Software Requirements Specification

for

MemeHub

Version 1.0 approved

**Prepared by Holiday Garrison
Tia Malley
Ashlyn Martin
Maggie Pettus**

CSE 4214 / 6214 – Introduction to Software Engineering

9/16/2022

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed overview of Group 3's social media platform, Memehub Version 1.0. It will describe the functions of the subsystem, the environment in which the software will operate, constraints that will limit the product's development, the features that will be included in the user interface, and the attributes and requirements that will be incorporated into the product.

1.2 Document Conventions

When reading this document, certains words will be in different typefaces and font sizes. For readability, titles and subtitles are bolded and have larger font sizes. Any italicized words or abbreviations will be defined in Appendix A: Glossary.

Every requirement for the system is to have its own priority.

These priorities are defined as:

- High: It is a critical requirement that the product must perform.
- Medium: It is a core functionality that the product should perform.
- Low: It is a functional enhancement that is not explicitly stated that the project should have.

1.3 Intended Audience and Reading

This document is intended for the stakeholder to understand the functionality and scope of the project and for the developers to assist in the development of the project. For the stakeholder, the order for reading this document should be Part 1 - Introduction, Part 3 - System Features, Part 4 - Other Nonfunctional Requirements, Part 2 - Overall Description, and Part 5 - Other Requirements, while skipping to Appendix A: Glossary for the definition of italicized words and abbreviations. The development team should read over the entire document, but jump to any section relevant to the task they are accomplishing. Below is a brief description of each part of the document:

- Part 1 - Introduction
  - This section offers a summary of the benefits, objectives, and goals for Memehub Version 1.0.
- Part 2 - Overall Description
  - This section describes how the system will be developed. It includes information on the functions, user classes, operating environments, and constraints.
- Part 3 - System Features
  - This section includes the core features that will be implemented on the Memehub website, as well as step-by-step instructions on how these features will work.
- Part 4 - Other Nonfunctional Requirements
  - This section discusses the performance, safety, and security requirements required for the project. It also notes the quality attributes that will be achieved in the software.
- Part 5 - Other Requirements
  - This section covers any additional requirements not covered elsewhere, such as database requirements.

- Appendix A: Glossary
    - The section includes additional information for italicized words and abbreviations that may be helpful to readers.
- Appendix B: Analysis Models
    - This section includes any pertinent analysis models.
- Appendix C: To Be Determined List
    - The section includes a list of the to be determined references for the project.

1.4 Product Scope

Social networking sites are becoming more significant in today's time, as they allow friends to interact via the internet. MemeHub is a social media platform designed for users to share their favorite memes among their friends. The main objective and benefit of this product is to provide an environment for friends to connect through comedy.

This platform will allows users to:

- Create and manage a personal account.
- Compose and edit posts that will appear on their profile and friends' timelines.
- Send and accept friend requests.
- See friends' posts on their timeline.
- Interact with friends by liking, disliking, and commenting on their posts.
- To share post of their friends.

1.5 References

- https://getbootstrap.com/docs/5.0/getting-started/introduction/
- CodeIgniter4 User Guide — CodeIgniter 4.2.6 documentation

2. Overall Description

2.1 Product Perspective

MemeHub is new, self contained web based software. It will be designed with future developments and companion systems in mind, though no such systems exist at this time.

2.2 Product Functions

MemeHub's major functions are as follows:

- The user is able to create and manage an account. The user is able to log in and out of this account.
- The user is able to create a post that is saved to their profile and added to their followers' timelines.
- A user with an account is able to add other users as friends and have their friends' posts appear on their timeline.
- A user is able to interact with others' posts, with actions such as liking, disliking, and commenting.

2.3 User Classes and Characteristics

Guest: A guest or non-account holding user is a user of MemeHub that has yet to create an account These users have the ability to create an account. They can view the profiles and posts of account holding users. The goal is to impress these users so that they will create an account with MemeHub.

Account Holding User: An account holding user is a user that has created an account with MemeHub. This user is able to log in and log out of their account. Account holding users are able to view other users' profiles and posts. While logged in, the user is able to manage their account, make posts, add other users as friends, and view a timeline of posts created by the user's friends. It is important to satisfy these customers, so that they will continue using MemeHub.

Admin: An admin will be able to manage all users and posts. They are expected to have a high knowledge of the programming languages and structures used to develop MemeHub and have at least a basic understanding of its structure.

2.4 Operating Environment

MemeHub will run on browsers that are HTML5 compatible. It will be developed in PHP 8.1 with the framework CodeIgniter4. The HTML will be styled using Bootstrap 5.0, which includes JQuery for some functionality. The needed database tables will be managed through PHPMyAdmin. MemeHub will be hosted on an Apache 2.4 server.

2.5 Design and Implementation Constraints

Data storage will need to consider the large possible number of posts and users that could be stored and whose information will need to be retrieved quickly. Tools are limited to what is freely available to the developers. Time is limited based on a development schedule set by stakeholders. Code will need clear documentation and have well-commented code for future maintenance that is not managed by the development team.

3. System Features

3.1 Account Creation and Login

3.1.1 Description and Priority

High Priority:

- A user must be able to:
  - Create a new account.
  - Login, and logout of said account.
  - Edit and delete said account.
  - This account must be secured with a username and password specified by the user at the time of account creation.

3.1.2 Stimulus/Response Sequences

Account creation:

- User clicks the “create account” button.
- System presents the user with an account creation screen.
- User enters required information into textboxes and confirms account creation.
- System stores user data in the database.

Login:

- User clicks the “login” button.
- System presents a login screen.
- User enters information into textboxes.
- System matches the user-entered data to an account in the database.
- System loads relevant data from the database and presents the user with a timeline page.

Logout:

- User clicks the “logout” button.
- System restricts user status to “guest.”

Edit:

- User clicks the “account” button.
- System presents the account page.
- User clicks the “edit account” button.
- System presents a page to edit the details of the user’s account.
- User enters desired information into the textbox associated with the account detail they wish to edit.
- User clicks “save” to confirm edits.
- System updates the data in the database.

Delete:

- User clicks the “account” button.
- System presents the account page.
- User clicks the “delete account” button.
- System asks for confirmation.
- User clicks “I want to delete my account.”
- System deletes all user-associated data from the database.

3.1.3 Functional Requirements

REQ-1: System has two “states” a user can be in: guest (not logged in or does not have an account), and account-holder (logged in).

REQ-2: System can restrict content based on the state of the user: guests cannot access anything other than the homepage.

REQ-3: System has functional database for storing user and post information.

REQ-4: An account-holder cannot access another user’s account (cannot edit/delete posts of other users).

3.2 Posting to Timeline

3.2.1 Description and Priority

High Priority:

- A user must be able to:

- ○ Compose new posts to their timeline.
- ○ Edit and delete said posts.

3.2.2 Stimulus/Response Sequences

Creating Post:

- (From timeline page) User clicks “create new post” button.
- System presents post creation screen that allows user to upload image and write a description about it.
- User clicks “choose image” button and chooses a file to upload.
- User enters text into description text box.
- User clicks “Post!”
- System stores the post data in the database.
- System refreshes the user’s timeline with the new post at the top.

Editing Post:

- User clicks the three-dot button on the top-right of the post.
- System presents dropdown of two options: edit and delete.
- User selects the “Edit” option.
- System presents post edit screen where the user can change the image and/or description.
- User edits the post as desired and clicks “Save Changes” button.
- System updates post information in the database.
- System shows an updated post.

Deleting Post:

- User clicks the three-dot button on the top right of the post.
- System presents dropdown of two options: edit and delete.
- User selects the “Delete” option.
- System asks user for confirmation.
- User clicks “Yes, I want to delete this post.”
- System deletes the post from the database and refreshes the timeline.

3.2.3 Functional Requirements

REQ-5: System must be able to process the following widely-used image file formats:

- JPG/JPEG
- PNG
- GIF

REQ-6: System must be able to link each post to its user.

REQ-7: System must be able to preserve all post data in database.

3.3 Send / Accept Friend Requests

3.3.1 Description and Priority

Medium Priority:

- A user must be able to:

    - ○ Send friend requests to other users of MemeHub.
    - ○ Accept friend requests from other users of MemeHub.
    - ○ The state of being "friends" must allow users to interact with each other in such ways as are addressed in **Feature 3.4.**

3.3.2 Stimulus/Response Sequences

Finding Other Users:

- User clicks on the "Find Friends" button at the top of their timeline page.
- System presents a page that lists all of the registered users.
- User clicks on a user's name from the list.
- System presents that selected user's profile.

Sending Friend Request:

- (From the profile of another user) User clicks the "Send Friend Request" button.
- System registers the request and informs the current user that the request was sent.
- System sends a notification to the requested user to inform them the request was sent.

Accepting Friend Request:

- System will create an alert next to a "Friend Requests" button on the user's timeline page.
- User clicks the "Friend Requests" button.
- System presents the "Friend Requests" screen which lists the name and username of users that have sent the current user a friend request.
- The user can then choose to "Accept" or "Decline" each request by clicking the respective buttons next to each request listed.
- Upon "Accept" the system will add that user to the current user's friend list, enable timeline interaction, remove that request from the list of friend requests, and update the database accordingly.
- Upon "Decline" the system will remove that request from the list of friend requests and update the database accordingly.

Removing a Friend:

- User clicks the "Friends" button from their timeline.
- System presents a page that lists all of their friends.
- User clicks the "Remove Friend" button next to the friend they want to remove.
- System asks for confirmation.
- User clicks "Yes, remove this person."
- System removes that user from the current user's friend list, updates the database, and refreshes the page.

3.3.3 Functional Requirements

REQ-8: System must be able to keep an updated list of all users for Find Friends page.

REQ-9: System must be capable of handling requests from multiple users at a time.

REQ-10 System must be able to link each request to the user that sent it.

REQ-11: System must keep track of each user's friends in the database.

REQ-12: System must enable privileged interactions between users who are friends.

REQ-13 System must disable privileges interactions between both users when a user unfriends someone.

3.4 Interaction With Friends

3.4.1 Description and Priority

Medium Priority:

- A user must be able to:
  - Like, dislike, comment on, and share the posts of their friends to their own profile.

3.4.2 Stimulus/Response Sequences

Liking a Post:

- User clicks the thumbs-up icon on a post.
- System registers the click and adds the user to those who have already liked the post in the database.
- System updates the page, increments the like-count for that post.

Unliking a Post:

- User clicks the thumbs-up icon on a post they have already liked.
- System registers the click and removes the user from those who have liked the post in the database.
- System updates the page, decrements the like-count for that post.

Disliking a Post:

- User clicks the thumbs-down icon on a post.
- System registers the click and adds the user to those who have already disliked the post in the database.
- System updates the page, increments the dislike-count for that post.

Undisliking a Post:

- User clicks the thumbs-down icon on a post they have already disliked.
- System registers the click and removes the user from those who have disliked the post in the database.
- System updates the page, decrements the dislike-count for that post.

Seeing Who Likes a Post:

- User hovers over the thumbs-up icon.
- System presents floating dropdown of users who have liked the post.

Seeing Who Dislikes a Post:

- User hovers over the thumbs-down icon.
- System presents floating dropdown of users who have disliked the post.

Commenting on a Post:

- User clicks the “Leave a comment” button on a post.
- System presents a textbox under post with “Send” and “Cancel” buttons.
- User enters a comment into the textbox and clicks “Send.”
- System adds the comment to the database.

- System updates list of comments to include the user's new comment.

Editing a Comment:

- User clicks the three-dot button on their comment.
- System presents dropdown of two options: edit and delete.
- User selects "Edit."
- System restores the user's comment to an editable state and presents "Save" and "Cancel" buttons.
- User makes the desired change and selects "Save."
- System updates the comment content in the database and refreshes the page.

Deleting a Comment:

- User clicks the three-dot button on their comment.
- System presents dropdown of two options: edit and delete.
- User selects "Delete."
- System asks for confirmation via a pop up.
- User selects "Delete" on pop up.
- System removes the comment from the post in the database and refreshes the page.

Sharing a Post:

- User clicks "Share" button on the post.
- System asks for confirmation.
- User clicks "Yes, share this to my profile."
- System updates the user's profile with the post.

3.4.3 Functional Requirements

REQ-14: System must be able to link likes to a specific user.

REQ-15: System must be able to link comments to a specific user.

REQ-16: System must keep a record of the users who have liked a post.

REQ-17: System must keep a record of the users who have disliked a post.

REQ-18: System must keep a record of all comment data.

REQ-19: System must be able to display all comments for each post.

REQ-20: System must sort comments by time/date created.

REQ-21: System must be able to copy posts from one user's timeline to another (sharing).

4. Other Nonfunctional Requirements

4.1 Performance Requirements

We will account for bugs and instances in which our website could crash. One instance that a social media website could malfunction is if a link or a page does not exist anymore, in cases like these we will generate a 404 error page to display, so the customer will know why that link is not working. In the event that images will not load due to poor internet connection or non-existence, we will make sure every picture has an alternate text assigned to it, so the customer will at least know what the image is about. We will also try to keep the server free of collisions in order to preserve the quality of the website–in doing so the website will load more efficiently. In order to reduce the hang up of gif memes and to reduce processing demands on the users' devices, we will limit the file size that the user can upload. The website will also have a Contact Us function at the bottom of the page to contact the developers to let them know of any issues they may come across.

4.2 Safety Requirements

There is always a risk that a site could be hacked, so in the user agreement license, we will include a warning about the danger that the user could lose data or that any information they put on the site could be open to a data leak. However, to prevent said leaks, we will use encryption keys. In fact, PHP already has a built-in encryption function, so the users' data is already protected. The customers will be directed to look at the IEEE document on safety because that is where we will be basing our requirements.

4.3 Security Requirements

To keep the users' data from being open and prone to hacks we will be using PHP which has its own built-in encryption. In fact, PHP Blowfish uses hashes to create and generate encrypted keys for passwords.

4.4 Software Quality Attributes

- **Adaptability:** The website will account for any image or text file the user may want to input for their desired meme.

- **Availability:** Each version will be completed by the designated deadline to create a reliable and steady timeline for stakeholders to keep up with.

- **Correctness:** Using HTML, CSS, JavaScript, SQL, and PHP the data will properly commit to the website in the correct manner.

- **Flexibility**: Developers will have at least one upload and develop at least one update to the website each year to account for any maintenance needed in the ever growing development of technology.

- **Interoperability**: To allow the website to fulfill this software attribute, developers will continue to build off of previous versions of the software, therefore legacy systems will still be able to run the website on a previous version that is compatible to that specific system.

- **Maintainability:** We will implement, organize and comment on the code used to create the website, so it can be easily maintained and updated by later development teams and those who did not help in the construction of version 1.0.

- **Portability**: The website will be developed in HTML5 in order to maintain compatibility with all major browsers with a specialized mobile view. This will allow users to chose their preferred browser and easily access the website at a large range of screen sizes.

- **Reliability**: The website will be monitored 24/7 to make sure the website doesn't falter or any issue does not go unresolved for a certain amount of time.

- **Reusability**: The website will be designed to be easily accessed and easily navigated, developers will call in elderly users to interact with the website as a test case to prove that the website is easily accessible.

- **Robustness:** Before opening the website to users, developers will troubleshoot and run multiple test cases on the website to account for and prevent any errors that could occur on release.

- **Testability**: In order to test a single problem that will not affect all of the other aspects of the websites, developers will create a copy of the said problem and work on it on a local host instead of the server, once the issue is resolved the developer will then upload the fixed problem onto the main back-end development of the website.

- **Usability:** We will allow the database to hold a certain number of customers and once that number is close to being met, we will expand the service by adding more servers to the project to account for the extra users.

5. Other Requirements

Will be completed at a later time.

Appendix A: Glossary

Will be completed at a later time.

Appendix B: Analysis Models

Will be completed at a later time.

Appendix C: To Be Determined List

Will be completed at a later time.