gusty.bike

Test Plan Document

David Velasquez

Chris Blankenship

Clay Sweetser

Section 1

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this document is to describe the schedule, requirements, metrics, and processes of the tests that will be used to evaluate the implementation of the *gusty.bike* project.

1.2. Scope

Our project is a website for a professor at a university. The website will be modern looking and easy to maintain and edit by the professor. Likewise, his students will able to submit their projects and get feedback from the submission section website.

1.3. References

- CPSC 430 Writing Assignment #2 Project Plan Document
- Group 5 Requirement Documentation
- Software Requirements Specification for Gusty.bike

1.4. Document Overview

The remainder of this document is split into 4 sections:

- Section 2, "Project Description", in which the project, its client, its users, and its requirements are described.
- Section 3, "Test Plan", in which the plan for testing the project is described. This section contains information on the strategies, metrics, and schedule for project testing, as well as the resources required for testing.
- Section 4, "Test Procedures", in which the processes for testing the project are listed and described.
- Section 5, "Appendices", in which additional information about this document is listed. This includes a glossary of terms, author information, and references to additional documents.

2. Project Description

2.1. System Overview

Our system is comprised of two server-based applications that work together. The first, DjangoCMS, handles the front facing website and content creating capabilities. These include the client will be able to edit at will, will have privileges to add/remove content and subsites. Students will be able to reach the site and consume the content, as well as submit their projects to the second application, Mooshak. This is a seperate app that is linked to the system and takes the code submitted by the students and gives feedback.

2.2. Client Characteristics

The client for this project is Gusty Cooper. He is a Computer Science professor at UMW. He has requested this project because he feels that his current set of websites are unmanageable, and wants a modern website that he may use as both a personal and professional blog, and a teaching aid.

2.3. User Characteristics

The intended users of the gusty.bike site include the client himself, Gusty Cooper, as well as his students and anyone else interested in his blog posts. Our client will use the site to post relevant course information in a simple and easily navigable manner, as well as use Mooshak as a grading platform for his student's work. The simplicity offered by DjangoCMS will likewise provide easy navigation to the aforementioned blog posts.

2.4. Functional Requirements

2.4.1. The client should be able to log into the website in order to add new content and otherwise administer the site.

2.4.2. The client should be able to log into Mooshak in order to manage programming assignments.

2.4.3. Students of the client, and anybody else, should be able to access the site in order to read the client's blog posts and class material

2.4.4. Students should be able to log into Mooshak to upload class assignments for grading.

2.4.5. Other visitors to the site should be able to read the client's blog posts and view other content.

2.4.6. Students and other guests should not be able to leave comments on the posts to the blog.

2.4.7. The client should be able to access the Virtual Machine in order to update the software, and make any low-level changes to the software

2.5. General Constraints

2.5.1. The website should integrate Mooshak2 in order to accommodate use by the client and his students. This can be done using a reverse proxy application (such as Nginx) to route incoming website traffic to multiple backend applications.

2.5.2. The website should be secure. This can be achieved by using a service such as Let's Encrypt to automate the acquisition and use of SSL/TLS certificates for HTTPS traffic. This will require the website to have an associated domain name.

2.5.3. The website should be hosted on a stable cloud platform, such as Amazon Web Services, Microsoft Azure, or Google Cloud Platform.

3. Test Plan

3.1. Testing Strategy

To test the system the testing teams will need to do a couple basic website usage tests. They will access the site, navigate it using the navigation bar to reach the various subsites of the system. The development team will provide a test user login in order for the test team to manage the site as an administrator and create posts. Likewise, a simple contest will be created for testing that the Mooshak server is functioning as intended.

3.2. Testing Resources and Staffing

The only thing necessary for end user testing will be a computer with internet access. The test team can use their prefered web browser to reach http://52.170.117.211/ and navigate the site. In order to test super user settings they can append /admin to the url and reach the admin login page and enter username: *tester* with password: *test*. For end user testing, they will only require a web browser to visit and log in to the website, as well as the ability to use an ssh client to log into the virtual machine. If the test team wishes to emulate client testing, and deploy the environment on another server, they will only require a server running Ubuntu Linux. Most of the configuration for the website can be recreated using a bash script, requiring only some user input, with firewall configuration and creation of website content needing to be done by hand.

3.3. Test Work Products and Record Keeping

Assuming the testing team and client face no issues accessing the website and logging into the administrator pages, the products of testing will include blog posts and pages they are able to create for the site. Likewise, as long as no issues are faced logging into Mooshak, new users and new contests can be created and act as proof that the tests were conducted and were conducted successfully.

3.4. Test Schedule

The Implementation Team will be spending one full week running tests on the website. The first day will be spent ensuring that both DjangoCMS and Mooshak can be accessed and logged into. The setting up of contests on Mooshak will take place the second day, and they will be tested on the third day. Likewise, changes to the content and appearance of the DjangoCMS will take place beginning the third day and extend the rest of the week. It is expected that the testing group will spend a similar amount of time, ensuring access, expanding on the Mooshak contests, and further modifying the CMS.

4. Test Procedures

All tests should be carried out 3 times, using the Firefox, Chrome, and Internet Explorer browsers. Each test will involve either navigating to a page on the website, or using a particular feature available on the page of the website.

The tests are grouped into 4 sections:

- Anonymous User Tests
  Tests that represent actions that all users of the website (including students and administrators) are able to carry out.

- Student-specific Tests
  Tests that represent actions that students using the website are able to carry out.

- Administrator-specific Tests
  Tests that represent actions that administrators using the website are able to carry out.

- Client-only Tests
  Tests that represent actions that the client of the website will be able to carry out.

4.1. Anonymous User Tests

Test 1. Navigate to http://52.170.117.211/

Test 2. Navigate to http://52.170.117.211/ and click the "Home", "Blog", "About", and "Courses" links. Attempt to navigate to each link on each page.

Test 3. Navigate to http://52.170.117.211/en/blog . Click the ___ button.

Test	Related to Requirement	Purpose	Test Case Data	Expected Results
1	2.4.1	Test ability to reach server	http://52.170.117.211/	Reach site without Nginx or HTML 404 error
2	2.4.3 & 2.4.5	Navigate the website	http://52.170.117.211/en/	Navigate the site and view the content without broken links or missing content
3	2.4.6	Ensure no comments can be left on blog pages	http://52.170.117.211/en/blog/	Comments cannot be left, anonymous or otherwise

The anonymous user tests will reflect the use of any anonymous user reaching the site and navigating it. The testing team should be able to easily reach the site with the provided address without any errors, and be able to do this using only a web browser. The website's menu bar will be the main form of navigation and testers should be able to reach all the subsites with it.

4.2. Student-Specific Tests

Test 4. Navigate to http://52.170.117.211/ . Click the link labelled Mooshak .

Test 5. Navigate to http://52.170.117.211/mooshak/Authentication.html, Fill out the "User" and "Password" fields. Click "Login"

Test 6. Navigate to http://52.170.117.211/en/blog . Ensure there are no links with the labels "Comment", "Add Comment", etc.

Test	Related to Requirement	Purpose	Test Case Data	Expected Results
4	2.4.4	Test ability to reach Mooshak	http://52.170.117.211/mooshak	Reach Mooshak with no errors
5	2.4.4	Test ability to log into Mooshak as a student	http://52.170.117.211/mooshak Username: team Password: team	Log into Mooshak and see what a student would see
Includes all tests Anonymous User Tests				

The student-specific tests for the website will overlap heavily with anonymous user tests for accessing DjangoCMS, therefore the testing team should focus on student use of Mooshak in this round of tests. This will include signing in to the platform, and submitting work to the test contests to ensure that the software is functioning correctly. The testing team will be able to do this using only a web browser, and using the provided information, log into the student-side of Mooshak.

4.3. Administrator-specific Tests

Test 7. Navigate to http://52.170.117.211/admin/ and login with the provided credentials. Navigate through the interface, create pages, etc.

Test 8. Navigate to http://52.170.117.211/mooshak/Authentication.html and login with the provided credentials. Navigate through the interface.

Test 9. Navigate to http://52.170.117.211/mooshak/Authentication.html and login with the provided credentials. Navigate through the interface.

Test	Related to Requirement	Purpose	Test Case Data	Expected Results
6	2.4.1	Test administrator access for DjangoCMS and add content	http://52.170.117.211/admin/ Username: tester Password: test	Access DjangoCMS super user controls and use them to add new pages, blog posts, people, etc.
7	2.4.2	Test administrator access for Mooshak	http://52.170.117.211/mooshak Username: admin Password: admin	Access Mooshak administrator tools
8	2.4.2	Test judge access for Mooshak	http://52.170.117.211/mooshak Username: judge Password: judge	Access grading tools on Mooshal

Includes all Student-Specific and Anonymous User tests

To test the functions of the website administrator, the testing team will need to ensure that basic, unprivileged, access to both the DjangoCMS content and the Mooshak platform is possible. Assuming it is, the testing team can then explore the administration functions of DjangoCMS: adding new pages, blog posts, and so on. The testing team will likewise need to explore the administration functions of Mooshak, including the creation of new users and contests, using a web browser to access the administrator tools of both DjangoCMS and Mooshak.

4.4. Client Only Tests

Test 9. Using an SSH cliient, connect to gusty@52.170.117.2 and login with the provided credentials.

9	2.4.7	Access the VM	ssh gusty@52.170.117.2 Password will be provided in-person	Access the VM the web apps are running on to further customize them.
Includes all Student-Specific, Anonymous User, and Administrator Tests. Only to be done by the client themselves.				

The client should be able to perform all possible tests, from simple anonymous access to advanced administrator functions to ensure that the website is working to their specifications. The client themselves should also be able to test access to the virtual server that the website is hosted on, to both better understand the file structure of the installed programs, and to learn how to make low-level tweaks (such as theme tweaks, add-on installation, etc.). This is the only test that requires tools besides as web browser and a slightly more advanced knowledge of Linux systems. It can be easily performed through a unix terminal emulator like those found on any Linux distribution and macOS, or GUI based ssh tools like PuTTY.

Once the server has been reached, testers must be very careful not to make permanent changes that could interrupt the normal operations of the system. If the testing team does seek to perform this test, they can do so using the setup scripts provided to deploy their own copy of the preconfigured software on their own server or virtual machine. This will make it possible to navigate through all the directories and locate the setup and startup scripts for the system to gain a better understand of the dependencies needed to run the website. This test will be required in order to generate an SSL certificate for the website, but only after the setup scripts have been run, and the server is configured with a domain.

5. Appendices

5.1. Glossary of Terms

Apache:

Web server software commonly used to host web applications
(https://httpd.apache.org/)

CMS:

Content Management System
A website/web application primarily used to organize and display textual and visual content and media.
(https://en.wikipedia.org/wiki/Content_management_system)

Django:

A Python-based web framework.
(https://www.djangoproject.com/)

DjangoCMS:

A CMS built using the Django web framework.
(https://www.django-cms.org/)

Mooshak:

Software used to manage programming contests on the web
(https://mooshak2.dcc.fc.up.pt/)

Nginx:

Web server and reverse proxy software. Also commonly used to host web applications.
(https://www.nginx.com/)

SSH:

A protocol for encrypted communication over computer networks.

TLS/SSL:

An encryption protocol for HTTP communication.

(https://en.wikipedia.org/wiki/Transport_Layer_Security)

UMW:

University of Mary Washington

(https://www.umw.edu/)

5.2. Author Information

Clay Sweetser

- Title and Table of Content Pages
- Introduction Section
- Project Section
- Test Procedures
- Glossary

Chris Blankenship

- User Characteristics
- Test Work Products, Record Keeping, and Schedule
- Test Procedures
- Glossary

David Velasquez

- Scope
- System Overview
- Testing strategy
- Testing Resources and Staffing
- Test Procedures

5.3. Additional Documents

Existing Websites:

- http://gusty.bike/
- http://gustycooper.org/
- http://gustycooper.github.io/

Git Repos:

- https://github.com/cblanke2/gusty430
-

Client's Logo