IS 226 Project: Software Requirements Specifications (SRS)

For

stepostr- a Step by Step Topic Documentation Content Resource Management (CRM) Web Platform

By:

Sardez, Angelito Jr. Ortiz

Student No.: 2016-30105

Master of Information Systems Student

University of the Philippines Open University

April 8, 2017

Table of Contents

Release Information

Project:	stepostr- a Step by Step Topic Documentation Content Resource Management (CRM) Web Platform
Internal release number:	1.0.0
Related documents:	Use Case Suite Use Cases Feature Set Features

Introduction

This document contains information about the functional, non-functional, and system requirements pertaining to the implementation, delivery, and maintenance of the stepostr platform project.

The details in this document are about the version 1.0.0 release of the project. Since this is the initial release, the features presented here already satisfies the minimum viable product requirements of the project thus all functionalities, specifically those which are listed as "Essential", are required to be delivered with no or considerably small compromise.

The sections in this document is organized based of information groups. The previous section listed the release information, while the subsequent sections will focus on the software requirements itself. Hence, the next sections will feature the use cases and each requirement groups and the details related to them.

Use Cases

According to the project's vision, stepostr aims to provide a Content Resource Management web platform specifically designed for easily documenting topics which are represented in step by step fashion. Given this statement, the platform should then satisfy a set of requirements from creating, managing and publishing such pages.

Aside from the page management part, there are other aspects of the platform which deals on the platform's maintenance which also need a set of requirements. Most importantly, the pages are aimed to be read by the platform's users so requirements specific to site navigation are also presented here.

Putting it all together, the overall use case diagram of the stepostr platform is as follows:

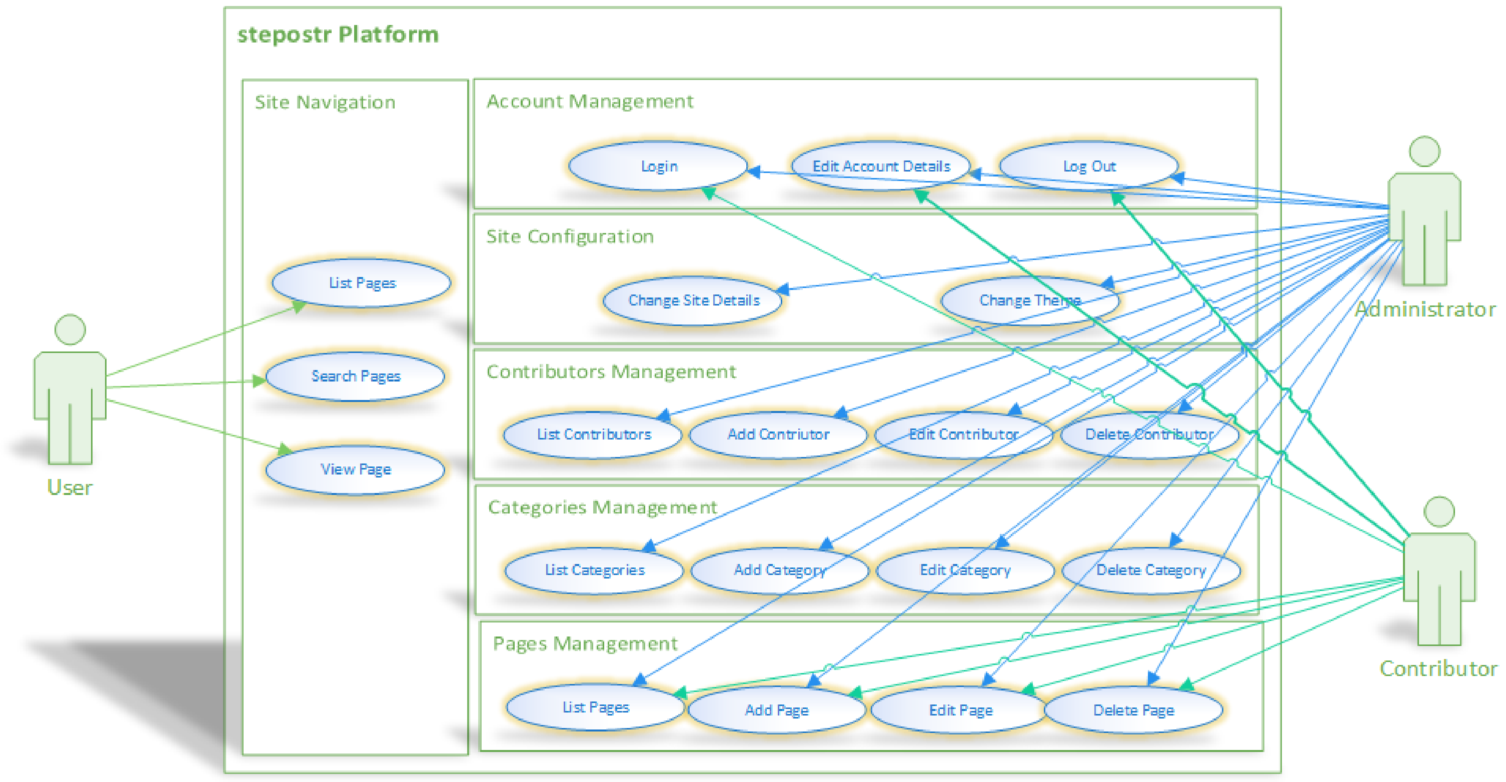

Figure 1: The aggregated use case diagram of the stepostr web platform.

Actors

The actors in the platform's use cases are:

- **User** – they could either be one time or regular visitors who browse the site to read published pages.
- **Administrator** – they basically maintain and configure the web site. They also maintain the accounts of those who can manage the web site and publish pages.
- **Contributor** – they are special type of users who can login to the administrative area of the site but they only have limited rights which includes managing posts or pages and their own account.

Each of the use cases presented in the diagram are discussed in detail in the use cases document. The list of use cases are organized in the use case suite document.

Functional Requirements

The functional aspect of the stepostr platform are grouped into the features that caters the need of each actors described in the last section. The subsequent sections will briefly discuss this features. More information about these features could be found in the feature set document.

Site Navigation

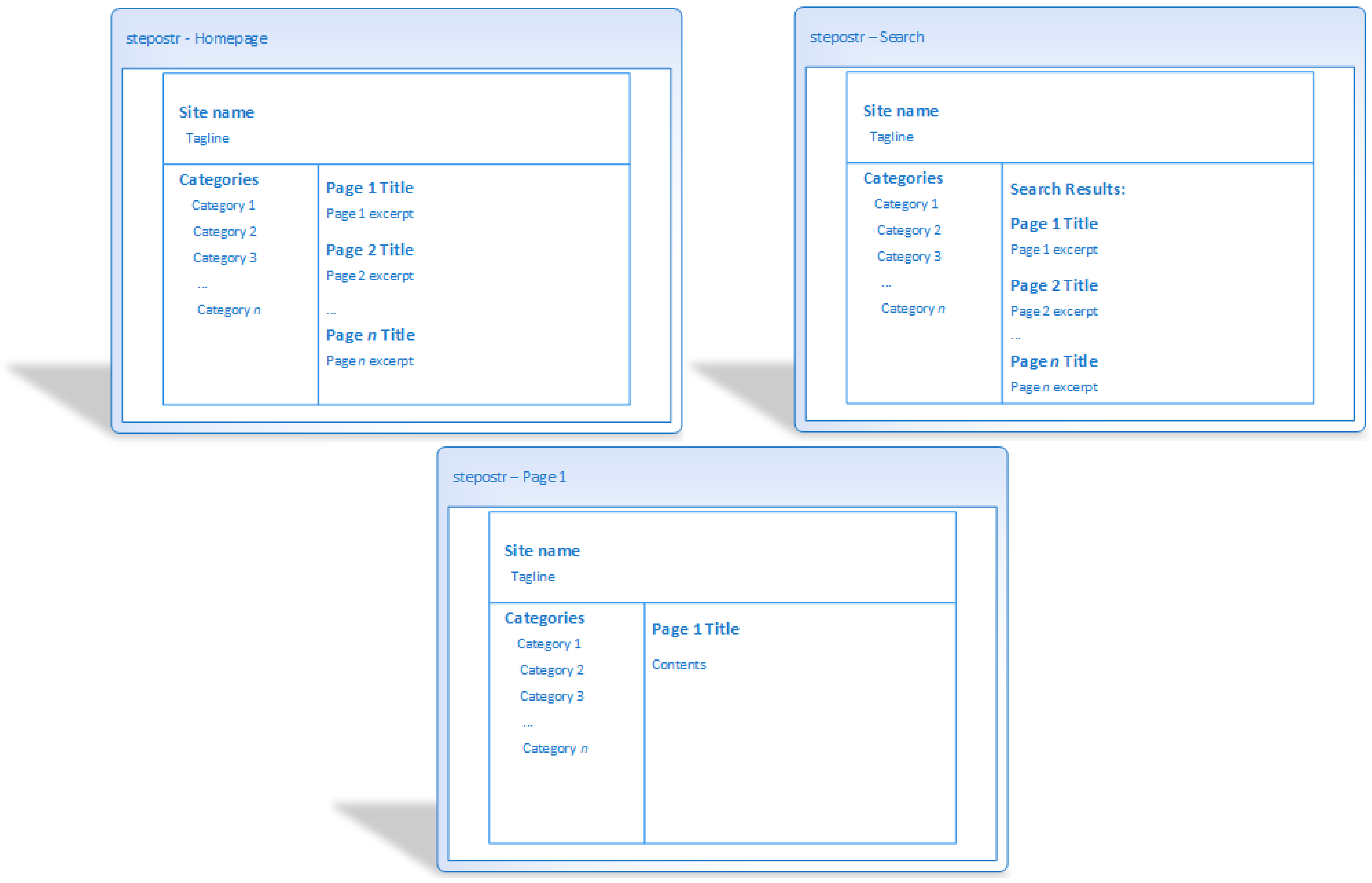

Figure 2: Site Navigation Wireframes

The site navigation contain functionalities aimed at the users. These features will basically enable them to use the platform effectively and efficiently by providing page listing and searching functionalities. This is where pages are rendered for viewing mode and where the site's theme configuration takes in effect.

More information about the site navigation feature could be found in the features document.

Account Management

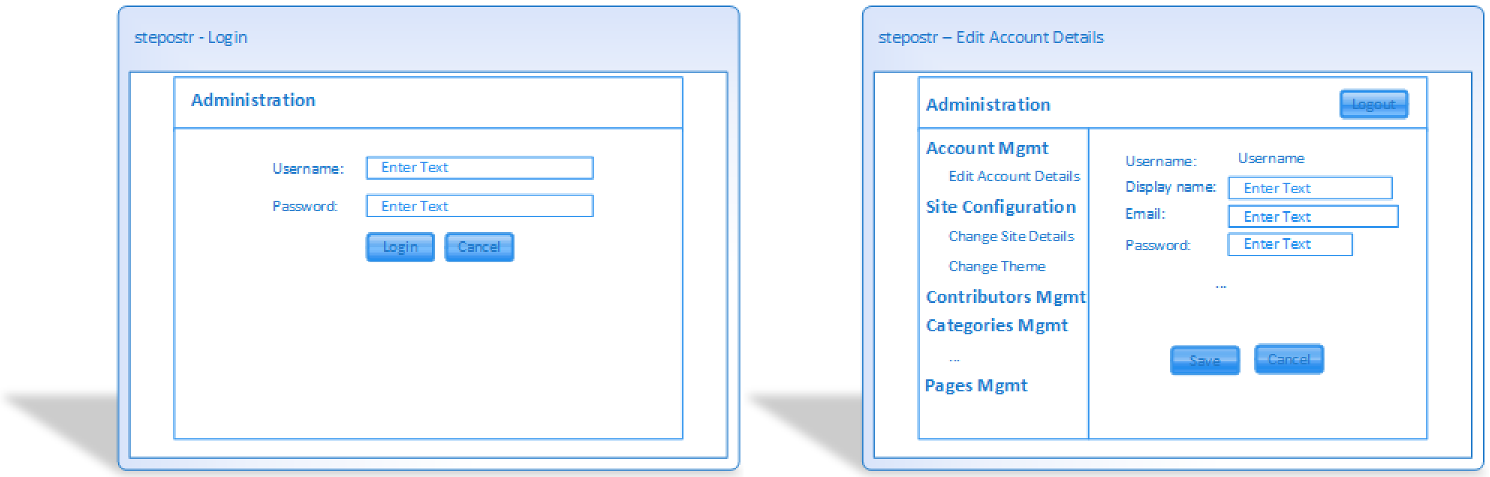

Figure 3: Account Management Wireframes

For the administrators and contributors, the features under account management group will let them access the site's administrative section in order to perform actions for maintaining the site and its contents. This is where they can also modify their account details.

For more details, you may look into the account management feature section found in the features document.

Site Configuration

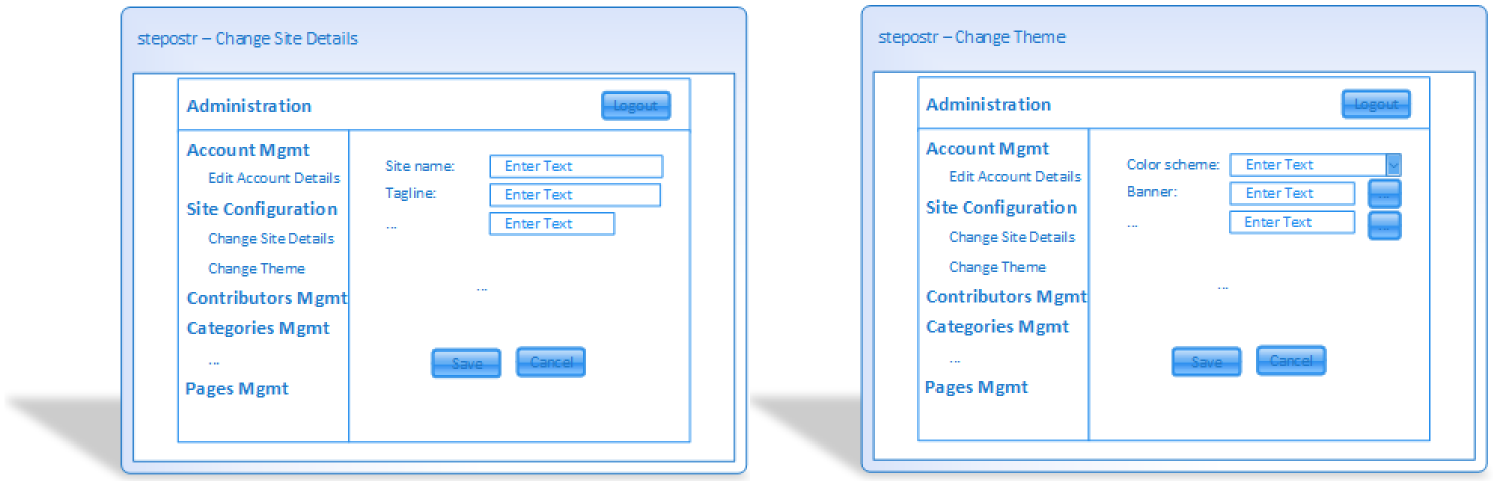

Figure 4: Site Configuration Wireframes

Personalization is also important since it gives an instance of the platform character and identity. The site configuration features lets the administrator modify the site's information and set the theme for the users to experience.

Site configuration feature is detailed in the features document.

Contributors Management

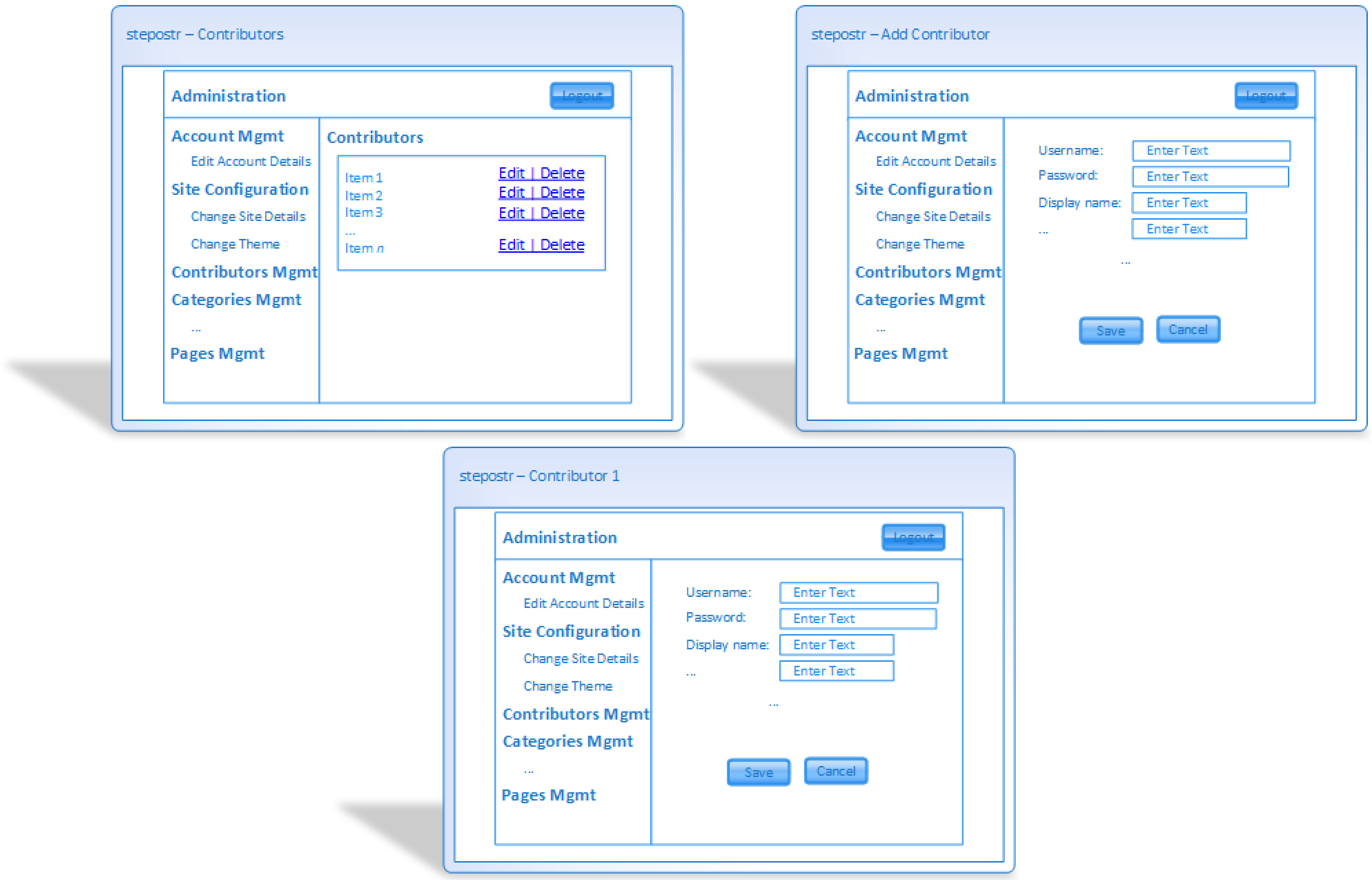

Figure 5: Contributors Management Wireframes

The content management of the platform is not just the responsibility of the administrator. The platform allows multiple contributors to post and manage pages for the users to read. Contributors management feature allows the administrator to create and manage contributors who could post pages in the site's administrative section.

The contributors management feature in the features document contains more information about the feature.

Categories Management

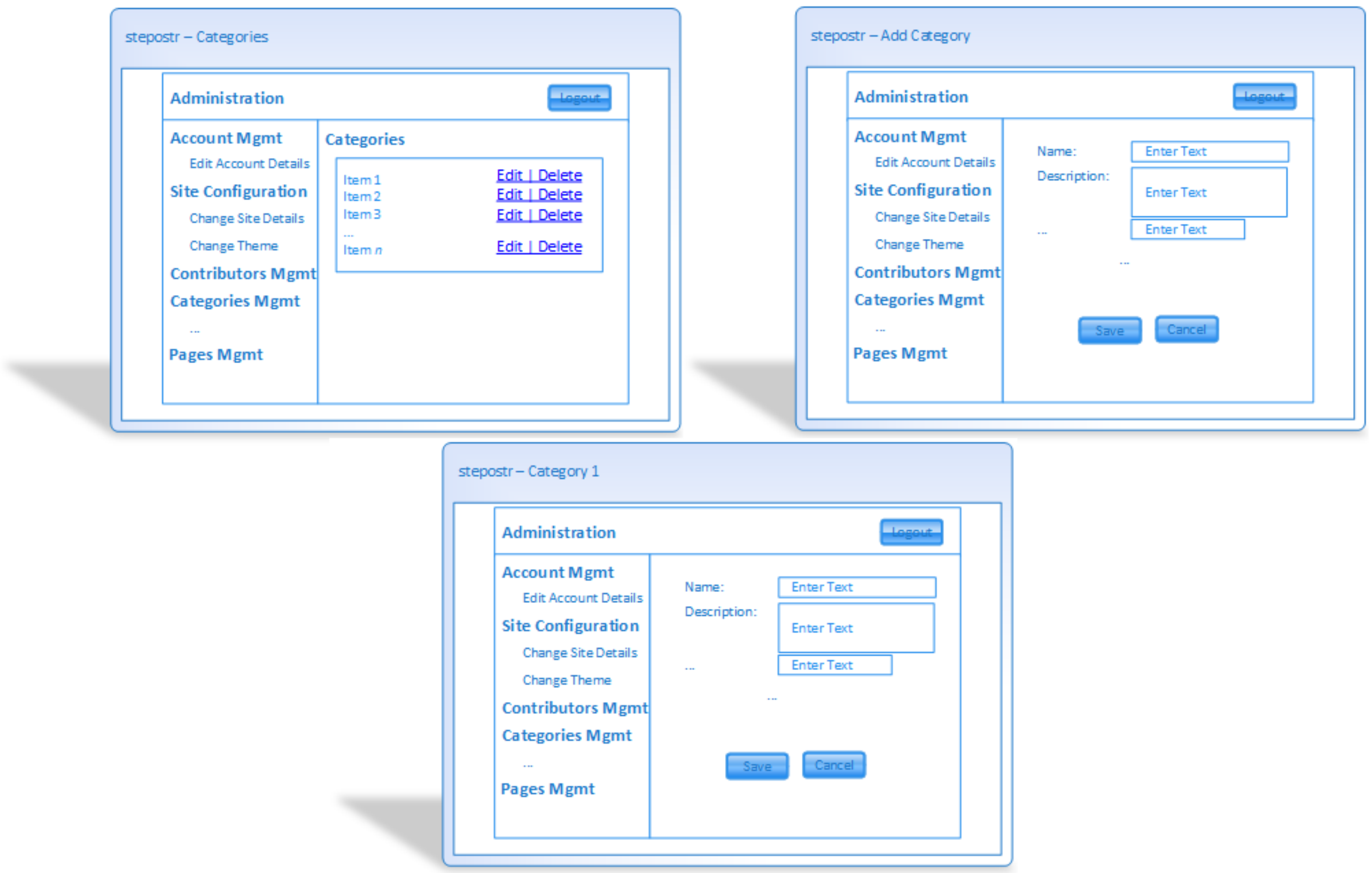

Figure 6: Categories Management Wireframes

Every page posted in the platform should be under a specific category. This is to organize them for viewing and even to optimize the page searching operation of the users. Categories management lets the administrator define and organize the pages' categories in this feature.

Details about the categories management feature could be found in the features document.

Pages Management

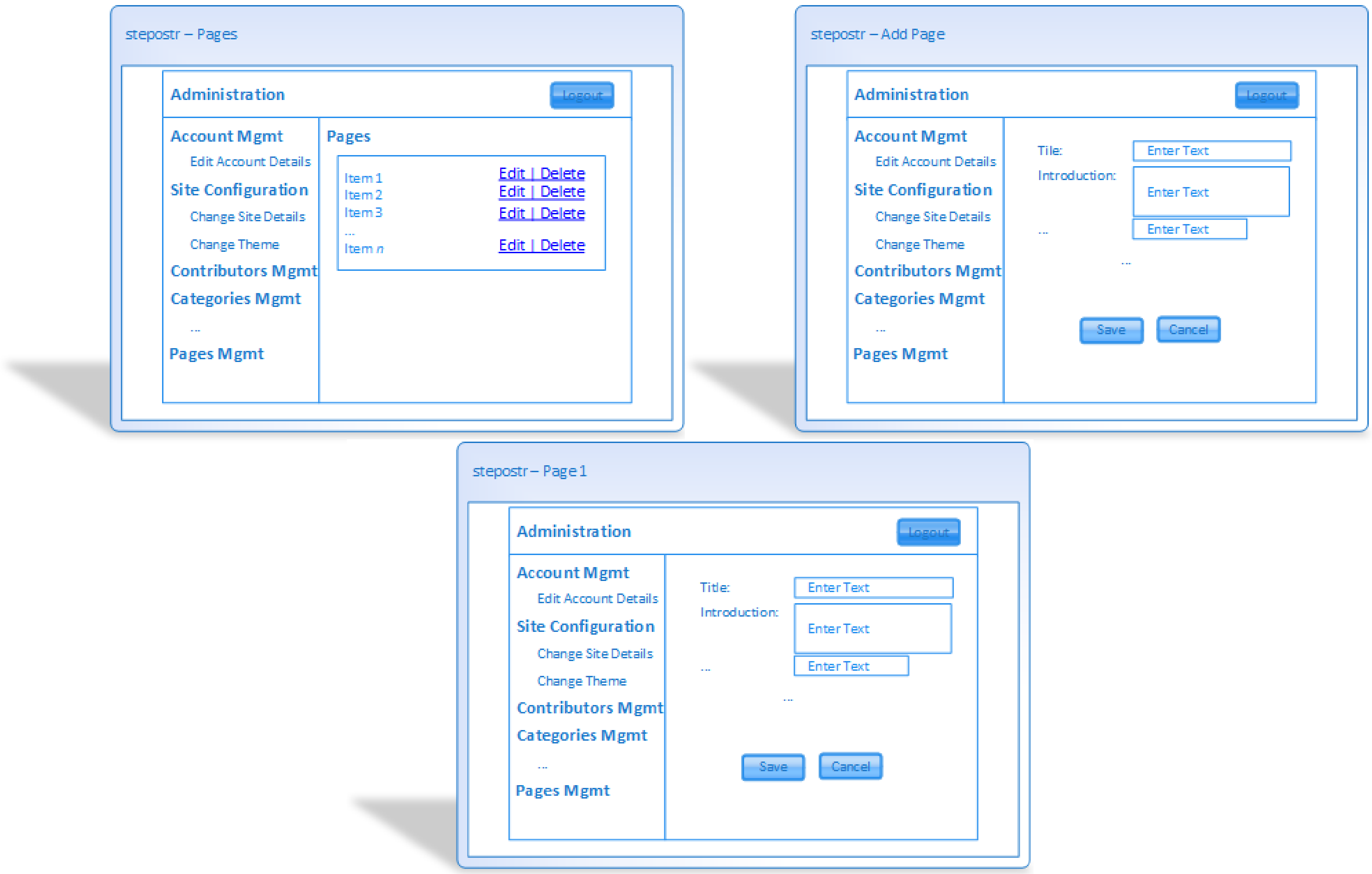

Figure 7: Pages Management Wireframes

This feature allows the administrator and contributors to manage the pages which includes adding a new one, editing an existing one, and deleting pages.

More information about the page management feature could be found in the features document.

Non-Functional Requirements

The non-functional requirements of the stepostr platform pertains to the considerations that should be implemented on top of the required features of the application. These requirements should not compromise the implementation and delivery of the functional aspect of the platform.

Usability Requirements

The usability goal of the stepostr platform intends to satisfy all the actors. The following requirements should be applicable to all actors:

- **Understandability** – The platform should be easy to understand. In the context of the users or visitors, navigating through the site should be self-explanatory. As for the administrators and contributors, the purpose of each functionality offered to them in the administrative section of the platform should be easy to comprehend.

- **Learnability** – The features offered in the administrative area of the platform should support learnable user experience. Each section and every element should have a brief description what they are for in order for the administrators and contributors to use them independently.
- **Operability** – The operations pertaining to use cases should be consistent. E.g. adding a category should result in a new pages category added to the database, deleting a page should result for the page to be removed from the database, and so on.
- **Attractiveness** – The platform's theming feature should provide a way for administrators to set the site's color scheme and also other visual properties such as the site's banner and background images.

Reliability and Up-time Requirements

The recommended up-time requirements for on-line deployment of sites that will use the stepostr platform is that they should be available 99% of the time in order to cater users around the globe which could visit the site anytime. This mean that it should be limited to only 3.65 days of downtime every year. In order to support this high-availability requirement, a web hosting provider or an Infrastructure as a Service (IaaS) provider which offers 99% uptime or better should be used to host the platform.

The reliability and up-time requirements completely depends on the site owner that will use the platform. This could be dictated or determined based on the purpose and nature of the site and \ or service-level agreement with their clients.

Security Requirements

Since the stepostr platform is a web application, it should be following best security practices and prevent most of the common web application attacks. The platform should be developed adhering the prevention of OWASP Top 10 attacks in mind. The risk featured in the features document are basically based on this list and each feature should mitigate them whenever applicable.

As for some of the specific security requirements, the platform's administrative section should be password protected. Only administrators and contributors should be able to access it. Role based security should be implemented as well wherein administrators should be able to access all the sections in the administrative area of the platform while contributors could only access their account details and the page management section.

The operations in the platform should prevent cross-site scripting (XSS) and injection attacks especially in the site navigation wherein anonymous users could search and browse pages.

Performance and Scalability Requirements

Whether the site that will use the stepostr platform will be hosted by a web hosting provider, Infrastructure as a Service (IaaS) provider or on premise, the site could scale up or down since the platform is implemented using the client-server architecture.

The stepostr platform is composed of the web application and its database. The web application could be hosted in multiple servers based on need and they could consume the same database where the database server is expected to handle multi-client load.

A load balancer is then required to sit in front of the web application instances to distribute the traffic to each nodes.

A sample on premise deployment of a site that uses the platform hosted in a 2 web server node is shown below:

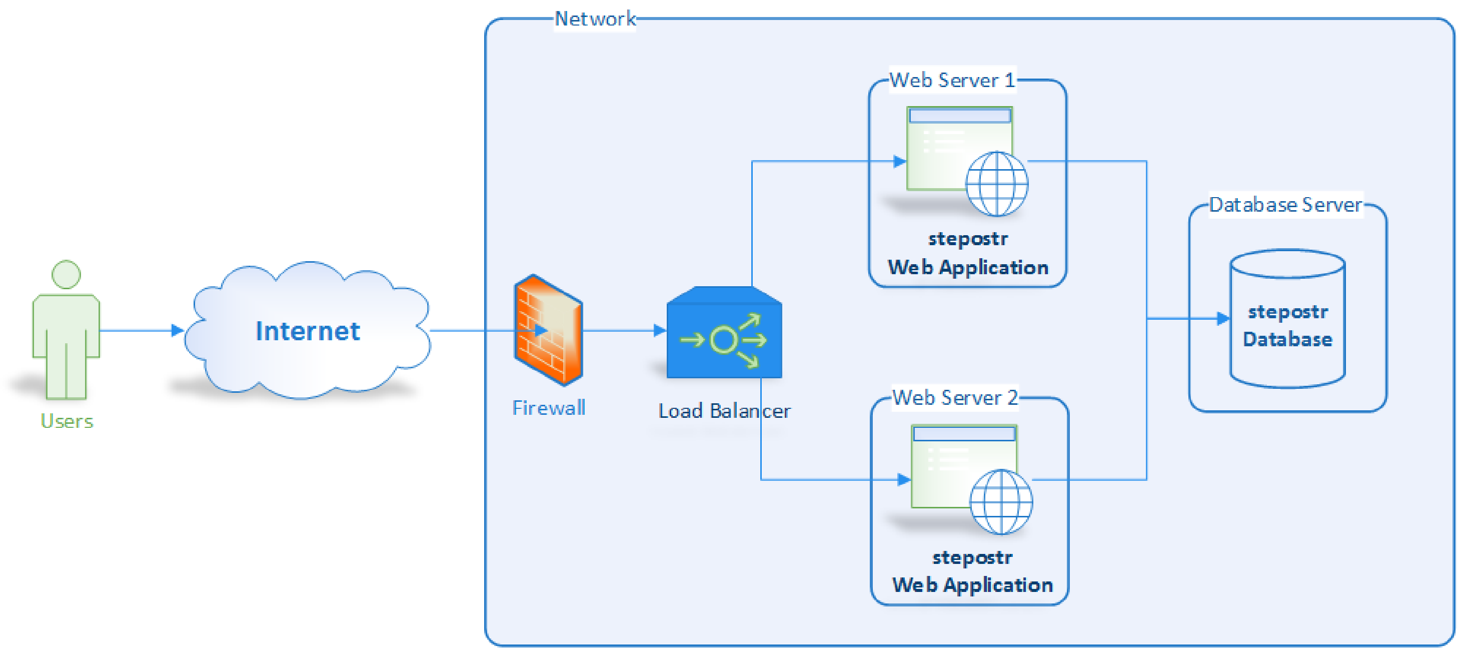

Figure 8: Deployment model of a 2 node web application based on the stepostr platform in an on premise network.

The same deployment model could be achievable in an Infrastructure as a Service providers wherein most of them, e.g. Microsoft Azure offers web \ cloud load balancers and availability sets. It depends on the chosen web hosting provider though whether they provide the same or similar flexibility.

Maintainability and Upgradability Requirements

The implementation of the stepostr platform should be highly maintainable. This is to address issues and introduce new features with reduced cost and quick time to market.

In order to achieve this, the source code management, application design and implementation must follow the following guidelines and principles:

- The code base should be uploaded to a collaborative code repository such as GitHub.
- The implementation of the objects within the application should follow the SOLID principles (single responsibility, open-closed principle, Liskov's substitution principle, interface segregation and dependency inversion) of object oriented programming and design.
- The implementation should be following a common web application design pattern like the Model View Controller (MVC) pattern.
- It is recommended that the codes should follow the PSR-1 Basic Coding Standards for PHP, for uniformity.

The design should be evolutionary rather than revolutionary in order to support harmless and near-uninterrupted upgrades. Breaking changes should be avoided as much as possible. A migration facility should be provided in case database changes will be introduce in future versions of the platform.

As for the deployment strategy during upgrades, one approach that could be implemented is the Blue-Green deployment pattern. This is an effective solution for achieving zero downtime during upgrades in a load-balanced, multi-web server deployment of the platform. Basically, the Blue pertains to the "cold" node which shouldn't accept traffic while the application's instance there is being updated. The Green then pertains to the "hot" node or the node that will continually accept the traffic while blue is currently being upgraded.

Once the application instance in the blue node finished upgrading, it will be switched to a green node and the previous green should be switched to a blue node and the application instance deployed there should start upgrading. This will be done for all web servers until all nodes finished their upgrades and switched to green.

Here is a deployment model that shows the Blue-Green deployment pattern implemented when upgrading stepostr platform application instances:

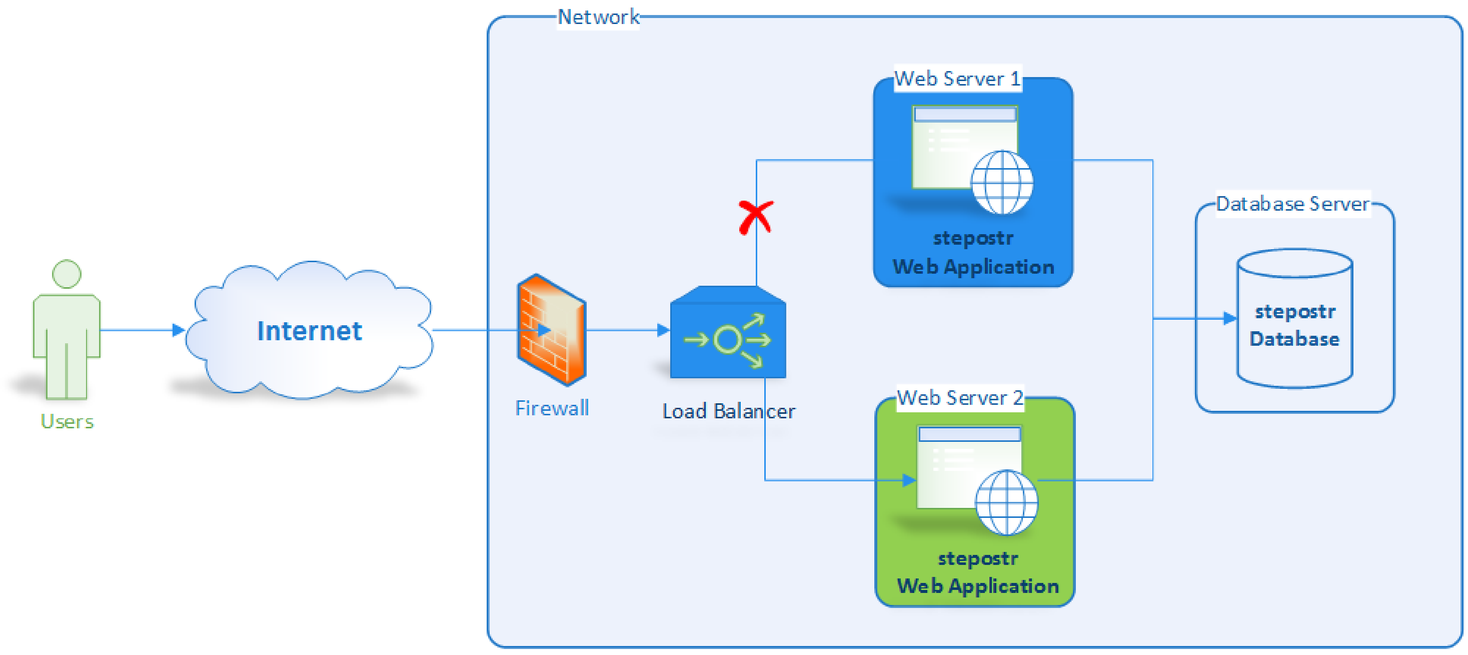

Figure 9: The Blue-Green Deployment Pattern implemented on web applications based on stepostr platform.

The key entity in the Blue-Green deployment pattern is the load balancer since it will decide which nodes should accept the traffic when there is an ongoing upgrade.

Since this model still falls under the client-server architecture, changes in the database (server) should also be backward compatible since the web applications (client) in the green nodes which aren’t updated yet might access the database whose structure might already been migrated by the blue nodes.

This deployment model is applicable to both on premise web application deployment as well as Infrastructure as a Service provider deployment like in Microsoft Azure which offers cloud load balancers and upgrade domains. Scaled-up application instances deployed to a web hosting provider might have a different upgrade strategy depending on the web hosting provider’s offering.

Supportability and Operability Requirements

One key non-functional requirement that the stepostr platform should have is logging. There are two classification of logging that should be implemented: system logs and audit logs.

System logs should contain significant events happened in the application like critical and fatal errors. They should be meaningful and as detailed as possible in order to provide a productive technical support when needed.

Audit logs on the other hand pertains to the operations performed by the users of the application. This information will also be useful in an event of support since it will give the support personnel additional information about the actions made by the users which eventually lead to (or before) an issue.

These logs could either be stored separately or in a single file as long as they are differentiable.

The regular backup of the database should also be setup in order to fallback in an event of catastrophe in any of the nodes in on premise or IaaS deployments, or in web hosting providers. The recommendation is to have the database backed-up every day during the determined off-peak hours.

Environmental Requirements

These requirements pertains to the minimum system properties that the environment (server \ node) should have before an instance of the stepostr platform’s application and database are to be deployed in them.

System Hardware Requirements

The following information are based on the **combined minimum hardware requirements** of the operating systems, web and database servers that could host the stepostr web application and database. Basically each node should have:

- 2 GHz processor or higher,
- 1 GB RAM or higher,
- 32 GB free hard disk space or higher, and
- Graphics card and monitor capable of 640x480 or higher

System Software Requirements

The next sections describes the **software requirements** of web and database servers required by the stepostr web application:

Operating Systems:

- Windows Server 2012 R2 Standard or higher, or
- Ubuntu 16.04.2 LTS or higher, or
- Red Hat Enterprise Linux 7, or
- CentOS 7

Web Servers:

- Internet Information Services 8.0 or higher (for Windows Server deployment only), or
- Apache 2.4 or,
- NGINX \ NGINX Plus

General:

- PHP 7.0
- MySQL 5.5

Data Import and Export Requirements

All data required and collected by the web application instance of the stepostr platform are stored in its database. The database should be accessible to the authorized users or other applications who intends to consume and process these data.