Fau CEN4010 Fall 2021

9/28/2021

Back Together

Project Team 24

Team Members:

Thomas Giglitotti (tgigliotti2020@fau.edu)

Kevin McInerney (kmcinerney2016@fau.edu)

Jonathan Silva (silvaj2020@fau.edu)

Thomas Moraes (tmoraes2020@fau.edu)

Conor Coleman (conorcoleman2016@fau.edu)

Milestone 1 Project Proposal and High-level Description

# Table of Contents

[**Table of Contents**](#_ruregyuqszsz) **2**

[**Executive Summary**](#_xp6j6svq9p16) **2**

[**Competitive Analysis**](#_9uivkfpm5hp0) **3**

[**Data Definition**](#_olxdvdpew2yb) **4**

[**Use Cases**](#_f8iwvlngnypd) **6**

[**Functional Requirements**](#_jz4au6a8koqm) **10**

[**NonFunctional Requirements**](#_kb9o97po37gn) **14**

[**System Architecture**](#_ht7vfhoygwnf) **17**

[**Team**](#_751xnu7e7yuj) **18**

[**Checklist**](#_pvb432kdngr6) **18**

# Executive Summary

Our project, titled “Back Together” is a website classed as social media, with the intention of getting users from their respective cities and neighborhoods back together (as the title implies), in a post-COVID world. To elaborate, as the pandemic comes to a close, this would serve as a method for producers of live events, such as venues/ticket companies (i.e., Ticketmaster), local city governments, universities, small businesses, charities, etc. to have a centralized solution for posting and encouraging users within a particular geographical area. Another goal of the project would be to give users the ability to post about and discuss events and other local happenings in light of COVID coming to an end.

We would accomplish these goals through a few systems. From the perspective of the user, you would create an account, log in, and be presented with a feed of local happenings from sources outside of the website, as well as any posts made directly to the website from its users. From the view of the feed, there will be access to comments about each post. You will also have the ability, for internally made posts, to message another user about an event. Another option will be to filter the feed, as well as search for events.

All website users will also have customizable profiles as is standard on most social media-based websites, where one can detail themselves and set a picture. This can help users identify events and posts from like-minded individuals.

# Competitive Analysis

|  | **BackTogether** | **NextDoor** | **Facebook** | **Ticketmaster** | **Eventbrite** |
| --- | --- | --- | --- | --- | --- |
| **Ability for user to view nearby events** | Yes | Yes | Yes | Yes | Yes |
| **Ability to filter feed** | Yes | Yes | No | Yes | Yes |
| **Forum/feed for regular users** | Yes | Yes | Yes | No | No |
| **Heavily location-based user demographics** | Yes (User-input ZIP code) | Yes (User-input address) | No (but does collect location) | No (Optional) | No (Optional) |
| **Ability to directly contact an event host** | Yes (Private message) | No | Yes (Limited) | No | No |
| **External Event Postings** | Yes (RSS feed) | No | No | No | No |

# From this table, it is conclusive that we will provide a combination of features best suited for a user to connect with other users and event hosts. One major missing feature would be ticket sales, however, this may be best handled by the external event providers as payment processing is outside of the scope of this project.

**NextDoor** is a platform aimed at connecting locals, however, it is not aimed in nature at connecting people to events, nor does it really have the intent of bringing people together post-COVID. We have a more direct intention with our features, such as the event feed.

**Facebook**, as a social media platform, doesn’t have the outright intention of connecting people, however it has such a wide scope that it often does so. We are more directly aimed at giving local users a chance to meet each other in a group setting.

**Ticketmaster** and **Eventbrite** are primarily ticket sales websites which also provide a user the ability to see events in their area, however, these events are always limited to events within the venues that use these services, and they don’t provide any social media capabilities.

# Data Definition

| **Term** | **Definition** |
| --- | --- |
| Website | The service as a whole, accessed over the internet. |
| User | Any person accessing the website. |
| Member | Any user who has made an account and logged in. |
| Non-member | Any user who has not made an account. |
| System | The website itself, internally. |
| Event Host | A user responsible for the planning and marketing of an event. |
| Event | A local (paid or free) attraction or gathering accessible to the general public. |
| Home Page | The “landing zone” for any User not logged in. |
| Feed | A collective listing of information sent by users, including event hosts. |
| Post | Textual information sent by a user to the feed. |
| Comment | Textual information sent by a user as a response to a post on the feed. |
| Profile | A page or modal containing information submitted by the user pertaining to themselves. |
| Private Message | Textual information sent by a user directly to another user, without being displayed to the feed or a third party. |

# Use Cases

Use Case - Create Post

* Overview:

A user comes to the system and is attempting to create a post. This can be any of either a general post, event, or you can offer and request services. The user will click on the designated button and select which option they wish to post and then fill in the required information.

1. Description:

Use case will describe how a user can create a general post, even, or services post.

1. Actors:
   1. User
   2. System
2. Preconditions:
   1. The user is connected to the Internet
   2. System is available
   3. User has an account and is logged in
3. Primary Flow:
   1. User arrives at Website Home Page
   2. User logs in or creates an account
   3. User selects to post
   4. Website will display an area to select the post type and input the information
   5. User selects which type of post to create
   6. User enters the desired information and submits
   7. System will confirm the post
   8. Terminate Use Case Post
4. Alternative Flows:
   1. User is not logged in
      * If user selects post (4c) without being logged into their account
      1. The website will request the user to login or create an account
      2. User will log in or create an account
      3. Continues to (4d)
   2. User does not have an account
      * If user selects post (4c) without having an active account
      1. Systems asks the user to create an account with the site
      2. Returns to (4a) if the user declines to make an account
      3. User selects to create an account
      4. System displays the account creation form
      5. User completes and submits the form
      6. System creates an account in the database
      7. Continues to (4d)
   3. Post exceeds maximum length
      * If while filling information the text exceeds the maximum length
      1. Webpage will display that the comment is exceeding length limit
      2. User will have to shorten the text
      3. Continues to (4f)

Use Case - Comment

* Overview:

The user comes across a general post that they would like to comment on. The user would have arrived at the home page and be logged in and then select a post to comment on. Then the user will enter their comment not exceeding the maximum and submit.

1. Description:

This use case will describe the process for a user to reply to a post with a comment.

1. Actors:
   1. User
   2. System
2. Preconditions:
   1. The user is connected to the Internet
   2. System is available
   3. User has an account and is logged in
3. Primary Flow:
   1. User arrives at Website Home Page
   2. User logs in or creates an account
   3. User selects a post to comment on
   4. Website displays an area to comment in
   5. User types the comment and submits
   6. System will confirm comment
   7. Terminate Use Case Comment
4. Alternate Flows:
   1. User is not logged in
      * If user selects to comment (4c) without being logged into their account
      1. The website will request the user to login or create an account
      2. User will log in or create an account
      3. Continues to (4d)
   2. Comment exceeds maximum length
      * If while filling information the text exceeds the maximum length
      1. Webpage will display that the comment is exceeding length limit
      2. User will have to shorten the text
      3. Continues to (4f)

Use Case - Private Message

* Overview:

The user can come across another user whether through past encounters or after seeing a post that they would like to message about something. After the user decides they want to message someone then they would select the user they want to message and then select to message them. This can be useful for posts about services so you can talk directly with the person supplying it. After the message is typed the user will submit and await a response.

1. Description:

This use case will show how one user can direct message another on the system.

1. Actors:
   1. User
   2. System
2. Preconditions:
   1. The user is connected to the Internet
   2. System is available
   3. User has an account and is logged in
3. Primary Flow:
   1. User arrives at Website Home Page
   2. User logs in or creates an account
   3. User decides they want to message another user
   4. User selects the other user and selects to message
   5. System will display a text box for the user
   6. User enters the message and selects submit
   7. System will confirm the message
   8. Terminate Use Case Private Message
4. Alternate Flow:
   1. User is not logged in
      * If user selects to message (4d) without being logged into their account
      1. The website will request the user to login or create an account
      2. User will log in or create an account
      3. Continues to (4d)
   2. Private message exceeds maximum length
      * If while filling information the text exceeds the maximum length
      1. Webpage will display that the comment is exceeding length limit
      2. User will have to shorten the text
      3. Continues to (4f)

Use Case - Access/Edit Profile

* Overview:

The user comes to the website and wishes to edit something about their profile. They will click on the profile icon at the top of the page and then select their profile. The user can choose if they want to edit their image, password, and other necessary or optional pieces of information they want on their profile.

1. Description:

This use case will describe how the user will access and edit portions of their profile.

1. Actors:
   1. User
   2. System
2. Preconditions
   1. The user is connected to the Internet
   2. System is available
   3. User has an account and is logged in
3. Primary Flow:
   1. User arrives at Website Home Page
   2. User logs in or creates an account
   3. User will select profile icon
   4. Website displays options for the user to select
   5. User decides what they would like to edit
   6. User fills out information and saves
   7. System will confirm the save
   8. Terminate Use Case Access/Edit Profile
4. Alternate Flow:
   1. User is not logged in
      * If user selects their profile (4c) without being logged in
      1. The website will request the user to login or create an account
      2. User will log in or create an account
      3. Continues to (4e)
   2. Edit exceeds maximum length
      * If while filling information (4f) the text exceeds the maximum length
      1. Webpage will display that the comment is exceeding length limit
      2. User will have to shorten the text and save
      3. Continues to (4g)

Use Case - Search/Sort

* Overview:

User arrives at the homepage and wants to search for something or sort the feed that they are seeing on the page. They would then enter in the text to search or select an option to sort the feed that they are seeing to better fit what they want at that time. The system will then display the results or new sorted page.

1. Description:

This use case will describe how a user will use the search or sort feature of the website.

1. Actors:
   1. User
   2. System
2. Preconditions:
   1. The user is connected to the Internet
   2. System is available
   3. User has an account and is logged in
3. Primary Flow:
   1. User will arrive at the home page and logs in
   2. User will enter search criteria or choose a preexisting condition to sort by
   3. System will display the results or change to sort the users feed correctly
   4. Terminate Use Case Search/Sort
4. Alternate Flows:
   1. User is not logged in
      * If the user attempts to search or sort the page without being logged in (4b)
      1. The website will request the user to login or create an account
      2. User will log in or create an account
      3. Continues to (4b)

# Functional Requirements

**Non-Member User:**

1. **Create an Account**

**1.1 Description**: In order to create an account the user must enter their Username, Password, Email, and location. The system will not allow the user to create the account if the Username or Email were previously used with another account. In order to make sure the user used the correct password they will have to reenter the password a second time in a Re-enter Password box. Username, Password, Email, and Location need to all be filled.

**1.2 Sequence of events:**

1. User enters their custom Username
2. User enters their Password
3. User enters their Password again to confirm
4. User selects their location
5. User clicks “Submit” Button
6. System validates the password
7. System verifies Username is available
8. System contains a home button to redirect user after signup

**1.3 Function Requirement Label**

REQ 1.1 CreateAccount.

1. **About Us Page**

**2.1 Description**: The system will provide information about the website and the people who created the system. This can benefit the user by providing a reason for signing up and using the system. The user cannot make any changes to this page at all and can use the home button to return.

**2.2 Sequence of events:**

1. User clicks “About us” on the navigation bar
2. User is redirected to the About us page
3. User can read the information provided
4. User clicks the Home button to return to the home page

**2.3 Function Requirement Label**

REQ 2.1 AboutUs

**Member User:**

1. **Create Post**

**3.1 Description**: On the home page the user has the option to create a post. They will click a button labeled “Post” and select a post type which can be either a general post, event, or an offer or request for services. The user will type in a text box and submit the post to be added to their Home page.The system will not allow the user to post if the user is not a member. The system will prevent users from editing others' posts.

**3.2 Sequence of events:**

1. User clicks “Post” on the Home page
2. User selects which type of post they wish to make
3. User types in the text box their desired information
4. User clicks “Submit” button
5. System will confirm the post
6. The post is displayed onto the Home page

**3.3 Function Requirement Label**

REQ 3.1 CreatePost

1. **Comment**

**4.1 Description**: The user finds a post on the Home page that they desire to comment on. The user will click a reply button located on the post. The user will type what they desire into a text box under the post and click a submit button. The system will not allow the user to post if the user is not a member. The user can comment on their own posts. The system will prevent the user from editing others comments.

**4.2 Sequence of events:**

1. User clicks the “Reply” button under a user's post
2. User types their comment into the text box
3. User clicks the “submit” button
4. System confirms the comment
5. The comment appears to all users under the post

**4.3 Function Requirement Label**

REQ 4.1 Comment

1. **Private Message**

**5.1 Description**:The user may come across a post in which they’d like to private message the user who posted it. This feature is good for services and private information being disclosed. The user who wishes to initiate the private message will click the target's profile button then a text box will appear. The user will type in the box and then click submit. The user receiving the message will then receive the message on their own private message page. Users must be logged in in order to private message. Private messages can only be seen by the users involved. The system will prevent users from editing any private messages.

**5.2 Sequence of events:**

1. User clicks on another user's profile
2. User clicks on the “Private Message” button
3. A text box appears where the user can type their message
4. User clicks submit once their message is complete
5. System confirms the private message
6. The private message appears in both users' Private Message page
7. Comments can be made under the message in order to maintain conversation (See Comments)

**5.3 Function Requirement Label**

REQ 5.1 PrivateMessage

1. **Access/Edit Profile**

**6.1 Description**:User clicks on their own profile name or icon to be brought to their profile page. User then clicks a button labeled “Edit Profile”. Here they can change their image as well as insert their first name or change their password.

**6.2 Sequence of events - Change Password:**

1. User clicks on their profile name or icon on their Navbar
2. User clicks on the “Edit Profile” button
3. User clicks on their starred out password
4. User enters a new password in a textbox that appears
5. User confirms the new password again in another textbox
6. User clicks “Submit” button
7. System confirms the password change

**6.3 Sequence of events - Change Image:**

1. User clicks on their profile name or icon on their Navbar
2. User clicks on the “Edit Profile” button
3. User clicks on their current image
4. A popup will display allowing the user to upload their own image
5. User selects Image
6. User clicks “Submit” button
7. System confirms new image upload
8. Image is displayed for all users

**6.4 Sequence of events - Change/Add First name:**

1. User clicks on their profile name or icon on their Navbar
2. User clicks on the “Edit Profile” button
3. User clicks on the text box labeled “First Name:”
4. User enters in the text box what they desire
5. User clicks “Submit” button
6. System confirms the First Name
7. First Name is displayed on the user's profile

**6.5 Function Requirement Label**

REQ 6.1 ChangePassword

REQ 6.2 ChangeImage

REQ 6.3 ChangeFirstName

1. **Search/Sort**

**7.1 Description**: User desires to search on the home page or sort their home page feed. The user clicks on a text box labeled “Search” where they can specify what they are looking for on their feed. The user can also select different options to sort their feed.

**7.2 Sequence of events - Search:**

1. User clicks on the text box labeled “Search”
2. User types what they desire into the textbox
3. User hits enter on their keyboard
4. System confirms the entry
5. System updates feed according to the entry

**7.3 Sequence of events - Sort:**

1. User clicks “Sort” button near the “Search” text box
2. User clicks a predefined way of sorting the feed
3. System confirms the selection
4. System updates feed according to selection

**7.4 Function Requirement Label**

REQ 7.1 Search

REQ 7.2 Sort

# Non-Functional Requirements

**Availability:**

1. **Website Accessibility:** Website will be accessible by anyone, from any location with internet connection, daily. A free account will be required for personalized services, as well as for location-relevant information.
2. **Downtime:** Server should only be down during outages and during maintenance, which will happen quarterly at most. Downtime for a maintenance break will last for an hour at maximum.
3. **Support:** Email support will be provided for any account, and responses will generally take place in under 24 hours.

**Compatibility:**

1. **Resolution:** The web page will be optimized for any screen displaying a resolution between 1024 x 768 and 2560 x 1080.
2. **Browser:** The website will be optimized to run on the browsers Google Chrome, Safari, and Microsoft Edge, and remain functional on alternative browsers, such as Mozilla Firefox, Internet Explorer, and Opera.
3. **Device Compatibility:** The system will perform on any operating system or device; however, the priority of the system is to have a clean PC interface to appeal to the majority of users, and as such will not be as mobile (or tablet) friendly as it will be desktop/laptop friendly.

**Usability and Compliance:**

1. **Web Content Accessibility Guidelines (WCAG) 2.1:** The website will be WCAG compliant, serving to be accessible to a wider range of users, including those with hearing loss, low vision, limited movement, photosensitivity, as well as other disabilities.

**Functionality:**

1. **Bugs:** At launch, there will be no recorded bugs, and system errors will be encountered at a rate of no more than 1 bug found per month.
2. **User Error:** If a user attempts to input an incorrect value into the system, such as a non-existent zip code, or an already-taken username, they will be thrown an error message describing the mistake and be directed to re-input an acceptable value.
3. **Feedback:** Users of the system will be able to submit emails to the development team filtered under the context “Bug/Error” for immediate remediation, in order to minimize the negative impact of said bug.

**Maintainability:**

1. **Maintenance:** Maintenance which causes site downtime will happen quarterly at most. This will give time for the development team to update the service for any changing standards or requirements, as well as maintain a modern appeal and alleviate system errors.
2. **Lifespan:** The server hosting the website will remain online 24 hours a day, 7 days a week until termination.

**Performance:**

1. **Load Testing and Monitoring:** Website traffic monitoring will take place 24/7, and load testing will occur every quarter alongside the maintenance break. Load testing will test how much spike traffic the site can handle, as well as how much delay the system returns under stress, while being monitored using Loader software.
2. **Search:** The search feature will return results in under 400 milliseconds, before any network delay/latency issues.
3. **Page Loading:** Pages will not exceed 2 MB in size, to accommodate for slow internet connections.

**Portability:**

1. **Conflictions:** The system will not conflict with any other applications or web services running on any users machines.

**Reliability:**

1. **Uptime:** The website will remain live for an expected 99.96% of the year, every year, otherwise calculated as all but 4 hours of the year.

**Scalability:**

1. **Coding Standard:** The website will be W3C compliant, using the W3C Markup Validation tool provided by W3.org. The system will use up-to-date languages, namely HTML5, CSS4, JavaScript ES2015, PHP 8.0.10, and jQuery 3.6.0.
2. **Traffic:** As the system grows in both popularity and usage, it will require more storage and bandwidth for optimal performance. To start, 100 MB of disk space for web page hosting, 200 MB of database storage, and 500 MB of bandwidth per month. The disk space will be increased in increments of 100 MB, the database in 200 MB, and the bandwidth in 500 MB.

**Security:**

1. **Account:** Users wishing to utilize the full services of the system, including events and posts local to the user, must create a free account on the site. This will include creating a username, a password, providing a valid email address, and inputting your zip code.
2. **Encryption:** The system will use both a SHA512 encryption, along with salt, in the algorithm to store passwords and other sensitive information (such as zip codes and names). This will minimize the chances of a persons’ location being linked to their name.
3. **Bot Protection:** The system will enforce an anti-bot Captcha system upon account creation to ensure a server overload or exploit does not take place.

# 

# 

# System Architecture

1. **Lamp Server**: The fau.edu Lamp Server is where the web development project will be hosted throughout the Fall 2021 semester.
2. **Jira:** Jira is the platform that will be used to implement SCRUM methodology, assigning tasks for each team member for submission and review.
3. **GitHub:** GitHub is a platform that will be used to share the source code to facilitate access and control of the project for team members.
4. **Microsoft Visual Studio (IDE):** Microsoft Visual Studio will be the standard IDE used for code development, editing, and debugging. Microsoft Visual Studio includes all of the programming languages needed for this project, which are:
   1. **Hypertext Mark-up Language (HTML):** HTML will be the language used to structure the website and provide its contents.
   2. **Cascading Style Sheets (CSS):** CSS will be the language used to optimize the visual appearance of the website with consistent themes.
   3. **Javascript:** Javascript will be the language used to simulate dynamic behavior within the website as well as optimize the user interface.
   4. **Hypertext Preprocessor (PHP):** PHP will be the language used for the server-side functionality and database connection of the website.
   5. **JQuery:** JQuery will be the language used for the client-side functionality of the website.
5. **mySQL Database:** the mySQL Database will be utilized for storing and handling the web-based application’s data. Developers will manage the data inputted by users into the system.
6. **Discord:** Discord is the standard communication platform which will be used for project related communications and meetings between team members. Discord can also utilize a screen share function, which will be helpful for debating and reviewing code.
7. **Browser Compatibility:** The system will be a web-based application that properly functions for at least two major web browsers.

# Team

Scrum Master:

Kevin McInerney

Leader and Product Owner:

Thomas Gigliotti

Front End:

Thomas Moraes

Conor Coleman

Back End:

Jonathan Silva

Kevin McInerney

# Checklist

* Team decided on basic means of communications
  + Done
* Team found a time slot to meet outside of the class
  + Done
* Front and back end team leads chosen
  + Done
* Github master chosen
  + Done
* Team ready and able to use the chosen back and front-end frameworks
  + Done
* Skills of each team member defined and known to all
  + Done
* Team lead ensured that all team members read the final M1 and agree/understand it before submission
  + Done