**Milestone 1 Project Proposal**

**and**

**High-Level Description**

*CEN 4010 Principles of Software Engineering, Spring 2021*

**Group 8:** Connection During Social Distancing

**James Kos:** jkos2018@fau.edu,

**Robert Blanchette:** rblanchettej2017@fau.edu,

**John Marder:**  jmarder2018@fau.edu,

**Michael Norberto:** mnorberto2017@fau.edu,

**Kamal Shrouder:** kshrouder2018@fau.edu

**Documentation Date:** 11 February 2021

**Revision History Table:**

**-Version 1:** Originally Created 2/11/21 at 1:46PM

1. **Executive Summary:**

During the pandemic, many people are especially isolated due to lockdown measures and social distancing. Many people lack connection with others, and many with a lack of understanding of technology are left with little to no communication methods. These issues can be especially damaging to the mental and physical health of many people. The point of PostHut is to help rectify this situation and to foster connections between people. Current social media applications seem to remove a large part of the human element with almost warring factions of users that instead of fostering connections only foster hate. They also fall short in terms of allowing communication for all, as they are often too complex for new users to effectively use. This in conjunction with the alienation of older users has meant that many social media platforms fall short of allowing for connections or even effective communication during the pandemic. PostHut seeks to solve this issue by putting emphasis on simplicity in the design to allow for users of all ages and conditions to connect with others with ease. We additionally plan on finding and nurturing connections between people with daily themes that users can post to. We believe that this would allow for more common ground between people, while also providing wholesome content over which to bond. These content themes, in addition to the simplicity and emphasis on connection and common ground would not only help promote interactivity, but would again help those currently lacking connections find them. We additionally plan on having the ability for users to see the newest and popular posts and get an idea of what is happening on the platform and in their community. All of these features and the discussion that goes along with them are likely to promote connections in ways that other social media platforms currently cannot do.

1. **Competitive Analysis:**

Analyzing competitive products available today. Present competitors’ features vs. your planned ones. First, create a table with key features of competitors vs. yours. Only at a very high level, 5-6 entries max. After the table, you must summarize what are the planned advantages or competitive relationship to what is already available.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | PostHut | Facebook | Twitter | Reddit |
| No friends/followers  implemented | ✔ | ✘ | ✘ | ✘ |
| Daily themed social media posts | ✔ | ✘ | ✘ | ✘ |
| Information about what is going on with the community | ✔ | ✔ | ✔ | ✔ |
| An option to see the most popular post/discussions | ✔ | ✘ | ✘ | ✔ |
| Discussions of ideas to make individuals connected | ✔ | ✘ | ✔ | ✔ |

According to the analysis, PostHut includes many key features that many mainstream social media platforms such as Facebook, Twitter, and Reddit have yet to offer. For instance, having a daily themed social media post. Each day, a new theme will be released and customers can contribute by sharing their pictures and discussing them further. This feature is very unique and can make our application achieve a growing community. Additionally, these themed posts would help promote interactivity and connection in the community by helping users find common interests. PostHut also contains a character count limit to five-hundred. This character limit is significantly greater than Twitter, so users can have longer posts and discussions. A friend/follower is used in many social media platforms that show specifically what only they post or share. PostHut also doesn’t have a friend or follower like many other social media platforms. This is because all members of PostHut will be treated as our friends, such as a community, so we all can deal with these convoluted times together.

1. **Data Definition**

This section serves as the “dictionary” of your document. It defines main terms, data structures and “items” or “*entities*” *at high or logical (not implementation) level* (e.g. name, meaning, usage, and NOT how the data is stored in memory) so it is easier to refer to them in the document. Focus on key terms (main data elements, actors, types of users etc.) specific for your application and not on general well known terms. These terms and their names *must be used consistently* from then on in all documents, user interface, in naming software components and database elements etc. In later milestones, you will add more implementation details for each item. You will later expand this section with more details.

User - A person who has a PostHut account with which they view and post content

UserPage - A data object that each user has containing information about them

PostHut - Social Media app that allows users to post images and text

Post - An object that is timestamped containing text or images which be voted on by other users

“Nice!” - Our app’s way of voting in favor of a post similar to ‘Like”, “Up-Vote” or “Heart”

userCommunity - A data object that contains all users of the PostHut app it will be accessible to the User as “PostHut Community” via the GUI displaying their UserPage

myCommunity - A data object that contains a specific user’s community based on criteria like town, county, or state and it will be accessible to the User as “My Community” via the GUI displaying their UserPage. (Possibly have a feature to connect to individual users.)

postQueue - A data object that contains all of the posts created by users of PostHut and it will be accessible to a User as “Go to Community Square” via the GUI displaying their UserPage.

popularPostQueue - A data object that contains posts that have received the most “Nice!” votes by the users and it will be accessible to a User as “What’s Really Connecting with People” via the GUI displaying their UserPage.

newPostQueue - A data object that contains the posts from the past 30 minutes and it will be accessible to a User as “What’s the Latest” via the GUI displaying their UserPage.

1. **Overview, Scenarios and Use Cases**

This section describes the project overview (in much more details) and likelihood usage scenarios of your product from end users’ perspectives. Focus only on main use cases. Simple text format is OK and preferable – tell us a story about who and how is the application used. Focus on WHAT users do, their skill level, not on HOW the system is implemented. You can expand use cases provided in high level document in future milestones.

Create Account

1. User navigates to PostHut webpage
2. User selects “Create New Account” option from sign-in page
3. User creates a username
4. User creates a password
5. User provides name, location, and email address
6. User clicks on “Sign-up” to complete process

Login

1. User navigates to PostHut webpage
2. User enters username and password then clicks “Sign-in” from sign-in page
3. PostHut directs the user to their UserPage

Make a Post

1. User has completed Login
2. User selects “Create Post” option from the GUI displaying their UserPage
3. User enters text up to 500 characters and/or image file
4. User selects “Share” to display their Post with the userCommunity

View All Posts

1. User has completed Login
2. User selects “Go to Community Square” option from the GUI displaying their UserPage
3. PostHut displays the postQueue to the GUI
4. User is able to scroll or return to UserPage

View Popular Posts

1. User has completed Login
2. User selects “What’s Really Connecting with People” option from the GUI displaying their UserPage
3. PostHut displays the popularPostQueue to the GUI
4. User is able to scroll or return to UserPage

View New Posts

1. User has completed Login
2. User selects “What’s the Latest” option from the GUI displaying their UserPage
3. PostHut displays the newPostQueue to the GUI
4. User is able to scroll or return to UserPage

Vote on a Post

1. User has completed Login
2. User has selected one of the options to view one category of Posts
3. User is viewing a Post
4. User clicks on the “Nice!” icon
5. PostHut adds User vote to the Post vote count
6. **Initial List Of High-level Functional Requirements**

This refers to the high-level functionality that you plan to develop to the best of your knowledge at this point. Focus on WHAT and not HOW. Keep the users in mind. Develop these functions to be consistent with use cases and requirements above. Number each requirement and use these numbers consistently from now on. For each functionality use 1-5 line description.

1. When a user selects “Create Post”, an empty text box will appear where the user can type characters in. Once the post is ready to be sent to the discussion board, the user will select “Share”. The post will now be on the postQueue with zero “Nice!” by default.
2. When a user is creating a new post, there will be a character limit of up to five hundred characters. Once a user gets over the limit, a warning will appear stating that they have acquired over the character limit and they will be unable to post the text. Once the post is within the character limit, the user is able to post.
3. The postQueue section will display posts and options underneath where a user can “Nice!” a photo. If the post is brand new, it will by default have zero “Nice!”. A user can Nice! each post only once. Once a user “Nice!’s” a post, the total number of “Nice!” will increment by one.
4. The “myCommunity” section will initially ask the user what town, state, or country they are from. Once the user fills the information in, PostHut will display posts that are in the vicinity of that location.
5. If a user selects “userCommunity”, it will show the accounts of all users of PostHut. If one selects one of the other users, it will display their name, which myCommunity they are in, and their previous posts.
6. On a discussion board, if the user selects “Go to Community Square”, then the postQueue will sort randomly.
7. On a discussion board, if the user selects “What’s Really Connecting with People”, then the postQueue will sort in order by who has the most “Nice!’ votes.
8. On a discussion board, if the user selects “What’s the Latest”, then the postQueue will sort in order by who has the newest posts.

**6. List Of Non-functional Requirements**

For example, performance, usability, accessibility, expected load, security requirements, storage, availability, fault tolerance etc. Number each. When possible, try to quantify these quality attributes.

1. Response Time: Users with a decent internet connection should not take longer than 2 seconds to load a page or 5 seconds to submit a new post. The computational complexity of sorting algorithms for posts should not exceed O(nlogn).
2. Usability: Since our application is honing on the prospects of being simple to manage and use, the usability of PostHut will seem as simple as hitting simple inputs.
3. Accessibility: The PostHut application will be available to access on all mobile devices as well as desktops and laptops. Mobile device accessibility is available for but not limited to Apple, Samsung, Google, and LG. Because this application is done through interacting with others, a stable internet connection is going to be needed to get as much out of the app as possible. No internet connection is unfortunately something we can’t work around.
4. Expected Load: fewer than 100 users.
5. Security Requirements: Passwords should be at least eight characters long and encrypted and stored somewhere inaccessible to users. All inputs should be checked to avoid SQL injection.
6. Storage: Passwords are encrypted and stored in a secret location. Posts, user information, and everything else should be stored in an SQL server.
7. Availability: PostHut should be available through browser access everywhere where an internet connection is available. It may also be accessed on mobile devices with internet connections through the app. Updates to the interface should not be noticeable as the browser could update while switching pages and the mobile app should be updated automatically by the device’s store app while not in use. Other updates should take place at midnight EST and not take longer than 5 seconds. They should also not cause any post currently being typed to be reset or unable to post; if an update requires such to happen, it must be announced at least three days before with a scheduled downtime period unless it is a critical security update; in that case, a notification with an apology must be displayed to users the next time they log in if it is within three days of the incident.
8. Fault Tolerance: No more than one post per 1,000 should fail to send due to errors on our end.

**7. High-level System Architecture**

Lists of main software products, tools, languages and systems to be used, list of core APIs available at this point, supported browsers etc.

You also have to decide on which frameworks you will use if any. These provide both user interface, as well as cross-platform and cross browser layout/css. All external code you plan to use must be listed along with their license.

***Tools:***

* **Brackets:** Programming and designing the browser experience/website
* **Flutter:** Programming and designing the mobile experience
* **Adobe XD:** GUI Mockups
* **Atom/Visual Studio:** Programming the backend
* **WinSCP:** Uploading the website files to the FAU Lamp Server
* **MySQL:** Maintaining and storing user data

***Languages:***

* **HTML:** Creating the basic framework of the website
* **CSS/SASS:** Creating a lot of the aesthetic of the website
* **PHP:** Creating extended functionality for the website (including forms and submissions)
* **Javascript:** Extending the functionality of the website including getting APIs and more
* **Dart:** Programming the app in Flutter
* **SQL:** Database for storing user data

***Core APIs/Frameworks:***

* **Bootstrap:** For modernizing the look and feel of the website
* **Flutter:** Also a framework and API for creating applications with complexity

***Supported Browsers/Operating Systems:***

* **Firefox**
* **IceWeasel**
* **Chrome**
* **Opera**
* **Microsoft Edge**
* **Android**
* **IOS**

**8. Team**

List student group names, name of Scrum master, product owner and initial roles for each member

**-Robert Blanchette:** Development Team

**-James Kos:** Product Owner, Team Lead, Backend Lead

**-John Marder:** Scrum Master, Backend Dev.

**-Michael Norberto:** Development Team, Github Master, Front End Lead

**-Kamal Shrouder:** Development Team

**9. Checklist**

For each item below you must answer with only one of the following: DONE, ON TRACK (meaning it will be done on time, and no issues perceived) or ISSUE (you have some problems, and then define what is the problem with 1-3 lines). Reflect these items in your Jira project space:

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

**10. Tasks Before Submission**

Teams must collaborate in creating M1 document by having working M1 document on their team GitHub repository (similar to managing code) so all team members can access it. Added advantage of doing it this way is that it builds teamwork and communication. We recommend having a folder for project documentation on team’s GitHub where milestones and other similar files can be kept.