- The client is the site administrator and the only user able to create, edit, and remove blog posts and competitions. The client is able to create and delete user accounts with usernames and passwords.
- Students of the client may be given accounts, allowing them to login and participate in the programming contests by submitting solutions via a file upload. They may view the client's blog.
- Anonymous users may view the client's blog, but may not submit answers to posted programming problems.

2.4 Product functions

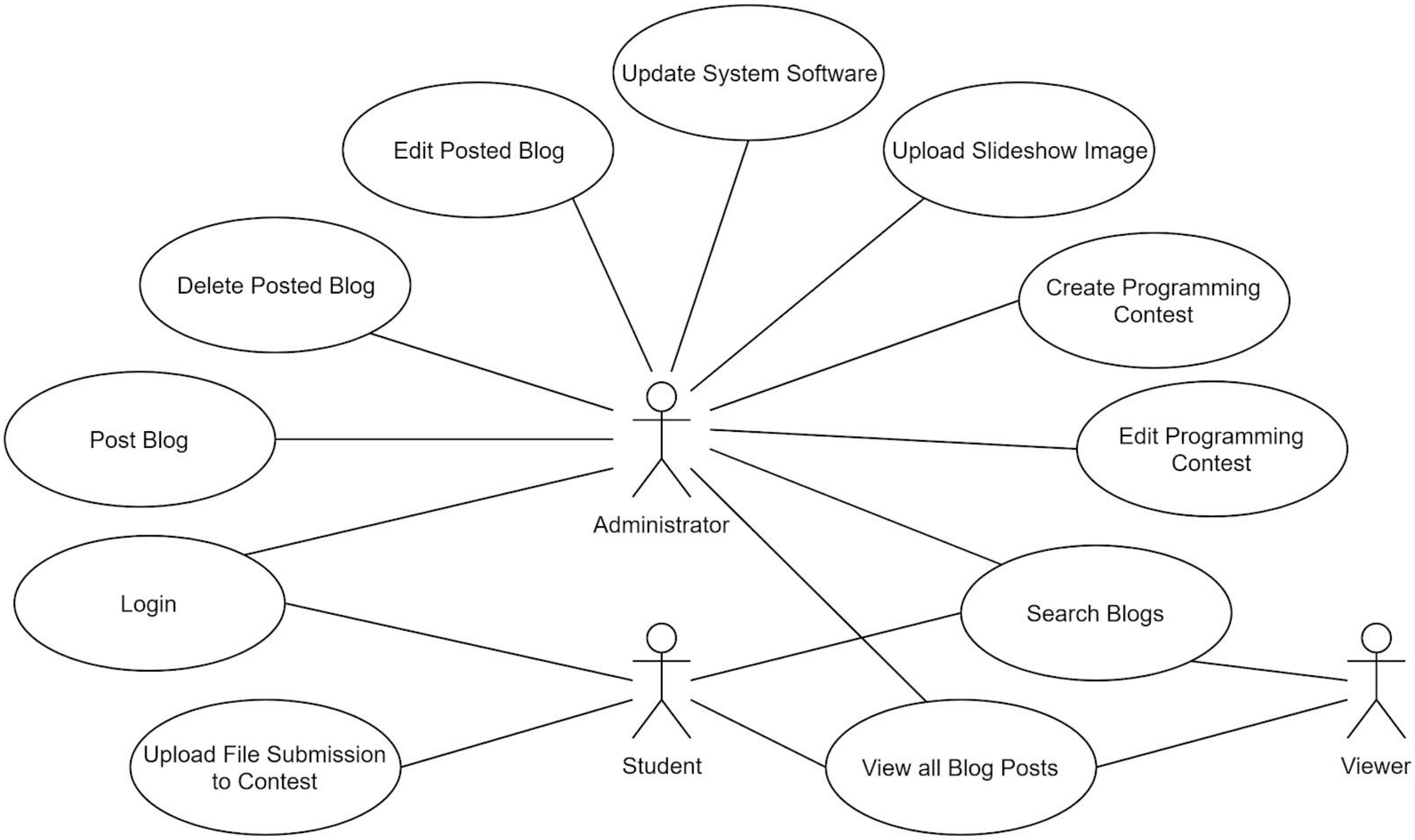
Upon opening the site in a browser, the user will see the home page with a selection of the client's images and a welcome message. From this page, a user can click a prompt to visit the client's blog, or see the currently running programming contest. At the blog page, the user will see the client's blog, and can search for posts with a search bar located at the top of the page. The search feature will match queries with blog content and tags or keywords added by the client. The client will be able to login as the administrator and click a button to start a new post. The post will be written within the website and will support Markdown formatting syntax. At the contest page, students will be able to login and submit their answers by uploading a source code file.

3. Requirements

3.1 Requirements Overview

The following section details both the functional and non-functional requirements of the website. Use cases are used to describe the functional requirements, which include the client's blog posting and maintenance of blog posts that are viewable by anyone, as well as the programming contest, which will be hosted by the client for student submissions. The non-functional requirements describe other constraints on the system that are important to the client, such as his need for ease of use, a flexible budget, and the site hosting. Below is a use case diagram to help visualize the functional requirements.

Figure 1 - Use Case Diagram



3.2 Functional Requirements

3.2.1 - Blog-posting

Description*:* Client will be able to post both text and picture content to the website, where it will be publicly displayed on the homepage. The blog will support Markdown syntax for special formatting. After a post has been made, the client will be able to edit the text and pictures of the content afterwards, and also delete the post altogether. Commenting by the users/viewers of the site shall be disabled.

1. Client will view the homepage of the website
2. Client will be prompted with an immediate option to post content
3. Client will be presented with a text box to post content for the blog
3a. Client will be presented with the option to post an attachment
4. Client will select tags for content to be applied
5. Client will submit text content optionally with a picture attachment
6. Client will be able to view posted content as it appears on the homepage.
Alternate flow 1: Client will be given the option to freely delete posts he has made previously
Alternate flow 2: Client will be given the option to modify or edit posts has made previously
6. Users will be given read permissions on the client's post; however, commenting shall be disabled.

3.2.2 - Blog Search

Description: Any viewer will be able to search the website for other content within the host site. This does not pertain to the contest-hosting service, if a different one is used than the initial site.

1. Users will be provided with a search bar for content within the website
1a. Search bar can filter between categories
2. Search bar will pick out keywords within the website content
3. Website will filter found content to keywords within the site
3b. Website will also search posts for tags applied in the search bar
3c. Search bar will filter content to keywords and categories

3.2.3 - Ease of Use & Resources

Description: Website will be easily updatable and manageable, with the tools available for ease of use and instruction. This will be done with a user manual written by the implementation team describing the structure of their code, and providing further resources on how to use the software they chose and/or coded in.

1. Client will need to update the software
2. Client will open provided documentation on maintaining the website
3. Client will read the provided resources on how the code is structured and written
4. Client will update the website using the content provided in the user manuals

3.2.4 - Admin and User Logins

Description: The client will be provided with an option to login to the website with administrative privileges. They will have the option to manage their own username and password for the website. The client/admin will also be given the option to create separate usernames and passwords for other users. The other users created by the administrator will not have the privilege of changing their username or password, and will not be granted administrative privileges.

1. On the homepage, the client will be provided with the option to login

Alternate 1a: The client will be provided an option to create an account or manage their current account

2. The client will provide their username and password and successfully login

Alternate 2a: Upon entering an incorrect combination, the client will be prompted to re-enter their username and password combination.

3. Once successfully logged in, the client will have permissions to update the website and post content.

Alternate 3a: The client will be given the option to create less privileged accounts, with a given username and password.

Alternate Flow:

1. The users (students) will be provided with a username and password from the client
2. The users will only be given login permissions to the service hosting the student contests.
3. Users will login using given credentials and be given access to participate in admin-hosted contests

Alternate 1b: If username/password combination is invalid, users will be prompted to re-enter their login credentials

3.2.5 - Contest Hosting Service

Description: The client will be able to host student contests, where students with login credentials can participate in either within or outside of the website's framework. The client will be able to post content questions or prompts that can be seen and answered by the student users.

1. Client will login to the contest provider with their credentials
2. Client will prompt to host a contest
3. Client will post a question for users to answer
4. Client will submit test cases to be applied to submitted content
5. Client will specify correct answers for each test case
6. A submission window will be open for logged in users to access
7. See 3.2.6

3.2.6 - Contest Student Submission

Description: The users will be presented with the question to be answered, provided by the client, and will answer them within the website.

1. Students will login via their username and password provided by the client on the contest-held portion of the website
2. Students will wait for a question to be posted by the client
3. Students will submit their content via the provided contest-service
4. Client will be given access to the provided submissions
5. Students will receive feedback on if their provided submission, on if their output matches the test case output or not.

3.2.7 - Add Site Content Page

Description: The client will be able to add pages to the blog to separate blog posts according to their functions. For example, a page category could be programming, bikes, or family, or any page the client chooses to add. These pages will create a new page which will host all of the blog posts in that particular category.

1. Once logged in, the client will view the homepage of the website.
2. The system will display to the client an option to add a blog page.
3. The client will indicate to the system that he wants to add a blog page by clicking the on screen prompt.
4. The system will display a text box that allows the client to enter in the name of their new page.
5. The client will type in the name of their new page in the text box and submit the change.
6. The system will display a success message and will add a new page to the blog with the name of the new blog page category. Now, any post associated with that page will be viewed there.

Alternate Flow:

6a. The system determines the page name is an invalid name, because a page already exists by the name entered.

6b. The system displays an error message to the client and flow returns to step 4 for the client to try again.

3.2.8 - Image Slideshow

Description: The client will be able to add images to an image slideshow featured on the homepage, for everyone visiting his site to view.

1. Once logged in, the client will view the homepage of the website.
2. The system will display to him an option to add an image to the image slideshow.
3. The client will indicate to the system that he wants to add an image through an on-screen prompt

4. The system will display a menu that allows the client to search his computer's documents for an image.
5. The client will chose and image from his computer and indicate he wants to insert it.
6. The system will add the image to the carousel for viewing by any viewer.

Alternate Flow:

6a. The image uploaded by the user could not be formatted into an appropriate aspect ratio and will display this message to the client. Flow returns to step 2.

3.3 Non-Functional Requirements

3.3.1 - Ease of use

The website's features will be simple enough that the client will be able to figure out how to use it within an hour.

Rationale: The client needs to be able to quickly and easily add content to the site without much hassle. The website will be easily accessible to the client and easy for him to maintain in his free time.

3.3.2 - Cloud hosted

The website will be hosted on some cloud based platform. Currently, the client deploys his other websites through AWS, though he is open to other cloud hosting options to best suit his needs.

Rationale: Hosting the website on a cloud based service allows the site to be persistent and allows it to be easily maintained and updated by the client.

3.3.3 - Budget limit

The website's budget will be flexible at around $100 per year. This amount may be negotiated further with discussion with the client.

Rationale: The client is willing to pay for a good quality website that will provide him with the features and ease of use that he needs. Currently, the client pays around $100 per year for his websites on AWS as a reference point, though he is willing to pay more or less for a better alternative.

3.3.4 - Logo

The website will include the client's logo. See appendix 6.3.1 for the client's logo.

Rationale: The client wants his logo to be displayed somewhere creatively on the website, to be seen by all who view it. His son created the logo and he is proud of it.

3.3.5 - User manual

The client would like to receive a user manual that will instruct him on how to use the system and update the system. This will instruct him on how to perform functions related to posting on his blog and setting up programming contests with the system. This manual will also contain details about the implementation and software the team has used for the project, so if updates need to be made to the system, then the client, as someone with knowledge of computer science, will be able to update it in the future.

Rationale: The client needs the website functionality to be easily accessible to him, and would find it helpful for a manual to be included for his convenience. He values having input with the implementation team and is very interested to see the technologies they use to implement the website, which he could expand upon in the future.

4. Non-requirements

The following section details features that the project will not include. These non-requirements are outside of the project's scope and will not be reflected in the final product. Instead, the following non-requirements will help to clearly define what the project will not include in order to clear up any misconceptions about the project's scope.

- The website will not be implemented using Wordpress.
- The website will not be required to be integrated with any existing University of Mary Washington system such as Canvas or Domain of One's Own.
- Neither a viewer nor a student will be able to create an account on the website. Only the client will have privileges to create accounts for students.
- The website will not allow any user, whether they be a viewer or a student, to comment on the client's posts. Only the client will be able to post any information on the blog.
- The system will not have to keep track of the time each student submitted their code for a particular problem for the programming contest.

- The site will not need to feature a scoreboard for the students to view each other's progress or ranking for the programming contest problems.
- The site will not need to be able to import old blog posts from other sources, such as the client's other websites.
- The site will not need to save the client's unfinished and not yet published blog posts to be finished later.

5. Assumptions

The following list includes underlying assumptions that are made about the website, its users, and its stakeholders in this document.

- The client will be able to continue maintaining the website after it has been implemented.
- The client will be responsible for giving the students their login information so that they can login with the correct username and password.
- The students will be responsible users of the programming contest since the client has created accounts for them, and will not upload malicious code.
- The implementation team will be able to research in order to choose and implement an appropriate cloud hosting service for the website.
- The implementation team will be able to choose and implement software appropriate for implementing the programming contest features, such as Mooshak or other alternatives they discover.

6. Appendices

See the following references for further details on the project.

6.1 Glossary of Terms

- Amazon Web Services (AWS): A popular cloud computing platform hosted by Amazon. The client currently employs this service to host his websites. See https://aws.amazon.com/ for more details.
- Jekyll: A static site or blog generator, currently implemented in the client's old website, http://gustycooper.github.io/gustycooper.github.io/. See https://jekyllrb.com/ for more details on Jekyll.
- Mooshak: A system that allows for implementation of a programming contest or similar functionality on a web platform. For more details see https://mooshak.dcc.fc.up.pt/ for Mooshak version 1 and https://mooshak2.dcc.fc.up.pt/ for Mooshak version 2.

- DMOJ: An alternative tool for implementing programming contests. See https://dmoj.ca/ for more information.
- DOMjudge: Another alternative tool for implementing programming contests. See https://www.domjudge.org/ for more information.
- Wordpress: A free and open source website management tool. This is provided by the University with Domain of One's Own.

6.2 Author information

The following documents each member's contribution to the document.

Anna: Sections 3.2.1, 3.2.2, 3.2.3, 3.2.4, 3.2.5, 3.2.6

Jessica: Sections 4, 5, 6.1, 3.1 first paragraph, 3.2.7, 3.2.8, 3.3

John: Sections 1, 2, 6.1, Figure 1, Formatting uniformity

6.3 Additional documents

6.3.1 - The Client's Logo

The client's logo features Jaffy the giraffe, a character of his creation that represents his website.

Figure 2 - Jaffy the Giraffe