# Software Requirements Specification

for

# MyFace

Version 1.0 approved

Prepared by: Jacob Mitchell,
Nicholas Manning, John Ferguson

Intro to SE Fall 2022 Group 8

September 16, 2022

# 1. Introduction

## 1.1 Purpose

- The Product being created is a Facebook-like social media clone, called MyFace. This SRS document covers product version 1, with this SRS document covering the entire scope of this product.

## 1.2 Document Conventions

- The properties of each section are indented with '-', with any pertinent information relating to each property of the section being indented once more with '-'.

## 1.3 Intended Audience and Reading

- This document will be useful for both internal operations, such as developers and project managers, as well as end users and testers.
    - Useful sections for developers/managers:1.1-1.5, 2.1-2.5, 3.1-3.5, 4.1-4.4, 5
    - Useful sections for end users/testers: 1.1-1.3, 2.1, 2.4, 5

## 1.4 Product Scope

- The goal of this software is to create a social networking site that has many similar features to current market options. This will enable users to connect with each other through various methods as long as they have a MyFace account.
- The functions expected of the final MyFace product are listed in detail under section 2.2.

## 1.5 References

- MyFace will use the Django framework for backend related structures, a reference to the django framework can be found here.
    - https://www.djangoproject.com/

# 2. Overall Description

## 2.1 Product Perspective

- MyFace is a new web application unrelated to existing systems. The purpose of this SRS document is to define the requirements for the web application and all of its components.

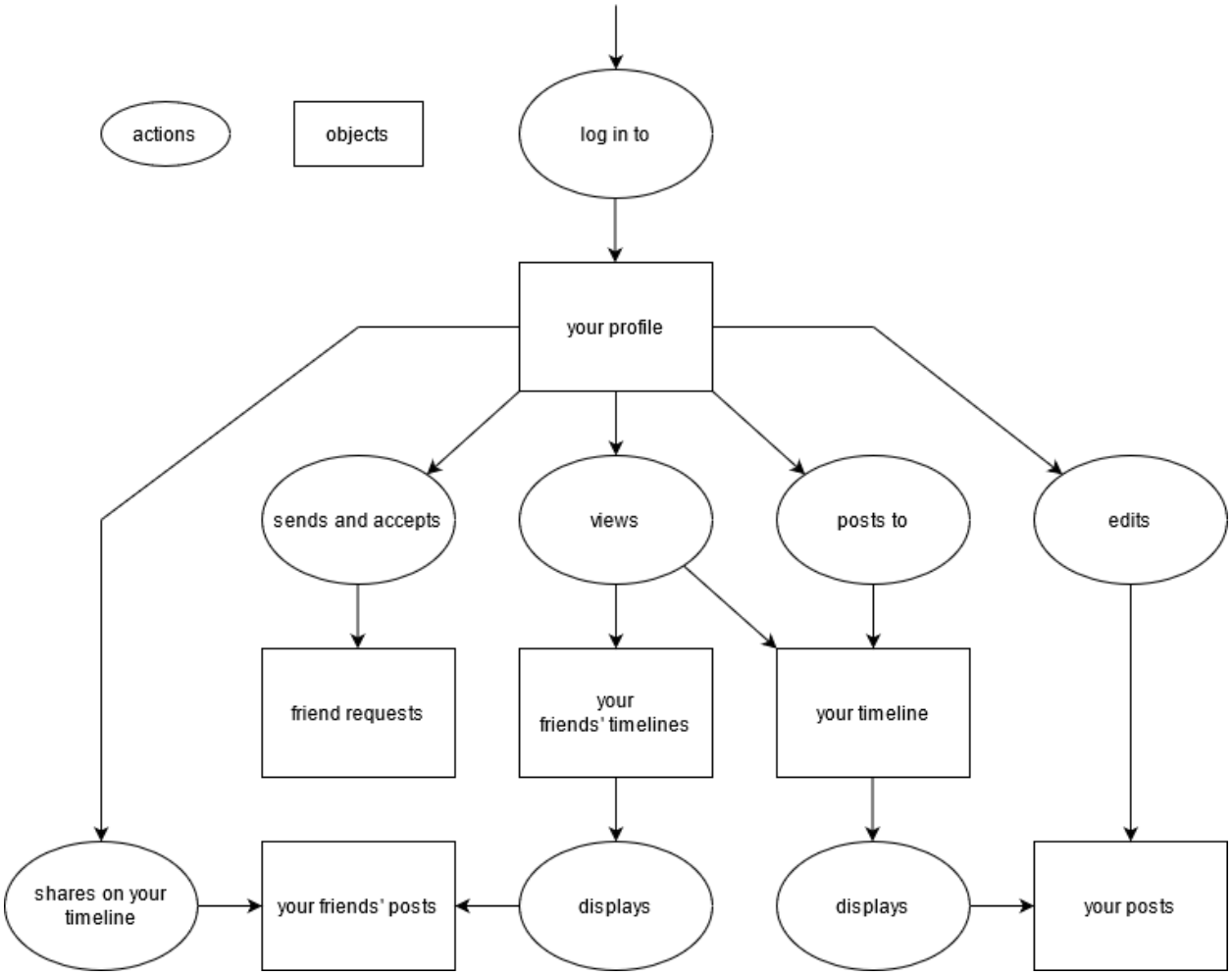
## 2.2 Product Functions

### 2.2.1 System Functions

- MyFace should have a login page where users can log in using a username and password.
- MyFace should support connections between user profiles, and another user's profile that is connected to one's own will be referred to as a friend.
- MyFace should store user profile information and friends in a database.
- MyFace should be able to display a user's timeline.

### 2.2.2 User Functions

- A user should be able to log into his or her account to be able to connect with his or her friends and access their timelines.
- A user should be able to compose, edit, and post statuses on his or her timeline.
- A user should be able to send and accept friend requests.
- A user should be able to like or comment on the status/timeline post of their friends.
- On their own timeline, a user should be able to share a friend's status/timeline post.




## 2.3 User Classes and Characteristics

The User is the only user class that must be implemented for MyFace. The User is expected to be internet literate and be able to use a web browser. Users may perform the functions as described in section 2.2.2.

### 2.4 Operating Environment

The server-side components of the website must operate within a Windows operating system environment. The client-side components of the system must operate within common web browser environments.

### 2.5 Design and Implementation Constraints

The developers are to implement the backend components of the web application using the Django framework. The developers may use a dedicated frontend framework but are not required to do so. MySQL is the database management system to be used.

## 3. System Features

### 3.1 System Records

3.1.1 Description and Priority

- The system must maintain a way of storing user information.
    - This includes name, login information, and other personal information in regards to the user.
- The system must maintain the selected username for users and display it within their own page.
- Priority: High

3.1.2 Stimulus/Response Sequence

- User signs up for a profile -> profile information is inserted into database
- User posts a status -> status information is inserted into database

3.1.3 Functional Requirements

- REQ-1: When a user signs up for a profile, the relevant information should be inserted into a database table for user profiles, and each profile should be uniquely identified by the username selected by the user.
- REQ-2: When a user posts a status, the relevant information should be inserted into a database table for user statuses, and each status should be uniquely identified by a status ID number.

### 3.2 User Registration

3.2.1 Description and Priority

- On the login page, the system should allow users to select to register to create an account.
- Priority: High

3.2.2 Stimulus/Response Sequence

- User clicks the register button on the login page -> User is redirected to the registration page, enters information into the form, and submits. The new User is added to the database.

3.2.3 Functional Requirements

- REQ-1: MyFace should have a button to redirect to the registration page on the login page.
- REQ-2: The registration page should have a form to enter information for new accounts.
- REQ-3: When the form is submitted, a new account should be created in the database using the information entered into the form.

## 3.3 User Login

3.3.1 Description and Priority

- The system must allow for users to input their login information and access the application.
- Priority: High

3.3.2 Stimulus/Response Sequence

- User inputs login information -> system responds by either accepting/denying the login success

3.3.3 Functional Requirements

- REQ-1: MyFace should have a login page that one can access via a web browser.
- REQ-2: The login page should have fields where a user may enter their username and password.
- REQ-3: MyFace should check whether the username exists in the database and whether the input password hash is the same as the password hash in the database that corresponds to the username.
- REQ-4: If the username does not exist or the password hash in the database does not match the input password hash, then then the login page should display “Incorrect Credentials”.
- REQ-5: If the username exists in the database, and the input password hash matches the password hash in the database, the user should be redirected to their timeline page, and a session should be created to keep the user logged in. Once logged in, the user may view their own timeline page and the timeline pages of his or her friends.

## 3.4 User posts

3.4.1 Description and Priority

- The system must be able to process a post from user
- The system must be able to post to the platform
- The system must allow for the post to be shared to other users on the platform
- Priority: High

3.4.2 Stimulus/Response Sequences

- User clicks on the option to allow them to post -> System responds by opening the window for a post to be made
- User inputs sentences into post window -> System processes that input
- User clicks on the post option -> System takes user input sentences and posts them

3.4.3 Functional Requirements

- REQ-1: On one's own timeline page, there should be a field where a user enters a string to be the content of a status
- REQ-2: When the content string is submitted, a new post appears on their timeline.
- REQ-3: Edit Status
    - On a status on one's own timeline page, a user should be able to select to edit the status.
    - When a user selects to edit a status, a field should appear that contains the existing status' content string.
    - The user should be able to submit the new status content string
        - If the status content string is different from what was posted before, the status content string should be replaced, and the status should display "(edited)" to show that it has been changed
        - If the status content string is the same as it was before, the status content string does not need to be replaced on the status, and the status does not need to display "(edited)".

## 3.5 User post limitations

3.5.1 Description and Priority

- The system should provide a character limit to the user for each post
- Priority: Medium

3.5.2 Stimulus/Response Sequences

- User creates a post -> post content is compared against a database of profane words
- User post contains no profanity -> post is allowed
- User post contains profanity -> post is not allowed

3.5.3 Functional Requirements

- REQ-1: Profanity filter
    - A database of profane words and profane statements/phrases will be set up to allow the websites checking of profane posts. This will allow the website to be friendly to users of all ages, and prevent speech that would be considered hateful towards others.

### 3.6 User post interaction

3.6.1 Description and Priority

- The system must allow for users to upvote, or downvote a post
- The system must allow for users to write comments on the posts of other users
- The system must allow for users to share the posts of others on their own timeline
- Priority: High

3.6.2 Stimulus/Response Sequences

- User clicks on a particular post -> system gives the user an option to like the post
- User clicks on a particular post -> system gives the user an option to share the post

3.6.3 Functional Requirements

- REQ-1: Like Status
    - On a status on another user's timeline, a user should be able to select to like the status.
    - The status should keep up with what users have liked the status
    - If a user has already liked a status and selects to like the status again, the status should be unliked by the user.
    - A status should display the number of users that have liked the status.
- REQ-2: Share Status
    - On a status on another user's timeline, a user should be able to select to share the status.
    - When a user selects to share a status, that status should be posted on the sharing user's timeline.
- REQ-3: Comment on a Status
    - On a status on any timeline, a user should be able to select to comment on the status
    - When a user selects to comment on a status, a form with a submit button should appear
    - When a string is typed into the form and the submit button is clicked, a comment should appear under the status

### 3.7 User to user interaction

3.7.1 Description and Priority

- The system should allow for one user to see a list of other users within the platform
    - A search ability would be preferable
- The system should allow for one user to befriend another user
- The system should allow the user to view the posts of another user
- Priority: High

3.7.2 Stimulus/Response Sequences

- User clicks on the profile of another user -> system allows user to view the timeline page of the other user

3.7.3 Functional Requirements

- REQ-1: On one's own timeline page, MyFace should have an input field where a user can input the username of another user to send the other user a friend request.
- REQ-2: The friend request should appear on the recipient's timeline.
- REQ-3: If a friend request appears on a user's timeline, the user should be able to select "yes" or "no" to accept the request.
- REQ-4: If a friend request is accepted, then the sending and receiving user should become friends, allowing them to access each other's timeline page.

### 3.8 User logout

3.8.1 Description and Priority

- The system should allow for the user to log out of the application
- Priority: High

3.8.2 Stimulus/Response Sequences

- User clicks on "exit page" -> system exits page of the visited user
- User clicks on "log out" -> system logs user out of account and awaits new login information

3.8.3 Functional Requirements

- User clicks on the logout option and the user is disconnected from the session.

## 4. Other Nonfunctional Requirements

### 4.1 Performance Requirements

- Windows based operating system
- Compatible on all processor types
- Compatible web based browser (Google Chrome, Safari, FireFox, Microsoft Edge)
- MyFace aims to handle many concurrent users active at once

### 4.2 Safety Requirements

- Profanity filter
    - Filter posts that may contain profane wording
- Explicit filter
    - Filter posts that may be explicit in nature
- Safe lighting for those who are prone to seizures
    - darkmode

### 4.3 Security Requirements

- User data privacy
    - Make sure that user's personal information cannot be pulled from the website in an inappropriate and or unauthorized manner
    - Users account activity is not visible unless both users are 'friends'
- User post privacy
    - User posts that are private cannot be viewed by other unauthorized users
- User login privacy
    - User login information is kept secured
- User location privacy
    - Unless publicly provided by user
- User profile privacy
    - Users profile can be set to restrict all incoming 'friend' request, or not viewable in searches at all

### 4.4 Software Quality Attributes

- Ease of use over ease of learning
    - MyFace use is intended to be intuitive after learning software functions
- Availability over adaptability
    - Web based, not created solely for an individual operating system
- Availability over portability
    - Web based, not application based
- Reliability
    - MyFace aims to be reliable on all web systems
- Usability
    - MyFace aims to be user friendly and intuitive to use

## 5. Other Requirements

- MySQL Database
    - Stores usernames
    - Stores user password hashes
    - Stores user posts

### Appendix A: Glossary

- Friend - A users friend is someone added through MyFace, where friends are able to see and react to each others post
- Post - A post can contain text or media, displaying to your friends basically 'what you're up too'
- Reactions (Likes) - Reactions to users post let them know you 'like' what they're doing
- Share - Sharing your own status, or a friends status, will simply copy the text/media contained within that status and post it as your own, crediting the user that created the original post

### Appendix B: Analysis Models

- See section 2.2.2.

## Appendix C: To Be Determined List

- Not yet applicable.