Department of Computer Science

This project has been satisfactorily demonstrated and is of suitable form.

This project report is acceptable in partial completion of the requirements for the Master of Science degree in Software Engineering.

Development of a Content Management System (CMS)

Project Title (type)

John Mai

Student Name (type)

Chang-Hyun Jo

Advisor's Name (type)

Advisor's signature Date

Ning Chen

Reviewer's name

Reviewer's signature Date

Abstract

Content Management System (CMS) is a web application that is used to create and share contents online. For this project, a CMS web application known as Community was developed. Community is a CMS that provides a place for users to be able to create and share contents with one another.

Community CMS is built using an Agile software development process known as Scrum. The project uses the Scrum process in order to develop a web application from start to finish. Scrum promotes quick releases of the product in each sprint and allows stakeholders to be closely involved throughout the project. There were three sprints during the development phase of the project, each developing and meeting the user stories that were assigned to them. Furthermore, because Scrum promotes quick releases of the product after every sprint, development and testing were done closely with one another throughout the project to ensure that the product meets the requirements that were set for them. Overall, the project was a success and Community the CMS web application was developed and delivered to the customer.

Keywords List:

Content Management System, CMS, Agile, Scrum, software development, software process, web application, web app development, user stories, requirements

Table of Contents

Tables and Figures:

1.0 Introduction

1.1 Description of the Problem

Content management system (CMS) is a software application that can be used to connect people everywhere together. People can use the CMS to communicate with one another by sharing their contents with each other. The CMS will allow people to be able to communicate more easily with one another on the internet. For this project, I used the Agile- Scrum software development to build a simple content management system (CMS). As a software engineer, building such a system requires proper planning and implementation of a software development process that will be most suitable to develop the system.

1.1.1 Content Management System (CMS)

Content Management System (CMS) is a software application that is used for creating and managing content. There are two types of CMS system that is widely used by businesses or organizations. These two types are known as the enterprise content management system (ECM) and the web content management system (WCM) [1]. The ECM creates contents that uses tools, such as the software along with organization or business process and incorporate both of these into its content [2]. The WCM is similar to the ECM but it focuses more on the web content than the business or organization's process to create content. CMS is a popular web application used by a lot of users everywhere. CMS creates an environment where its users are able to create, advocate, and manage contents while being able to share the contents with their audiences.

Why is a CMS system needed? CMS is a good web application that can be used to share contents and ideas with one another. In this day and age, everything can be shared through the internet. Almost all sites that you get your information from such as news or even Reddit are all types of content management system. One study shown that implementing content management system in their organization has promoted teamwork and communications amongst the users within the organization [15]. Employees are able to use it to communicate,

post problems and help troubleshoot them for everyone to see on the CMS. This makes communication in the workplace more efficient for employees. Higher education institutions also claim that learning CMS may also be a better learning tool than the learning management system (LMS). They believe that the learning CMS is capable of creating personalized learning environments for their students because when students post or create contents the application can use that data to create these personalized learning environments for them.

CMS has always been a popular web application used widely by everyone. Like I mentioned before, it makes communicating and sharing contents more easily accessible. There are some key features that make is necessary to make a CMS. These key features are format and content management, editing, publishing, searching and retrieving contents. Scrum-Agile software development framework would be most beneficial to build this system since Scrum-Agile promotes quick to market products and adaptability when it comes to requirements changing of the web application.

1.1.2 The Scrum-Agile Software Development Framework

Agile is an iterative approach to the software development process. The process is usually divided into multiple time-boxed (fixed time) iterations [6]. Each iteration is treated as its own mini project with its own goals and objectives. During each iteration, user stories and features will be implemented, tested and released to stakeholders so that they can make sure it is what they want. This methodology allows more flexibility and it will meet the customer's expectation since they are more involved with the development process [6].

Stated by the Manifesto of Agile, the Agile process promotes individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following plans [4]. These manifesto allows the process to be more inclusive of the customer and allows more flexibility when it comes to requirements changes. This will help build a more complete software that meets the customer's requirements.

Scrum is an Agile development that can be used for the software development process. Scrum focuses on building software that meets the business needs. In Scrum, there are four phases to its lifecycle. The four phases are: planning, staging, development, and release. The figure below shows the four phases and the activities that goes along with each phase.

Figure 1: The Scrum-Agile Software Development Framework

Since Scrum is an Agile development it also follows the iteration format as well. In Scrum the iterations are called Sprints and each sprint can last up to 30 calendar days [8]. In order to be able to keep track of progress during these sprints, the Scrum team will use product backlog, sprint backlog, and burndown charts specifically sprint burndown and velocity burndown charts. These are used to make sure that the team is on track with the project.

Furthermore, in Scrum, daily stand up meetings are required so that each scrum team member can state how they are doing on the project, tasks they need to complete that day and what objectives or goals are they setting for themselves [8]. These stand up meetings help make sure that every scrum team member is on track with the project. Sprint reviews are done at the end to go over all the action items that was made in the beginning of the sprint and also to create burndown charts to see what the team can do to improve on next sprint [9].

The Scrum team usually consist of a Scrum master, product owner and the scrum team. The scrum master is like the coach for the team. The Scrum master is there to remind the team of the task at hand and the vision that needs to be met. The Scrum master usually helps resolve

any issues that the team can run into during development. The product owner on the other hand represents the customers. They will often be in charge of prioritizing the features that they would want to be developed in each sprint. The product owner must communicate efficiently with the Scrum team to ensure that the product will be built accordingly and meet the requirements. The rest of the Scrum team are consisting of developers and maintainers. The Scrum team is often consisted at most seven members. Small team is better for collaboration and they can communicate more thoroughly and efficiently with one another.

1.2 Project Objectives

In order to accomplish this software project, three objectives were set for the project. The three objectives are mentioned below. With these objectives in mind it helped guided the processes and steps that were taken during this project.

Objective 1: To properly implement the Scrum-Agile software development framework for the content management system.

For this project, I have decided that Scrum-Agile would be the most suitable software development framework to use to build a CMS web application. With Scrum and Agile incremental process that promotes quick small releases of features. The process also allows more customer involvement so that the product can be completed according to the requirements customers have set forth.

This objective was achieved by implementing the four phases of the Scrum process. The four phases are: Planning, Staging, Development, and Release. I ensured that work products such as user stories, product and sprint backlog, burndown charts, stand up meetings and sprint reviews are all properly created so that I am able to thoroughly follow the Scrum-Agile development process.

Meeting this objective has demonstrated my ability to implement a software development framework. It demonstrated that I am able to create a project plan and carry it out to the end

and deliver a working software product. It also shows that I am able to elicit requirements and create a software manual for the working product.

Objective 2: To improve programming skills by learning and implementing PHP, HTML, CSS, Bootstrap, and MySQL database in order to create a content management system.

The second objective is geared towards learning and improving my programming skills. After careful research, I have decided that it would be best to build this CMS web application using PHP, HTML, Bootstrap and the MySQL database to house my data. These programming languages allowed me to create the front-end and the back-end development of the CMS system. MySQL database retrieves and sends data to and from so that users are able to login and create content posts.

This objective was met by demonstrating proper programming and delivery of a working CMS web application. It challenged me to learn a new programming language that I have not learned before. My skills in programming was strengthened by accomplishing this objective. I also was able to determine the complexity of the system and estimate the time that was needed to produce the product. Executing this objective, gave me a better understanding of how the development aspect of the overall software development process.

Objective 3: To develop a simple responsive content management system (CMS) web application that is user friendly.

The end goal of this project was to be able to create a simple responsive content management system. Being able to properly implement the last two objectives ensured that this objective was met as well. Implementation of the Scrum-Agile software development process ensured that the proper steps to software development were taken and also ensured that all requirements are thoroughly elicited and reflected what the customer wanted in the product. This ensured that the CMS were built to meet the requirements that had been set for it. Furthermore, proper implementation of the programming language also ensured that the product was built since that is the method used to build the product in the first place.

Meeting this objective demonstrated my ability to carry out the project plan and see it through from start to finish. The product was released and handed over to the stakeholders for review. The final product determined that I have been able to execute the appropriate software development process for this project.

Overall, these three objectives were achieved once the CMS web application was built. I was able to utilize my understanding and knowledge of a software development process specifically a Scrum-Agile process and incorporate it into building a CMS web application. I was able to plan and carry out a Scrum-Agile project from start to finish. This project, has allowed me to take all that I have learned from the master program and successfully apply them to a software development project.

1.3 Development Environment

Building the content management system (CMS) required proper tools and environment. For this project, I used a Retina display Mac Book Pro laptop as the hardware to develop the CMS. The Mac Book Pro is currently running on Mac OS Sierra with a 2.3 GHz Intel Core i7 processor. It has 8GB of memory and using NVIDIA GeForce GT 650m 1024 MB and Intel HD Graphics 4000 1536 MB as the graphic card. I used a dual monitor in order to have multiple windows open at once for better display. All of the development was done on this laptop.

To build the system, I used a text editor software to write the code for the system. The text editor that I used was Sublime Text. I used it to write both the front end and the back end program.

For the front-end development of the CMS web application I used HTML, CSS, and Bootstrap to create the styling of the web application. The HTML and CSS gave the page a simple structure. I used it to design the navigation bar and different pages for posting and viewing the contents on the web application. The Bootstrap helped with the styling, giving the page with a simpler responsive style and also buttons and borders around different areas of the web application.

The back-end development was done using PHP. I was able to use PHP to write the different functions for the web application. Functions such as creating, posting, and viewing the contents were done using PHP. PHP also helped write functions for signing up and login in as well. The Web application used the Apache server in order to host the web application. I used XAMPP which included the Apache server and MySQL database as well. MySQL database was used as the database to house all the data collected from the web application. Data such as user information and the written content were stored and retrieved from the database.

Lastly, documentation and work products were also created for the web application. I used Microsoft Office Suite 2016 to create the final report. The Microsoft Suite included Word, Excel and PowerPoint. These tools were used to create any work products and documentation that were needed in the final report.

1.4 Operational Environment

Since the CMS is a web application, it operates on a web server. In order for users to be able to access the CMS web application they will need to be able to access the internet and go on the website Community. The website will be compatible and be able to run on any of the following browsers: Firefox, Chrome, Safari, and Internet Explorer. The website design should be responsive. Being responsive allows the website to respond properly to other smart devices used by users. It is also able to run on Mobile, desktop and tablets. Smart devices with operating system such as: Android and iOS are able to run the web application. Other computer operating system such as Window OS, Mac OS, Chrome OS, and Linux are also compatible to operate the web application as well. As for hardware components, any working computers, laptops, smart phones or tablets will be able to run and access the web application.

2.0 Requirements Description

2.1 Functional Requirements

Functional requirements are used to specify how the application will perform when under a set of condition. Developers tend to use the functional requirements in order for them to know what needs to be implemented in the system. Usually these functionalities are implemented to help users be able to accomplish their tasks when using the system. The various functional requirements that are implemented can go from data manipulations to a specific functionality that is implemented to clearly define what the system is supposed to do.

- FR-1: Users shall be able to input any contents when they select the content input field.
- FR-2: When the user clicks the submit button, the application will post the content that they have submitted.
- FR-3: When the user clicks the edit button, the application will display the post that they want to edit.
- FR-4: When the user clicks the save button during an edit, the application will post the revised version of the post.
- FR-5: Users shall be able to delete the post they have made when they click the delete button.
- FR-6: The application will log user's information, when user submit their credentials on the signup form.
- FR-7: The application shall display all the contents that have been posted by users.
- FR-8: User shall be able to log off, when they click the logout button.

2.2 Nonfunctional Requirements

Nonfunctional requirements specify a property, characteristics or some constraints that the system must display and/or experience. The nonfunctional requirements describe how the system is supposed to be or work. Nonfunctional requirements and quality attributes can be

used interchangeably to describe these characteristics. There are many different types of quality attributes, some of them are described as performance, safety, availability or any characteristics that are deemed important to its users.

- NFR-1: The application shall be available at all times.
- NFR-2: The input field will alert user, if there is nothing in the input field when submitting.
- NFR-3: When the user sign in, the application will verify the credentials they have input and permit them access into the account if it is correct.
- NFR-4: If improper credentials are submitted by a user, the application will display a warning.
- NFR-5: The application shall display appropriate fonts and sizes for readability.
- NFR-6: The system shall log users off when they exit the web application.
- NFR-7: The application will post the content user have selected within 2 seconds after it has been submitted.

2.3 Architecturally Influential Factors (AIFs)

Architecturally Influential Factors (AIFs) are factors that stakeholders or developers considered are important when it comes to making decision for the architecture of the system. These factors usually affect the way the system will be built.

- AIF-1: Resource is an important AIF. Depending on the resources that we have it will affect the software product that we will be building. Using programming language that is available or only understandable to the developers can change the way the software is being built. Therefore, resource will affect how the product is built.
- AIF-2: Stakeholders/customers is another AIF. The stakeholders and customers have a say in how the product should be built. The product needs to conform to the requirements that have been set by the stakeholders. If these requirements are not followed properly then the stakeholders will not accept the product and it would not be completed. Therefore, it is important to meet the stakeholder's requirements.

- AIF-3: Time is an important AIF as well. The product needs to be completed in a timely manner. Without time management this cannot happen. Poor time management can lead developers to not have enough time to build functions that may be deemed as high in priority to the stakeholders. This will also not meet the stakeholder's requirements and therefore would not be complete. Therefore, time is important because it could affect the overall system's architecture.

3.0 Architectural Design

Community, the CMS, uses a Client-Server architectural framework. The users that go on and uses the web application is the client and the data and tools that are used to run web application are the server aspect of the architecture. The diagram below shows how a Client-Server architecture looks like and interacts with one another. The CMS will be using a type of Client-Server architecture known as the Multitier architectural framework which usually there are several different tiers to the system. For the CMS we will be using three tiers for the system. The three tiers are the presentation layer, the logic layer, and the data layer. These three tiers work together in order to make the web application function properly.

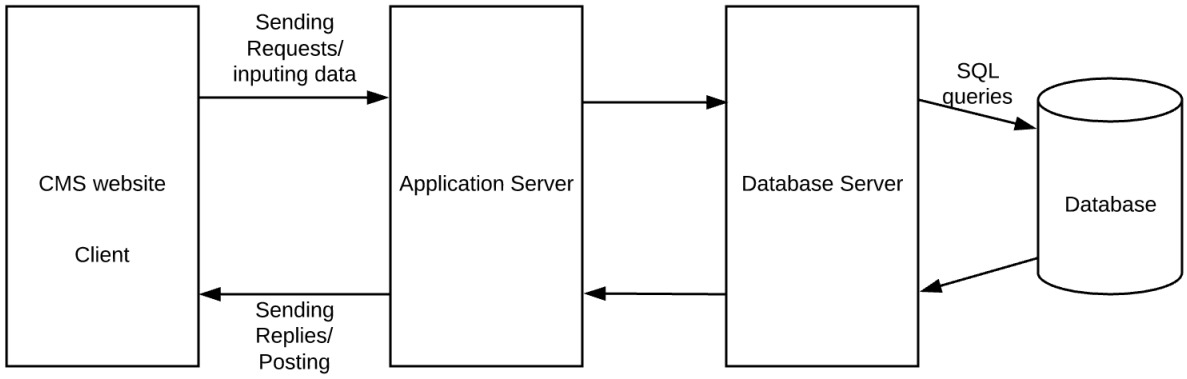


Figure 2: CMS Client-Server Architecture

The First layer is the Presentation layer. This is the most superficial layer of the three tiers. The presentation layer is where the user interface or the front-end web server is. The presentation tier displays all the information that is needed for the web application. The user interacts with this layer in which it sends a command to the other tiers. The presentation layer will be where users will interact with the CMS in order to sign up, sign in and create posts. The programming languages that tools that are used in this layer would be the Bootstrap framework, HTML, and JQuery library.

The second layer is the Logic layer of the web application. This layer is where the back-end of the web application comes in to play. This layer processes the dynamic contents of the web application. Different PHP functions will be called out in this layer so that it can function appropriately. The logic layer communicates with both the presentation layer and the data layer. The presentation layer will prompt the logic layer to either fetch data from the data layer or function accordingly. The logic layer will then do as it is prompted to do so. The programming languages and tools that are used in this layer would be PHP.

The last layer is the data layer of the web application. This is where the back-end database is housed. Data storing and management all happens in this layer. Oftentimes the logic layer will send a query to the database to fetch the data and then the data tier will be able to send it back to the logic tier in which it will send the data to the presentation tier to either display it there or to give access to a function. For the CMS I used XAMMP which has MySQL database and Apache server. MySQL database is where all the user information and blog posts data are sent and stored in.

The following diagrams shows how the Multitier Architectural framework looks like. The CMS uses this client-server framework to function as a web application.

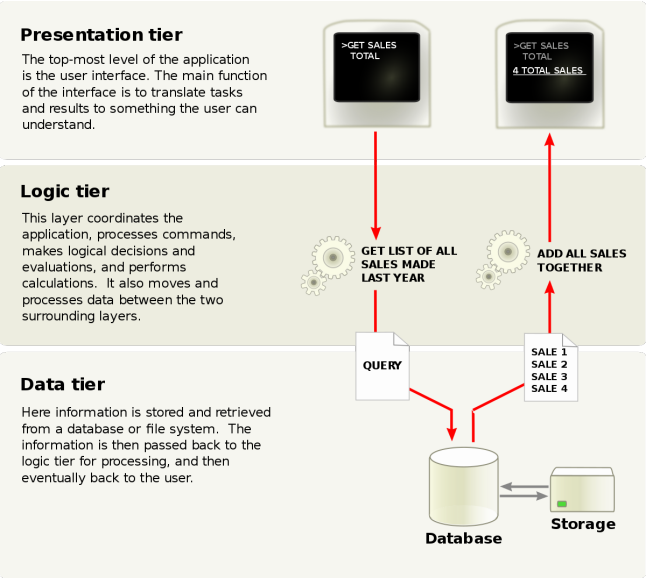


Figure 3: The Three-Tier framework

4.0 Pregame: Planning and Staging

In the pregame stage of the Scrum-Agile framework, the planning and staging tasks of the project was performed. The vision and scope of the software product are presented in this section which ensures that I have a clear idea of how the software product should be. The requirements section states clearly all the functional and nonfunctional requirements that have been set for the product.

Following the Scrum Agile software development framework, I had my stakeholders created User Stories for the project in order to see what features were important to the stakeholders or what features they would like to see in general. The User stories were used to come up with the product backlog and the sprint backlog for project.

Estimation for time cost and resources were made in the planning and staging phase of the process to ensure that there are enough time, money and resources allocated to building such a product.

4.1 Vision

For this project, a simple content management system (CMS) known as Community was built for a customer who have expressed interest in a software product that would allow them to communicate with other users. Community allows users to be able to post their content on the web application where other people may be able to see it as well. The web application also indicates which user is posting the content as well for everyone to see.

The CMS will largely help users from different organization be able to post contents and share it with one another. By posting and sharing contents with one another, it hopes to bridge the gap of communication between two parties. The CMS is interactive and allows users to create their own account or identity for the web application. This also allows them to take ownership of the things that they create or share with other users.

As the CMS becomes popular within the organization, we are looking to expand it to other types of work place or organization that may want to implement the system within their work place. The CMS could be used in business settings, educational teaching settings, or just leisure and friendly settings to share one ideas with each other.

4.2 Scope

Often times it is necessary for people to share and communicate with one another in the community or organization. The scope of this CMS is to provide users an outlet to post and share their contents with other users. The system allows users to register themselves on the web page and be able to securely sign in and out of the web application. They will then be allowed to create, edit, delete, and manage their contents as they wish on the application.

New users will be able to create a new account before using the web application. Once the user has created a user account the user may then sign in to the web application using the login credentials that they have created for themselves.

Once in the CMS web application the user will be able to see posts that have been created by other users. There will also be a field that allows them to post their own contents as well. Once posted, users may manage their contents by editing or deleting the post altogether. Users may only be able to edit or delete their own post and not other user's post in the web application.

4.3 Goals of the Project

For this web application, we focused on three goals for the project. The first goal is to ensure that the web application is built to meet the customer's requirements. The requirements that have been set for the product is an important guideline to follow. We worked closely with the customer and made sure that we have a working product to our customer's content at the end of each iteration. Successfully implementing the requirements set by our customers meant that we have successfully created the web application.

The second goal of this project is to make sure that our web application functions properly and appropriately. Carefully planning out the development process and carrying out necessary software tests during the development lifecycle ensured that we built a web application that met the stakeholder's requirements. The user stories created allowed us to organize and monitor our tasks on the sprint backlog.

The last goal for this project is to ensure a safe and secure CMS web application. The web application allows users to be able to sign up for a user account and also allows them to sign in whenever they want to post a content. This ensures that every user is registered and saved in the database. This allows us to monitor and ensure a safe space for all our users.

Meeting these goals ensure that our web application is built to how the customer wants it to be built.

4.4 Stakeholder Profiles

Stakeholder	Major Value	Attitudes	Major Interests	Constraints
Project Manager	Leading the project to meet the organization manager expectations	Strong commitment throughout the project.	Building a product that would be good for the company.	Cost and time to build the product
Organization Manager	Improve employee's or organization productivity and communications	Committed throughout the releases and excited to put the product into good use for the company.	Cost and productivity of employee should be more beneficial than the cost of building the product.	Cost of building product.
Content User	A better way for communication with one another	Strong enthusiasms for the product	Availability and accessibility to communication	Accessibility and security.

Table 1: Stakeholder Profile

4.5 System Features

The system features of Community are presented in this section. These system features were created to showcase the different features that were implemented in the CMS web application. The system features show what the system is capable of doing and how it can be used.

SF-1: User Registration

SF-2: User Sign in

SF-3: User Log Out

SF-4: Create Content

SF-5: Manage Content

SF-6: Submit contact form

SF-7: User Profile

4.6 User Classes and Characteristics

User	Characteristics
Community CMS Users	• Sign up to be a part of the CMS Community • Login to account • Post content • Edit own content • Delete own content • View contents posted • Submit contact form • Log Off

Table 2: User Classes and Characteristics

4.7 User Stories

The team and the stakeholders sat down together and came up with the user stories for the project. Each stakeholder wrote features that they would want to see in the software product and wrote them on index cards. The index cards were submitted to us and we have created a

table and logged them. The User stories are carefully prioritized based on their business values and risk. These user stories are to be placed in a sprint and product backlog and are to be created into tasks for each iteration based on their prioritization.

ID	User Story
US-1	As a user, I want to have a login field so that I can be able to login to my account.
US-2	As a user, I want to be able to have a field so that I can input my content.
US-3	As a user, I want to be able to have a submit button where I can click submit so that I can post my content.
US-4	As a user, I want to have an edit button so that I am able to edit my content when I want to fix something.
US-5	As a user, I want to have a delete button so that I am able to delete my post if I wish to.
US-6	As a user, I want to be able to sign up so that I could create my own account on the web application.
US-7	As a user, I want it to have an author field so that I am able to see who is posting which content.
US-8	As a user, I want the system to alert or not allow me to click submit if there is no content to be posted.
US-9	As a user, I want to be able to contact the admin if I have any questions about the website.
US-10	As a user, I want the system to be able to save the edited version so that only what has been edited is posted.
US-11	As a user, I want the latest post to show at the top so that I know which posts have been posted recently.
US-12	As a user, I want to be able to see all the latest posts on the front page so that I can be able to read them.
US-13	As a user, I don't want more than 5 posts to be posted on each page of the homepage so that it would not be too long to scroll.
US-14	As a user, I want the system to save my login information so that I may be able to sign up and access the account anytime without having to become a new user.

US-15	As a user, I want to have a logout button so that I may log off the web application once I have finish visiting it.
US-16	As a user, I want to have a user profile page to display my user activities.
US-17	As a user, I want to be able to see a list of the posts that I have created and manage the posts.

Table 3: User Stories

4.8 Product Backlog

After the user stories have been submitted to us by the stakeholders, I put the user stories into the following product backlog along with creating tasks that needs to be executed in order to ensure that the user stories are completed. The product backlog also indicates which user stories/ tasks have priority compared with one another. There is also a field that indicates the estimated hours that would be required by the team to complete each story point or task.

ID	User Story	Task	Priority	Estimated Hours
US-1	As a user, I want to have a login field so that I can be able to login to my account.	Create Login field Add login function	High	4
US-2	As a user, I want to be able to have a field so that I can input my content.	Create Post content field Add post function	High	5
US-3	As a user, I want to be able to have a submit button where I can click submit so that I can post my content.	Create submit button	High	4

US-4	As a user, I want to have an edit button so that I am able to edit my content when I want to fix something.	Create edit button Add edit function	High	4
US-5	As a user, I want to have a delete button so that I am able to delete my post if I wish to.	Create delete button	High	3
US-6	As a user, I want to be able to sign up so that I could create my own account on the web application.	Create Sign Up form Create Sign up Function	High	5
US-7	As a user, I want it to have an author field so that I am able to see who is posting which content.	Create user tag	Medium	3
US-8	As a user, I want the system to alert or not allow me to click submit if there is no content to be posted.	Create alert	Low	2
US-9	As a user, I want to be able to contact the admin if I have any questions about the website.	Create contact form	Low	3
US-10	As a user, I want the system to be able to save the edited version so that only what has been edited is posted.	Connect front end to back end	High	5
US-11	As a user, I want the latest post to show at the top so that I know which posts have been posted recently.	Create post chronological posting order	Low	3

US-12	As a user, I want to be able to see all the latest posts on the front page so that I can be able to read them.	Create post on home page	High	5
US-13	As a user, I don't want more than 5 posts to be posted on each page of the homepage so that it would not be too long to scroll.	Create pagination on homepage	Medium	4
US-14	As a user, I want the system to save my login information so that I may be able to sign up and access the account anytime without having to become a new user.	Connect backend user library with frontend	High	4
US-15	As a user, I want to have a logout button so that I may log off the web application once I have finish visiting it.	Create Logout button	Medium	3
US-16	As a user, I want to have a user profile page to display my user activities.	Create User profile page	Medium	4
US-17	As a user, I want to be able to see a list of the posts that I have created and manage the posts.	Create Post Lists with management tool	Medium	6

Table 4: Product Backlog

4.9 Sprint Story Board

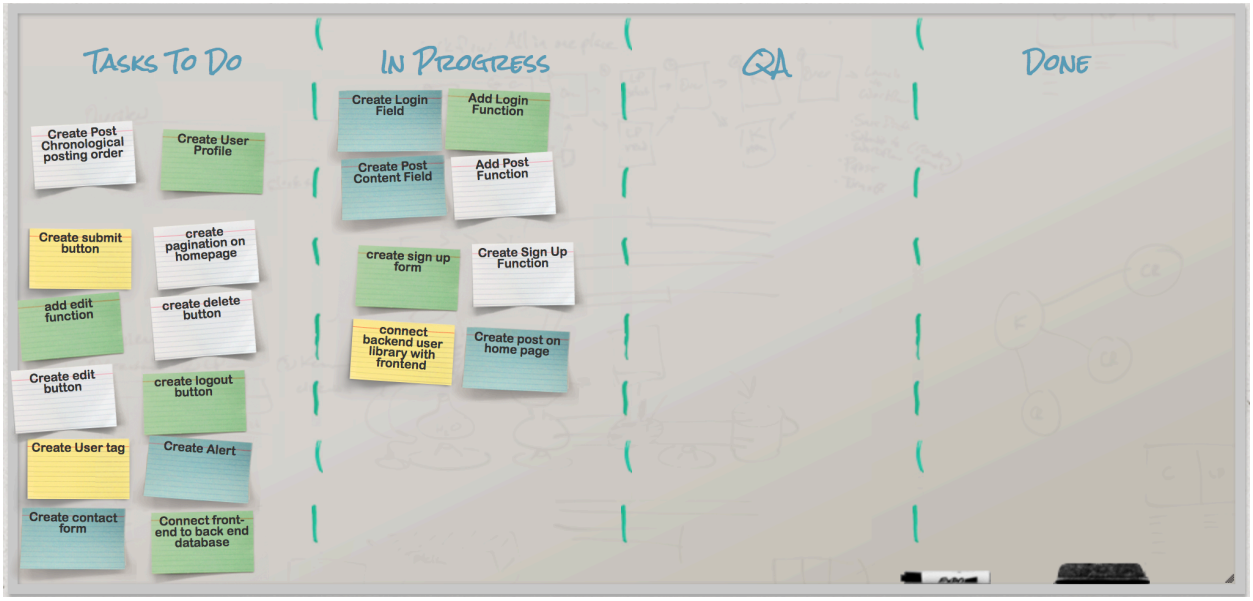


Figure 4: Sprint Story Board

4.10 Estimate time, cost and resources

In order to estimate time, cost and resources that are used throughout this project there are several methods that I took into account. Mainly time was a big concern throughout this project because there was a specific deadline that we had to meet. Most resources were readily available to us therefore; it was not a big factor in this project.

The project schedule was used to help estimate the time that was needed or needed to be allocated for each phase of the project. The project started in January 2018 and will end in May 2018. Time was estimated for the research phase, requirements design and requirements phase, the three sprints, and the release phase. The time was assigned and properly allocated to each phase based on how much work needs to be done for each of the phases.

As the project started, and user stories were created for the sprint runs, we assigned story points to prioritize each story into each of the sprint. We also created a product backlog where estimated time and priority were assigned to each of the user stories and tasks that were made for the user stories.

Both the developers and the stakeholders came together during the making of the product backlog in order to estimate and assign time to each of the tasks that were created. During the process, we rated each task based on how long we think each task would take to be completed. The average of the votes was taken and assigned to each tasks accordingly.

The release planning table were also used to keep track and monitor how much time and resources are needed for the project. We will be using the release planning table to set milestones for the project and make sure that everything will be going according to plan. As the project progresses into development we will be able to track and adjust the time if needed for each iteration.

4.11 Release Planning

Once the user stories and logistics have been planned in this section, I created a release plan schedule. The Release plan schedule indicates when each sprint would start and end and what would be released at the end of each sprint. The sprints all run for 3 weeks each or 15 days long for each sprint. Sprint I will focus on building the main CMS functionalities. Sprint II will be addressing more CMS functionalities that does not have as high of a priority as functionalities in Sprint I. Lastly, Sprint III will address enhancement features to the system. Each Sprint has user stories that are organized into it as well. The table shows the three sprint planning in greater detail.

Name	Type	Estimated Start	Estimated End	User Stories	Requirements
Sprint I	Main CMS functionalities	February 5, 2018	February 23, 2018	US-1 US-2 US-3 US-6 US-12 US-14	Initial release of the app with basic functional requirements
Sprint II	CMS functionalities	February 26, 2018	March 16, 2018	US-4 US-5 US-10 US-15 US-16 US-17	More functional/ nonfunctional requirements for enhancement
Sprint III	Enhancement Features	March 19, 2018	April 6, 2018	US-7 US-8 US-9 US-11 US-13	Enhanced features to satisfy other non-functional and business values.

Table 5: Release Plan Schedule

5.0 Development

The development phase will consist of three sprints or iterations. In each sprint there will be initial sprint planning where user stories and the system features to be built will be decided for that iteration. There will also be a managing section where it shows how the sprint will be managed and run throughout its course. The design phase will depict some of the functionality that was incorporated during that sprint. The testing section will show small acceptance tests that were done to ensure that all functionalities are working as it should. Daily scrum meeting will also be recorded and accounted for as well. Finally, we will end the sprint with a sprint review where we would go over the sprint backlog to ensure that the functionalities and user stories that we have planned for in the beginning are completed at the end of the sprint. Also, burndown charts will be used to show how we did during the sprint and see if there are ways we can improve for next sprint.

5.1 Sprint I

5.1.1 User Stories

User Story Code	User Story
US-1	As a user, I want to have a login field so that I can be able to login to my account.
US-2	As a user, I want to be able to have a field so that I can input my content.
US-3	As a user, I want to be able to have a submit button where I can click submit so that I can post my content.
US-6	As a user, I want to be able to sign up so that I could create my own account on the web application.
US-12	As a user, I want to be able to see all the latest posts on the front page so that I can be able to read them.
US-14	As a user, I want the system to save my login information so that I may be able to sign up and access the account anytime without having to become a new user.

5.1.2 Sprint Planning

For Sprint I of the project, I got with the stakeholders to determine which user stories were to go on for the first Sprint. The user stories were chosen based on its prioritization of how important they are based on functionalities. The first Sprint had user stories based on the main function of the CMS web application. The User stories that were chosen are listed in the user stories section. The development team then created a Sprint task board to keep track of their progress during the sprints. The Sprint task board is display below.

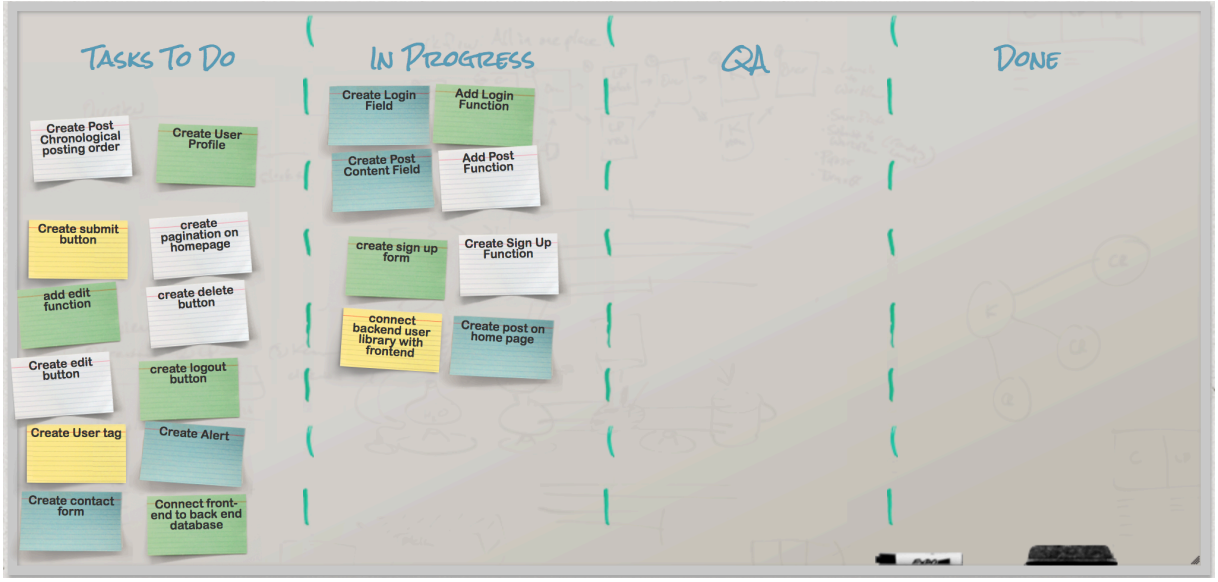


Figure 5: User Story Board for Sprint I

5.1.3 Daily Stand Up

Normally daily stand up meeting would be conducted with scrum master and development team members but since I am the only person working on the project, daily stand up meetings is a time for me to go over tasks that have been done and tasks that still needs to be done. During this stage I would still keep these three questions in mind to help guide the planning of the sprint:

1. What have I completed since last meeting?
2. What will I work on today?
3. What are some of the obstacles that is preventing me from meeting the sprint goals?

These 3 questions were asked in order to gauge my progress during the sprint. During each stand up issues would be addressed. I would also set daily task goals to be completed as well.

5.1.4 Designing

For the design of Community CMS web application, I started by creating the homepage of the web application. The homepage consisted of the navigation bar and a field where the posts are posted. The login field and a separate login page was created after. The homepage also had sidebar that had a list of the most recent post that were posted as well. The navigation bar has different navigation links that would be linked later when other functionalities of the web page are created. For this sprint the sign-up page with its functionalities were created and linked to the navigation bar. The field on the homepage will be able to display different posts that the user has created.

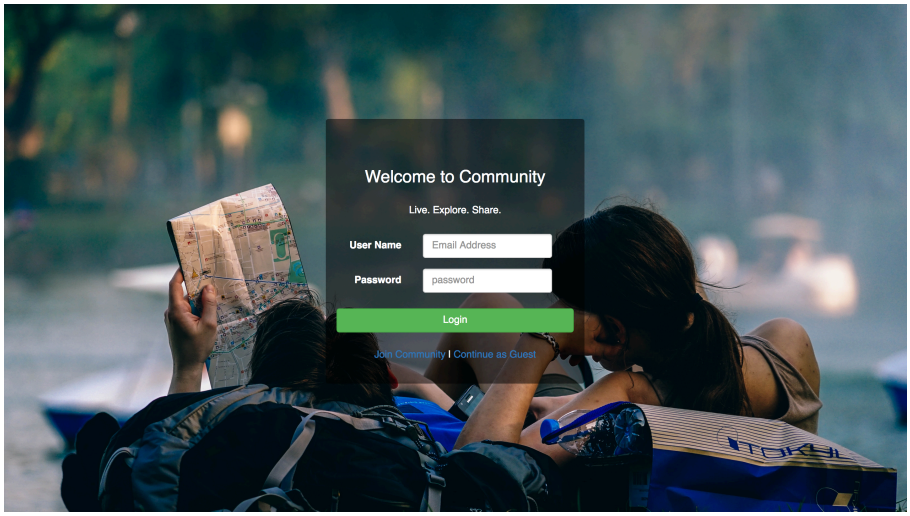


Figure 6: Community's Welcome Page Design

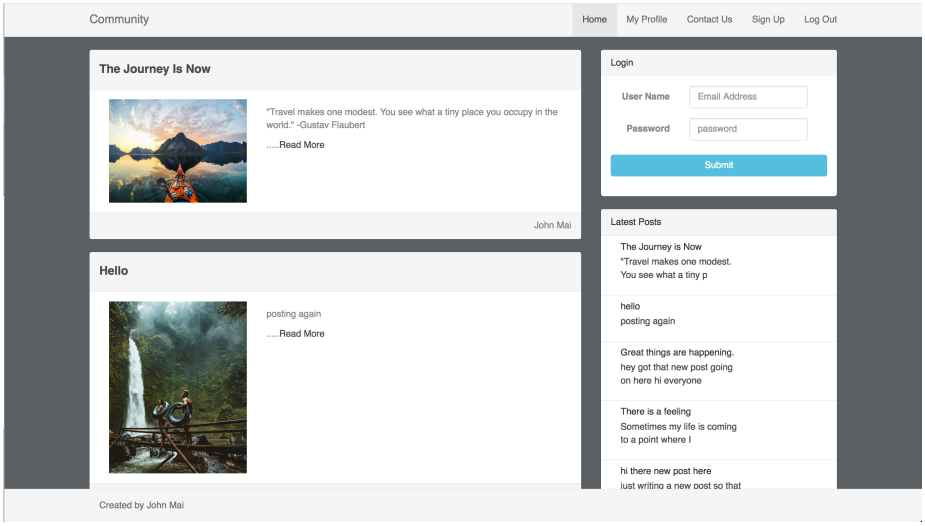


Figure 7: Community's Homepage Design

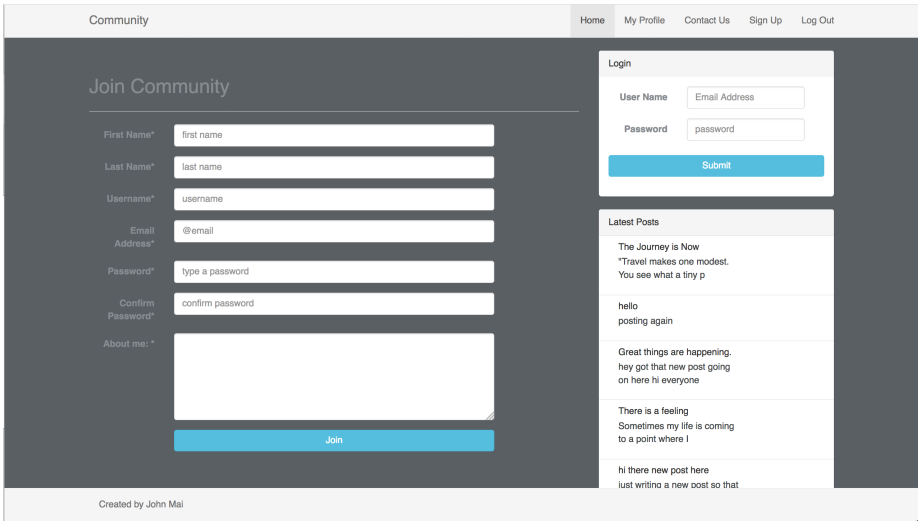


Figure 8: Community's User Sign Up Design

5.1.5 Testing

US-1

The login field and functionalities were created but had not worked initially since there were no data to use for sign up. The website would just allow anything to be "login" at the moment. Once the user data has been added the functionalities were able to log users in with the proper credentials.

US-2

A separate page was created to house the posting field with its functionalities added as well. After creating a post and sending it to the database the homepage where the posts are to appear was able to retrieve the data from the database and post it on the appropriate field in the homepage. The posting so far is posted with the latest posts posting after the older posts. This would need to be modified in later sprints so that the latest post will appear at the upper top most of the website. The title field and description field were able to be posted appropriately and corresponds with one another.

US-3

The submit button for the posting page was also created and when testing it out it was able to submit the posts to the database where the data is housed. The webpage was also able to retrieve the data and post the data back on the homepage as well.

US-6

Another separate page was created for the user sign up. The user is able to go in to the page and sign up as a new user. The fields are all required for the user to fill out therefore, if a field is not filled out the user will not be able to submit the form. Once all fields are appropriately filled out the user will be able to click the submit button to submit the form. The user information will be logged into the database in the user table. Users will be able to sign in once their credentials are submitted and the login function can retrieve the information from the user table.

US-12

With US-2 created the users may now see the posts that they have submitted in the posting page. The posts were able to post in the homepage appropriately without any big issues. The only issue that we ran into during this was that the post did not post the latest post at the top but rather after the older post at the bottom. Other than that the posts were able to post with the title of the post matching the description of the posts. The images were also posted appropriately as well.

US-14

This user story ensures that the user credentials have been saved to the user table of the database. The user is able to sign in when they type in their email address and their password. This means that the backend user table was able to link up appropriately with the front end allowing the login function to retrieve and match up the credentials appropriately.

5.1.6 Sprint Review

After Sprint I was finished I conducted a sprint review. In the sprint review I went over the story board to ensure that the tasks were completed and tested. I did run into minor issues during testing but all were fixed before the end of the sprint. I also looked over issues that I had during the sprint and wrote down recommendations to improve on for next sprint. The story board also shows that most of the tasks that were assigned for this sprint are either done or in QA. The burndown chart was also created to show how I did for Sprint I.

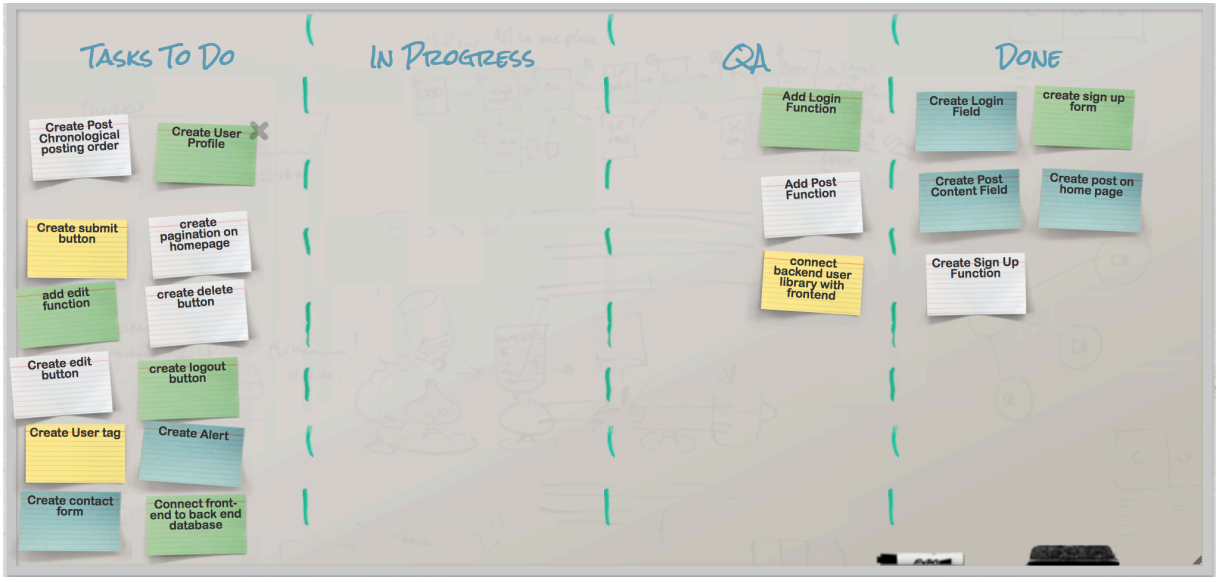


Figure 9: Sprint Review User Story Board

5.1.7 Sprint Burndown Chart

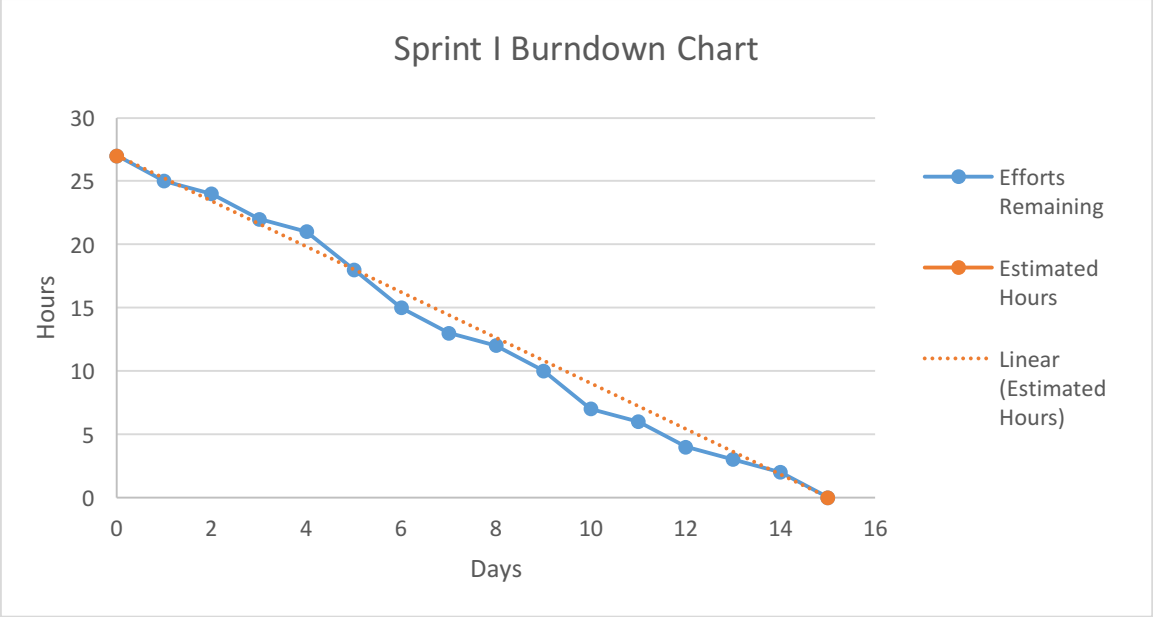


Figure 10: Sprint I Burndown Chart

The burndown chart shows the efforts that I needed to complete compared with the estimated hours that the I had initially came up with in order to complete all the tasks in the sprint. Each sprints ran for 15 days or 3 weeks minus the weekends. The burndown chart for Sprint I shows that for the most part I was on schedule with the estimated hours for Sprint I. There were some setbacks in day 4 that caused the graph to show a spike in the schedule but I was able to make it up in day 7 and day 10 to catch up to the schedule. I was able to finish the tasks by day 15 for Sprint I.

5.2 Sprint II

5.2.1 User Stories

User Story Code	User Story
US-4	As a user, I want to have an edit button so that I am able to edit my content when I want to fix something.
US-5	As a user, I want to have a delete button so that I am able to delete my post if I wish to.
US-10	As a user, I want the system to be able to save the edited version so that only what has been edited is posted.
US-15	As a user, I want to have a logout button so that I may log off the web application once I have finish visiting it.
US-16	As a user, I want to have a user profile page to display my user activities.
US-17	As a user, I want to be able to see a list of the posts that I have created and manage the posts.

5.2.2 Sprint Planning

Sprint II started following up on more of the features that are needed to be added to the web application. I picked out the User stories that I believe are of priority following Sprint I. In this sprint, I wanted to focus on improving some of the main functionalities while adding more functionalities to the website. The tasks were picked based on the user stories that I have chosen for this sprint. The story board shows which tasks were in progress during this sprint.

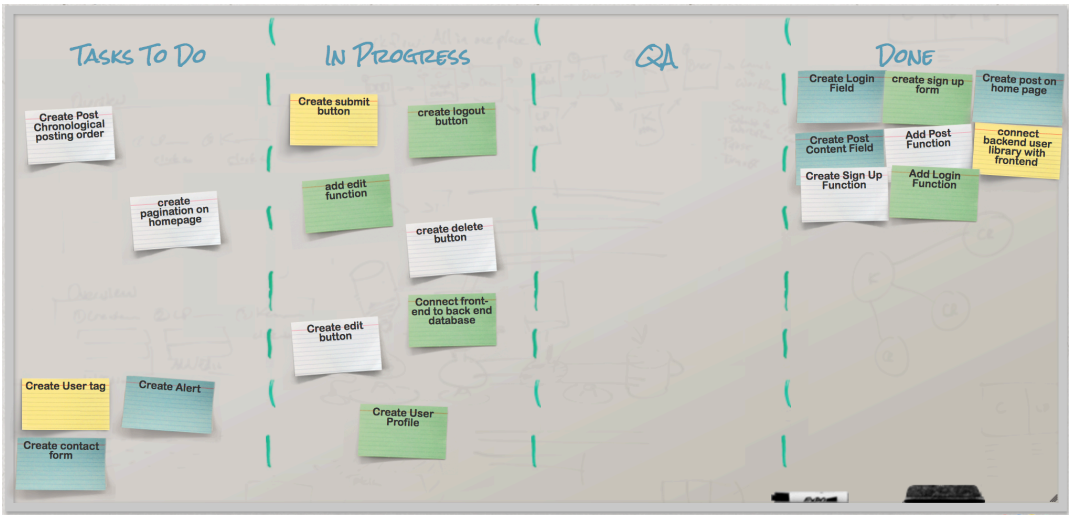


Figure 11: Sprint II Story Board

5.2.3 Daily Stand Up

For daily stand up meetings I went over the three questions that were mentioned to see what I will be working on and what I have already worked on. Also during this time, issues that arise during the development of the web application were also brought up in order to find solutions to help mitigate the problems. Most stand up meeting would consist of me setting daily tasks goals to be met for that day.

5.2.4 Designing

For Sprint II, the main thing that needed designing was the profile page. The profile page consisted of other tabs such as the main dashboard, the posting field, the post lists and the comment list. The posting page was moved from the home page to the profile page since it made more sense if a user want to be able to post something to go to their profile page to do so.

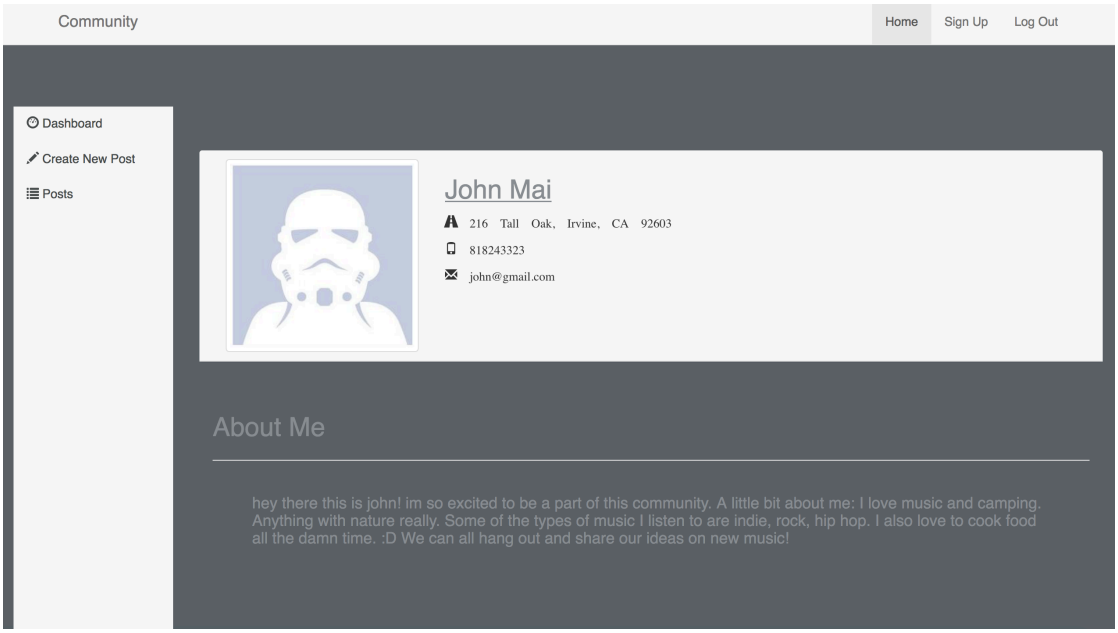


Figure 12: Community's User Profile Design

5.2.5 Testing

US-4

The edit button and feature was added to the post list section. The edit button was able to retrieve data of the post that it wants to be edit. Upon submitting the edited version of the post the system would create a new post instead of editing the old one. This feature was an issue for the system and I needed to look more closely into it.

US-5

The delete button was added to the post list section as well. the posts were able to delete when the delete button was clicked. The posts were taken off both the post list and also the homepage as well. All data of the post was cleared from the blog_posts table.

US-10

When testing this feature, the editing button was not able to post the edited version. Instead, the edited version was created and posted as a new post. With this problem the editing feature was placed into QA for much longer duration of Sprint II.

US-15

The logout button was created and implemented. When clicking on the logout tab the user will exit the web application.

US-16

The profile was created and displays user information, post listing, comment lists, and also users are able to create a new post under the new post tab as well. The post listing has different buttons where users can manage their post such as edit, view or delete the post altogether. The user profile can only be accessed once the user has been signed into their account.

US-17

The list post is created and when tested, the list is able to display all the posts that have been posted. Along with the list of posts it also indicates who the author of the post are and also allows users to edit, view or delete the post as well.

5.2.6 Sprint Review

For this sprint review I went over the issues that I had with this sprint. One of the main issue that was brought up during this sprint is the editing function. The editing function was not able to display proper edited post. It created a new post instead of edited the old post. I was well aware with the issue and was able to fix the issue.

During the review I also went over all the tasks to see if I had completed all the tasks that I have set forth during this sprint. All the tasks were once again completed in this sprint and were all tested. Some are still in QA such as the editing function but other than that most of the other tasks are moved over to being done on the story board. The story board below shows the tasks that were completed during Sprint II.

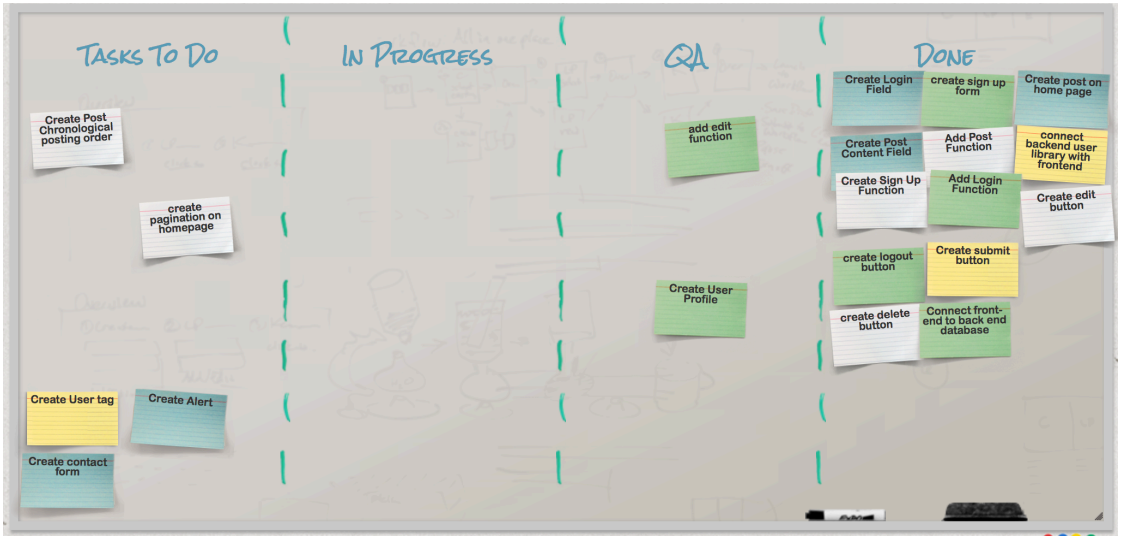


Figure 13: Sprint II Review Story Board

5.2.7 Sprint Burndown Chart

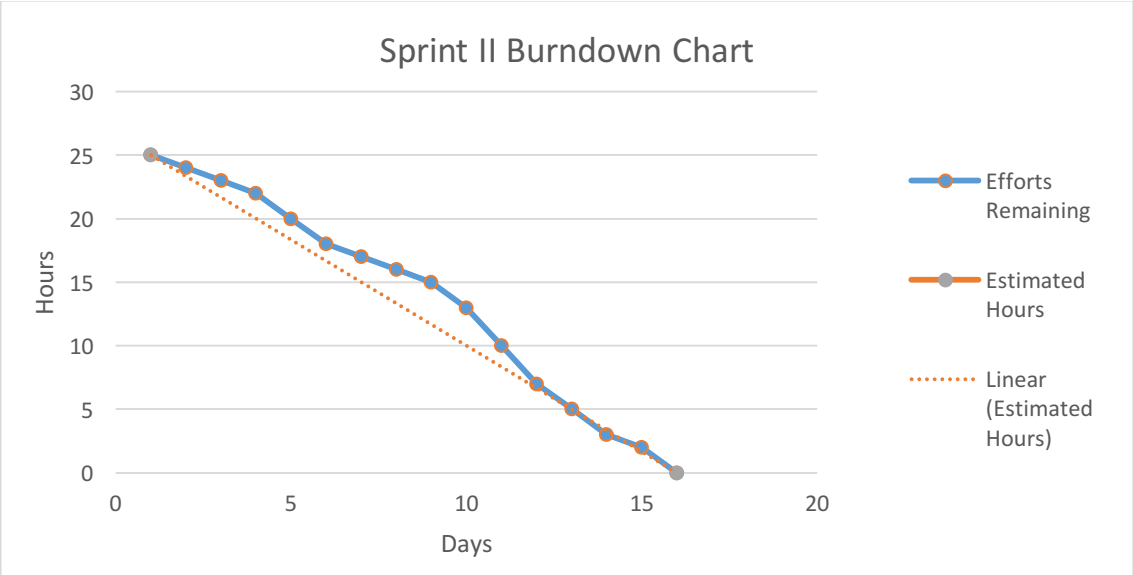


Figure 14: Sprint II Burndown Chart

During Sprint II, the burndown chart indicates that I was not on schedule for the most part during this sprint. I was a lot slower during the beginning of the sprint and then made up the slow effort at the end by catching up to the estimated schedule. Dealing with the editing function during this sprint caused the burndown chart to be a bit higher than the estimated hours.

5.3 Sprint III

5.3.1 User Stories

User Story Code	User Story
US-7	As a user, I want it to have an author field so that I am able to see who is posting which content.
US-8	As a user, I want the system to alert or not allow me to click submit if there is no content to be posted.
US-9	As a user, I want to be able to contact the admin if I have any questions about the website.
US-11	As a user, I want the latest post to show at the top so that I know which posts have been posted recently.

US-13	As a user, I don't want more than 5 posts to be posted on each page of the homepage so that it would not be too long to scroll.

5.3.2 Sprint Planning

Sprint III is the last sprint of this project. For this sprint I focused on enhancement features of the web applications. In this sprint, I had all the medium or low priority user stories that were left over be added to the sprint. These user stories are important to the CMS web application but they are not as important as the other user stories that were presented in Sprint I and II. These user stories had the lowest priorities and were fine being added at the end based on time constraints of the project.

The following user story board shows the remaining tasks that are needed to be completed during this sprint. All the other tasks are now in the "done" column while the remaining tasks to be completed are in the "in progress" section.

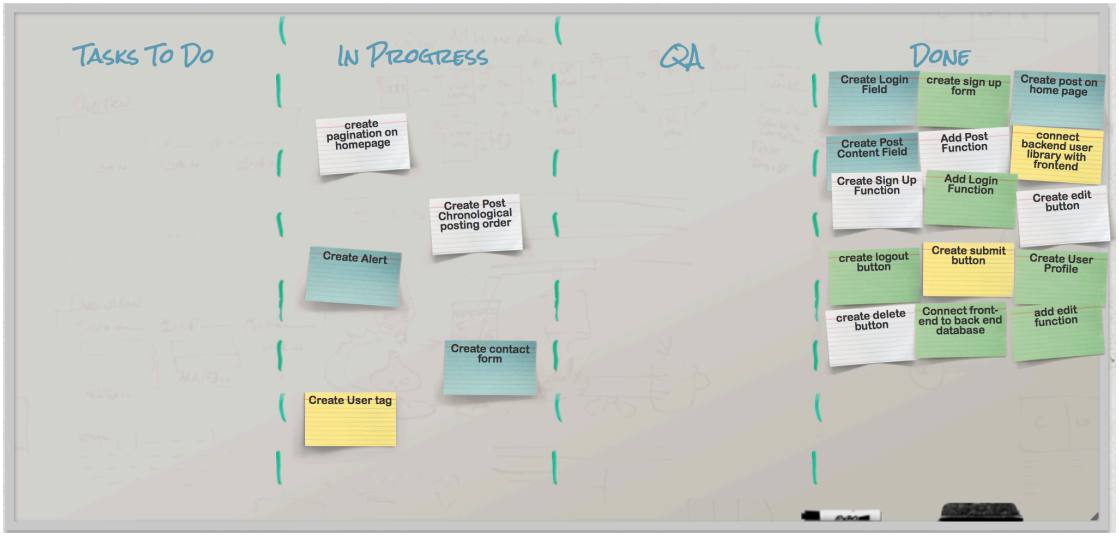


Figure 15: Sprint III User Story Board

For this sprint planning I also discussed about my plans for the acceptance testing of the overall system and also the release plan of the web application as well. Documentations of the system are created during this sprint as well.

The manual and the video demonstration was also planned for during this sprint cycle. Since all the work for the release will follow after the product has been built completely.

5.3.3 Daily Stand Up

During daily stand up I once again asks the three main stand up questions. At this point in the sprint I was on top of my duties and all the tasks properly planned to be completed. The remaining of the sprint was used to create documentations and video demonstration for the release of the product.

5.3.4 Designing

There was not much designing done during this sprint since all the designing and implementations were already done in previous sprints. The tasks in this sprint was to enhance certain features of the CMS web application. During this sprint a contact form was also created as well.

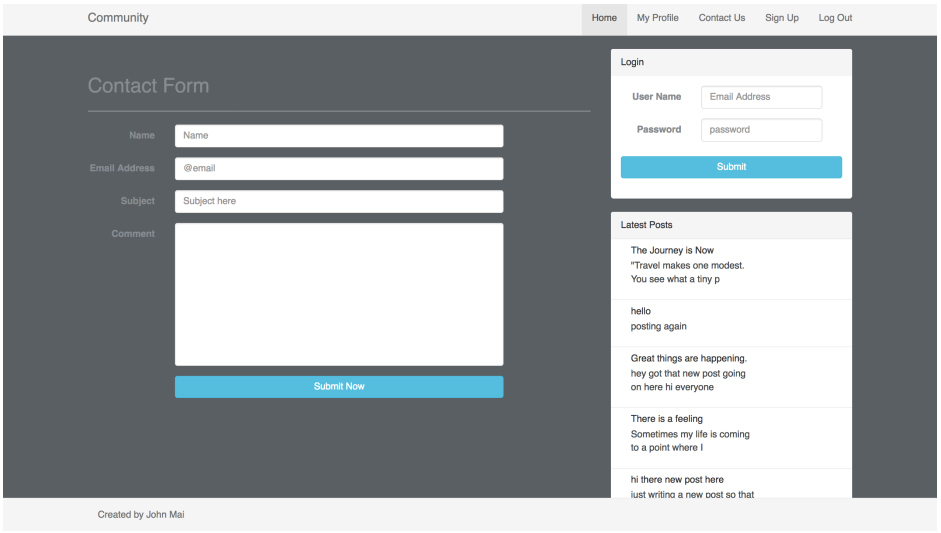


Figure 16: Community's Contact Page Design

5.3.5 Testing

US-7

The author field was added to the posts since it didn't show who posted the post before. While testing, the author field printed the email address of the poster instead of the name since it

grabbed data from the blog_posts table which did not have the name of the user on it. After joining the user table with the blog_posts table the user's first and last name was able to print on the posts instead.

US-8

Upon testing the posting and submitting forms function, the function would allow users to be able to submit even without anything to submit. In this sprint I made it a requirement to have something be written before being able to submit. Now when I tested the function out users were not be able to submit unless certain fields are filled in.

US-9

For this user story, a contact page was created. When testing out the function, the information that was submitted on the form was able to be sent to the contact table. There the admin may be able to look at the comments that users have left and/or export the table.

US-11

The posts list was always in descending chronological order. Upon addressing this user story in this sprint, the postings are now in ascending chronological order with the latest post being at the top of the homepage.

US-13

Adding this functionality, I had created a pagination function in which now after 5 posts on the homepage a new page will start. This helps taking away the clutter and having the homepage run endlessly.

5.3.6 Sprint Review

For this sprint review I went over all the tasks that I had completed. This sprint was fairly more easy to do since most of the tasks were to just implement enhancement features to the website. I was able to move all the tasks to the "Done" side of the story board. This sprint ran a bit faster than the other sprints since I didn't run into any issues at all.

The remaining of the sprint review was to go over the Release plan and make sure that the documentations were being completed as well as a video demonstration for the web application as well. An acceptance test was also planned so that I can make sure the web application run appropriately before handing it over to the stakeholders. Once the acceptance test is done the product will be able to be shipped off to the customer.

I also did a sprint retrospect for this sprint in order to address any issues that I have found during the whole project. I wrote down ways that I can improve for next project when it comes along. I also wrote down recommendations for enhancement for this project as well. Sprint retrospect was very informative and I was able to understand what needs to be done next project in order to improve the process.

Figure 17: Sprint III Review Story Board

5.3.7 Sprint Burndown Chart

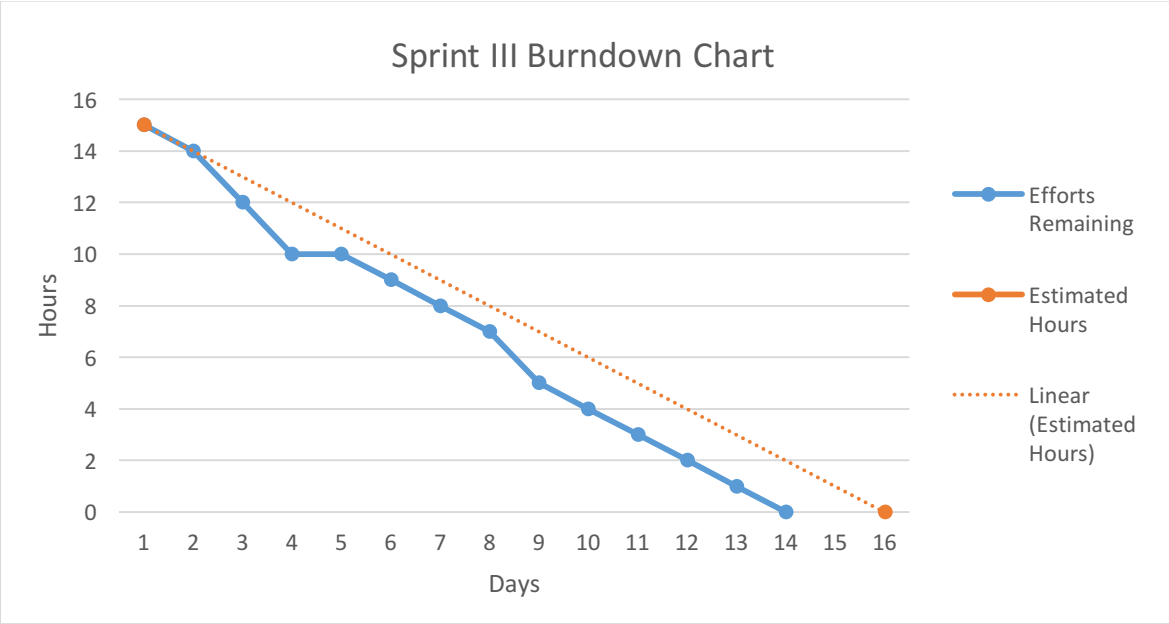


Figure 18: Sprint III Burndown Chart

In Sprint III, the burndown chart indicates that the development team was ahead of schedule. Since Sprint III only required the team to update or implement enhanced features to the web application, the team was able to finish the tasks early and ended on day 14 rather than day 15. The remaining time of Sprint III was used to work on documentation for the project.

6.0 Implementation

6.1 Source File Structure

The file structures below are organized into different functionalities of the web application. Some of the global variables that the files share are $_GET which is used to send data that is visible to everyone and is not as sensitive, $_POST is used to send sensitive data from a form, and $_SESSION which is used to store data that would be used throughout the web application such as when users are being login to their account and requires access to certain part of application.

User Login		
Variables:	**Data structure:**	**Description:**
$_POST $_SESSION $_GET $get_username $get_password	Username: (varchar) Password: (varchar)	For login, $get_username and $get_password talk to the database to and matches the password to the username. If username matches with password the $_SESSION allows the user to gain access to profile.

Sign Up		
Variables:	**Data Structure:**	**Description:**
$_POST $conf	fname: (text) lname: (text) username: (varchar) email: (varchar) password: (varchar) confirm_password: (varchar) aboutme: (text) $date: (date)	For new users, they may use this to sign up to become a new user to the web application. $_POST sends data that they have input into the form and $conf is used to ensure that the password they have typed matches with the confirm_password.

Posting Contents		
Variables:	**Data Structure:**	**Description:**
$_GET	post_id (int) title (text) image (text) description (text)	Contents can be posted to the main homepage and individual posts page through this file structure.

Create Post		
Variables:	**Data Structure:**	**Description:**
$_POST $_FILES $_SESSION	title (varchar) description (text) date: (date) user: (varchar) image_name: (varchar) image_tmp: (varchar) image_size: (int) image_ext: (varchar) image_path: (varchar) image_db_path (varchar)	this file structure allows user to be able to create a new post, the $_POST sends data to the blog_posts table after users have input and submit the data. $_SESSION is needed to allow users access to this page. $_FILES for the image file.

User Profile		
Variables:	**Data Structure:**	**Description:**
$_SESSION	user_first: (text) user_last: (text) user_email: (varchar) user_about: (text) user_image: (varchar) password: (varchar) user: (varchar)	Users will be able to gain access to user profile once login, $_SESSION shares the data to different pages of the web application.

Edit, View, Delete Post		
Variables:	**Data Structure:**	**Description:**
$_GET $_SESSION $_POST $_FILES delete_id edit_id post_id	title (varchar) description (text) date: (date) user: (varchar) image_name: (varchar) image_tmp: (varchar) image_size: (int) image_ext: (varchar) image_path: (varchar) image_db_path (varchar)	This file structure allows users to edit, view or delete. Delete_id uses the id to delete the specified post. Edit_id uses the specify id to retrieve the posts content in order to edit. Post_id is used to retrieve the specified post for viewing.

Contact Admin		
Variables:	**Data Structure:**	**Description:**
$_POST	name: (text) email: (varchar) subject: (text) comment: (text) $date: (date)	if users submit a contact form to ask questions or right comments this function uses the $_POST to send data to the contact table.

Database Data Structure:

These are the three main tables that are in the MySQL database. These three tables hold data for the blog postings, contact form and user information.

Table Name:	**Blog_posts**	**contact**	**user**
Table Data:	Id (int) Title (text) Description (text) Image (text) Date (timestamp) Author (text)	Contact_id (int) Name (text) Email (text) Subject (text) Comment (text) Date (timestamp)	User_id (int) User_role (text) User_first (text) User_last (text) User_name (text) User_email (text) User_password (text) User_about (text) User_image (text) Date (timestamp)
Description:	Table that holds the blog post content when user creates a post	Table that holds the comments and contact information when user use the contact form.	Table that holds the user information when user sign up for an account.

6.2 Reference List of Files

Community is a CMS web application. Users will be landing on the index.php page (welcoming page) in the Community folder upon arrival on the web application. The page will then lead users to either homepage.php which is the homepage or sign_up.php which is the sign up page

where users can create an account. The Profile folder, house all the user profile related functions such as creating, editing, viewing and deleting posts.

A reference list of files in folders they are presented in:

Community:

- *index.php:* welcome landing page of Community CMS.
- *homepage.php:* Community main homepage.
- *database.php:* database credentials to run database.
- *blogpost.php:* individual blog posting page.
- *sign_up.php:* user sign up page.
- *contact_page.php:* user contact page.
- *heading.php:* homepage navigation bar.
- *footer.php:* footer with creator name.
- *side.php:* side panel with login field and generated post lists.
- *login.php:* login functionality
- *logout.php:* logout functionality

Profile:

- *index.php:* user main profile page.
- *database.php:* database credentials to run database.
- *sideadminnav.php:* side navigation panel
- *heading.php:* navigation bar for profile page
- *createposts.php:* creating contents page
- *editposts.php:* editing contents page
- *postlist.php:* list of posts created with edit, view, delete functionalities.

7.0 Acceptance Testing

7.1 Testing

Acceptance testing is done after the CMS application has been built to test if all the functions performs as they should. Furthermore, acceptance testing helps ensure that all the user stories that the stakeholders have set in the beginning of the project has been met by this application. I came up with acceptance criteria that each user story should meet. When performing the test, I want to ensure that the acceptance criteria be met for each of the user stories. If the tests work as the acceptance criteria has stated, then the user story has been met and the test passes. The table below shows all the user stories and the acceptance criteria that I came up for each of them to meet. The result shows if they have pass the test or not with notes on any issues that I ran into while performing the tests.

ID	User Story	Acceptance Criteria	Result
US-1	As a user, I want to have a login field so that I can be able to login to my account.	• Login field has input for email and password • If password and email matches, then user get sent to the user profile page.	pass
US-2	As a user, I want to be able to have a field so that I can input my content.	• Posting field has a title field and a content/description field that users can write in. • users are able to upload images	pass
US-3	As a user, I want to be able to have a submit button where I can click submit so that I can post my content.	• A submit button at the end of the content field. • When submit is clicked on the content is posted on homepage immediately.	pass
US-4	As a user, I want to have an edit button so that I am able to edit my content when I want to fix something.	• Edit button by the post list. • Content is retrieved in the edit section • Users are able to change old content	pass

US-5	As a user, I want to have a delete button so that I am able to delete my post if I wish to.	• Delete button by the post list • When users click delete, the all of the post content pertaining to that specific post is deleted from database and homepage.	pass
US-6	As a user, I want to be able to sign up so that I could create my own account on the web application.	• Sign up page with input fields: first name, last name, username, email, password, about me. • Sign up content is added to the user table in SQL when submit is clicked on.	pass
US-7	As a user, I want it to have an author field so that I am able to see who is posting which content.	• Author who created the post; both first and last name is display at the footer panel of each posting.	pass
US-8	As a user, I want the system to alert or not allow me to click submit if there is no content to be posted.	• Users are unable to submit content when submit button is clicked. • Each field indicates that it needs to be filled in if there is no content in there.	pass
US-9	As a user, I want to be able to contact the admin if I have any questions about the website.	• Contact page with contact fields: name, email, subject, comments. • Comments content is added to contact table in SQL after form has been submitted.	pass
US-10	As a user, I want the system to be able to save the edited version so that only what has been edited is posted.	• When users click submit, the edited content will be posted in place of the old content. • The post on the homepage will be updated.	Did not pass. had to rewire the coding so that edit will update old content and not make a new posting. Pass second test after fixing bug

US-11	As a user, I want the latest post to show at the top so that I know which posts have been posted recently.	• Post organized chronologically in the homepage with the latest timestamp appearing at the top of the page.	Pass
US-12	As a user, I want to be able to see all the latest posts on the front page so that I can be able to read them.	• After post have been submitted, posts will appear in order on the front homepage.	Pass
US-13	As a user, I don't want more than 5 posts to be posted on each page of the homepage so that it would not be too long to scroll.	• No more than 5 posts appears on homepage. • Another page is created after 5 posts have appeared on homepage.	Pass
US-14	As a user, I want the system to save my login information so that I may be able to sign up and access the account anytime without having to become a new user.	• Login information is submitted when user sign up for a user account. • Login Information is sent to the user table in SQL.	Pass
US-15	As a user, I want to have a logout button so that I may log off the web application once I have finish visiting it.	• Logout tab present at the top right corner of the website. • When logout is clicked the user is taken away from the webpage. • User is not able to go into my profile page.	Pass
US-16	As a user, I want to have a user profile page to display my user activities.	• Profile page is linked to the "my profile" tab. • Profile page displays proper user information and about me section. • Profile page has tabs that allows user to manage their postings or view postings.	Pass
US-17	As a user, I want to be able to see a list of the posts that I have created and manage the posts.	• Post list is updated when new posts are added • Edit, view, delete tabs are by each post listing so users are able to manage the postings.	Pass

Table 6: Acceptance Test for Community

7.2 Results

The final results from the test indicates that the CMS web application was a success. After testing each features in the program to ensure that each user stories and their acceptance criteria has been met, I marked down on the table by each user stories if it had passed or failed. All the user stories pass without any issue except for one.

The user story on editing had an issue when it was being tested. The edited version would not post in place of the old version rather it had made a new post instead. Since the acceptance criteria was not met, I had to go back and fix this issue. Once the issue was fixed I ran the test again to test the editing feature. The second time the editing feature works properly and was able to meet the acceptance criteria. In the end, I was able to mark down that US-10 had pass the second time it was tested again.

Overall, following the Scrum-Agile software development plan really helped the project to run smoothly. The Scrum development allows for small release of the project where I was able to test each features individually and ensure that they would pass the acceptance criteria. If a feature did not meet the user story acceptance criteria, then they would need to be fixed or addressed immediately. The organization and methodology of Scrum-Agile framework allowed me to be very thorough with the development of the CMS web application.

8.0 Release

The Release phase is where the final product is ready to be delivered to the customer. The final product has been thoroughly tested and is ready to be sent out. This section shows the installation and operating instruction to help users and customer be able to use and operate the CMS web application.

8.1 Installation Instructions

Downloading the Source Code

- Retrieve source code from https://github.com/jmai11/597-Project-
- A github account will need to be created in order to be able to download the source code.
- Download the source code and save it into the htdocs folder of the Xampp application. (see Xampp installation)

Downloading Xampp

- Go to: https://www.apachefriends.org/index.html
- Download the version specific for the platform that you are using.
- Follow its installation guidelines.
- Make sure that Apache Web Server and MySQL database are running.

Retrieving and Importing Data into Database

- In a web browser go to localhost/phpMyAdmin
- Create a new database
- Retrieve file: cms_community.sql from the SQL folder in the source code
- Import this file into the database.

Database Configuration

- In the source files, navigate to the database.php file.

- There should be two files one in the main source file and one in the profile folder.
- Change the variables to the parameters that you have for your database.
    - $server= 'localhost';
    - $username='the username you have set up';
    - $password='the password you have set up';
    - $db= 'name of your database';

Running Community: CMS web application

- Open the Xampp control panel and start both Apache server and MySQL database.
- In a web browser, type: localhost/name of the folder you cloned from the source code and saved in htdocs of Xampp.
- A welcome to community page should display like in figure 19 below. You can now start exploring by either creating an account or visiting as a guest.

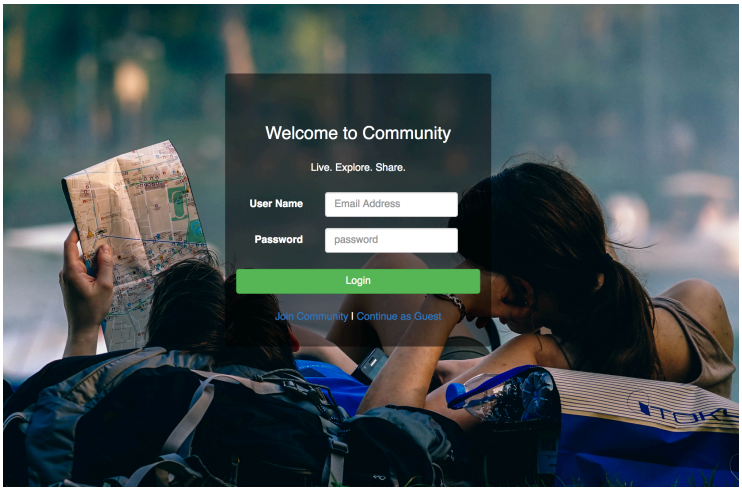


Figure 19: Community: CMS Welcoming Page

8.2 Operating Instructions

In order to run Community, ensure that Xampp is running in the background with Apache Server and MySQL database properly running as well. The following operating instruction sections are broken down into operating instructions for different features of Community.

Sign Up as New User

User can sign up by filling out the Join Community form under the Sign Up tab of the CMS. Or on the welcoming page they can click on Join Community which will take the new user to the same form. When done filling out the form, click submit to submit the form.

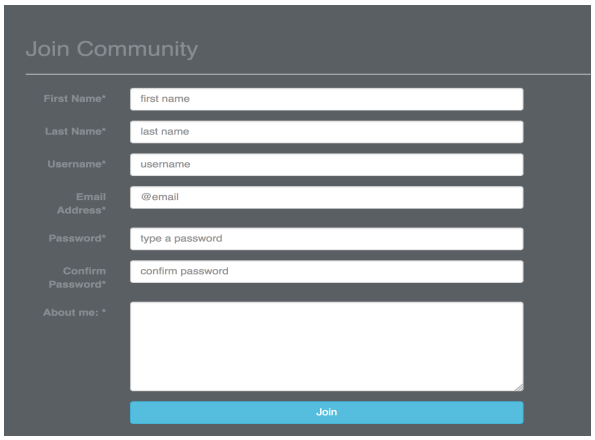
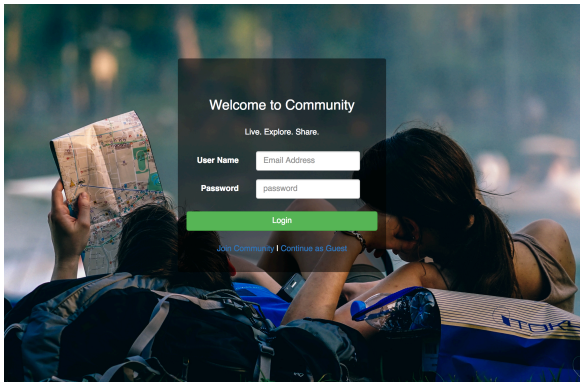


Figure 20: Sign Up Form

Sign In

Signing in can be done in two areas of the web application. The first is the welcoming page where users first land on the web application. The second login in option can be done on the web application main homepage. The user just need to type in their Email Address and Password in order to login.



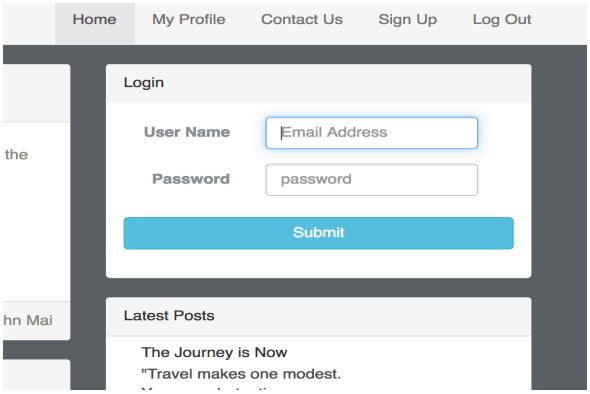


Figure 21: Login Fields to Sign In

Create a Post

To create a post, users have to be logged into their account. Only through the My Profile tab can a user be able to create a new posting. Once in My profile tab, the user can click on create new post from side menu tab to go to the Write to the Community page. Here, the user can add a title, description and images for the post. Once the fields are filled out click submit to post the content. The content will be displayed on the homepage.

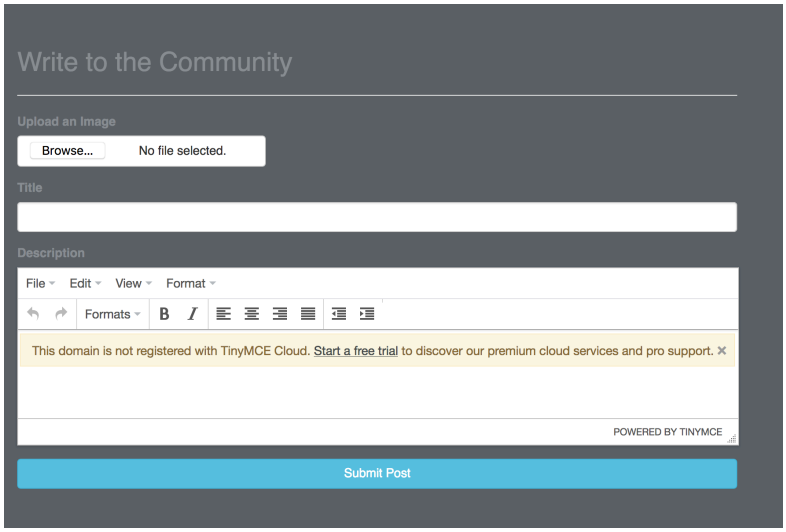


Figure 22: Creating a Post

Edit, View, or Delete Posts

In order to Edit, View or Delete posts users have to be signed into their account. Once in the user profile, the user can navigate to the Posts listing tab on the side menu. In there, the user will be able to see 3 different buttons, one to edit, view or delete. The user can click on one of the buttons and perform the function.

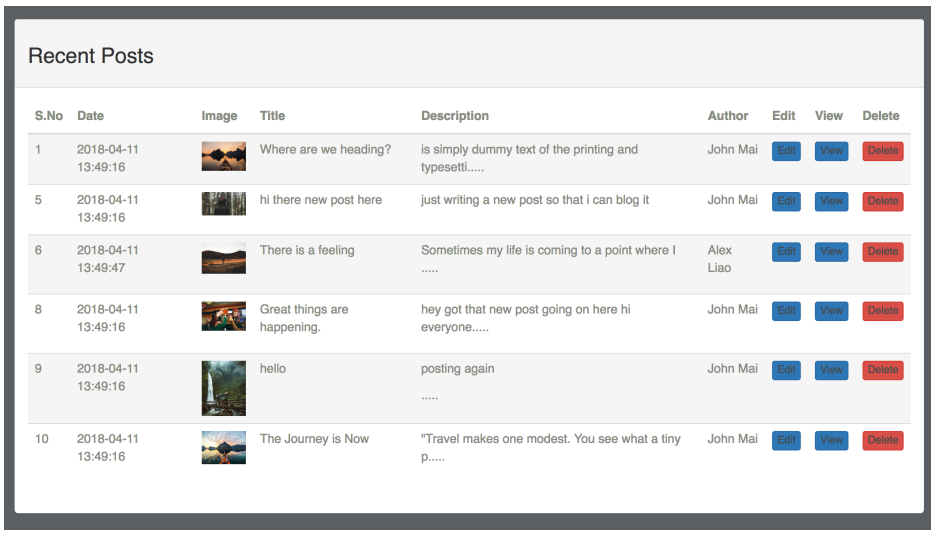


Figure 23: Edit, View, or Delete functionalities

Log Out

In order for user to log out of their account, the user can find a tab that says Logout in the top right hand corner of the web application. Clicking on that tab will log the user off of Community.

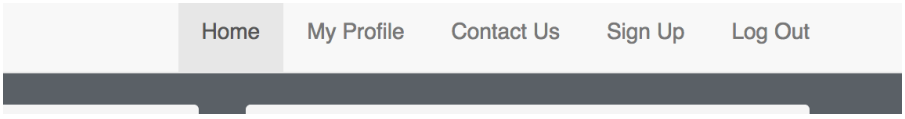


Figure 24: The Log Out function

9.0 Recommendations for Enhancement

Based on the overall project, in this section, I have proposed several recommendations for enhancement to the product. If this project was to be extended or continued in the near future I would most likely want to improve the product with these enhancements:

1. Improvement on profile page: Improve profile to allow users more control in updating their profile and managing/or editing their about me section. Users are stuck with the about me section that they have submitted when they sign up for an account.
2. Add commenting: Commenting should be added to the web application to allow dialogue and communications between users.
3. Improvement on design: update bootstrap to give the CMS web application a more posh and modern look and design.
4. Subscribe function: This function would allow users to be able to subscribe of follow other users as friend on the site.
5. Alert function: Alert function will alert users when a new post is added or specific users they are following has just posted something.
6. Add Admin feature: add an admin feature to allow user(s) to be able to moderate the CMS web application.
7. Embedding videos/audio files: Update the posting to allow embedding videos or audio files in post. Users are only able to add descriptions and images so far.
8. Add reporting bug function: This feature would be added to the contact form and would be good for web application maintenance.
9. Update Security features: allow users to choose their sign in feature for better security measures.
10. Privacy setting: allow users to be able to update or manage their security features such as blocking other users from seeing their contents or blocking unsolicited users from their contents.

10.0 Closing Statement

The CMS web application is a popular tool widely used among users and organizations alike. It helps promote and provide a space for people to be able to connect and share ideas with one another on the world wide web. By developing and implementing Community the CMS web application, it helped me to have a better understanding of the Software Development Life Cycle (SDLC). The development project was a great learning experience in which I was able to utilize the skills that I have learned throughout the course of this program. Some of the skills that I was able to utilized throughout this project were requirements gathering, architectural designing, planning and staging, creating user stories, implementing user stories for development, sprint planning, testing and deployment of the application. Furthermore, I developed the CMS by implementing an Agile Scrum development process. Scrum allowed me to create a working product with small releases at the end of each sprint. This allowed me to be able to test it more thoroughly throughout the development lifecycle and also ensure that it meets the requirements. The Scrum approach is good for software development as it promotes quick releases and stakeholders' involvement every step of the way.

Community CMS was built using a combination of front-end and back-end web development. The front-end development used were HTML, CSS, and Bootstrap framework. For the Back-end, I used PHP and SQL query language in order to send and retrieve data from the MySQL database. By developing the CMS I was able to utilize skills learned from programming in order to be able to write and code the CMS. The development process also allowed me to see how Sprint run plays into the development process of the software as well. By implementing user stories in each sprint, I was able to focus on prioritizing and completing the tasks that were important for the particular sprint.

Overall, the development project of a CMS allowed me to be able to utilize the knowledge and skills learned throughout the program. I was able to create a software product by implementing the software development life cycle from start to finish.

11.0 Acknowledgement

I would like to express my gratitude towards Professor Chang-Hyun Jo for his guidance and feedback throughout the course of this project.

I would also like to thank Professor James Choi for giving me an incredible opportunity by accepting me into the Master of Software Engineering program. He provided me with encouraging words and guidance during my application process into the program.

I am also very grateful and fortunate to be able to learn from insightful professors/faculty members throughout this program who truly are passionate about software engineering.

Lastly, I would like to thank my group/team that I have worked with from the start of this program to the end of this program. They made the journey and late night working on projects bearable. Keep striving for excellence!

12.0 Bibliography

1. Rouse, Margaret. 2014. *Content Manager System (CMS).* http://searchcontentmanagement.techtarget.com/definition/content-management-system-CMS

2. Graham, Adam. 2015. *CMS or ECMS or WCMS- What's the difference?* http://info.exsquared.com/ex-squared-blog/cms-or-ecms-or-wcms-whats-the-difference

3. D. B. Delgado, "Inspiring Teamwork & Communication with a Content Management system" *in Proceedings of the 35th annual ACM SIGUCCS fall conference*, Orlando, 2007, pp. 55-59.

4. Cunningham, Ward. 2001. *Manifesto for Agile Software Development.* http://agilemanifesto.org/

5. Wiegers, K. Beatty, J. 2013, *Software Requirements Third Edition,* Microsoft Press, One Microsoft Way, Redmond, Washington 98052-6399.

6. Stellman, A. Greene, J. 2015, *Learning Agile,* O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

7. L.S. Prakash, N.S. Krutti, A. Sajeev. "Review of Challenges in Content Extraction in Web Based Personalized Learning Content Management Systems" *2010 ACM*

8. Mountain Goat Software. 2018. *Scrum.* https://www.mountaingoatsoftware.com/agile/scrum

9. Scrum, 2018. *What is Scrum?* https://www.scrum.org/resources/what-is-scrum

10. Wikipedia, November 2015. *Multitier Architecture.* https://en.wikipedia.org/wiki/Multitier_architecture#Web_development_usage.

11. Wikipedia, November 2015. *Client-Server Model*. https://en.wikipedia.org/wiki/Client%E2%80%93server_model.

12. W3Schools, 2018. *PHP 5 Tutorials.* https://www.w3schools.com/php/default.asp

13. Apache Friends. 2018. *XAMPP Apache + MariaDB + PHP + Perl.* https://www.apachefriends.org/index.html

Appendix A

GitHub Repository

The source code to this project is hosted on GitHub. This includes all the files and documentation for CPSC 597 Project course.

To view or download the source code please visit:

https://github.com/jmai11/597-Project-