**CEN 4010 Principles of Software Engineering**

**Fall 2019**

**Team 4**

**Milestone 3 Project Proposal and High-Level Description**

**The Freelancers**

**Campus Snapshots**

**Members:**

**Aaron Haim – ahaim2016@fau.edu**

**Daniel Fradlin – dfradlin2017@fau.edu**

**Richard Urena – urenar2016@fau.edu**

**Yiyang Liu – yliu2015@fau.edu**

**Revision 2: November 3, 2019**

**Revision 1: September 28, 2019**

**Created September 23, 2019**

**Executive Summary:**

*Campus Snapshots* is a web system that people can access and inform users about issues and events around campus. It provides real time snapshots via a map that shows the campus status. If an issue arises, it can allow administration to provide help as needed to those markers located on the map. It can also be used to report events happening on campus such as games, concerts, activities, etc. *Campus Snapshots* is made with the intention of allowing campuses to more easily communicate items in an instantaneous fashion throughout the area. It can be used as an itemized list of problems to solve, a way to better organize what is happening, or as a way to send out notifications about specific problems that cannot be solved at that current moment. It is, in a sense, a one stop shop to know any major developments within a campus.

**Competitive Analysis:**

|  |  |  |
| --- | --- | --- |
| **WAZE app** | **FAU Mobile App - Maps** | **Campus Snapshots** |
| Traffic update | Search campus buildings | Report campus issues and events |
| Speed Traps | View popular locations | Show status report of instance |
| Accidents |  | Information about all current events/issues |
|  |  | Search campus buildings |
|  |  |  |

There are some great advantages to implementing such a system. One major advantage is that the student to administrator ratio is about 22:1. With more eyes, more issues can be reported quickly. It allows for the administration to have extra eyes and stay up to date on the markers. Also, with the ability to upload photos of the reported issues, it will allow for issues to be validated and for there to be no false reporting.

Another great part of this system is that if students decide to host their own events, they can add that event to the site for other students to join if they are interested. FAU doesn't have a club for everything so this system makes it easier for students to find others that share the same hobbies/interests.

As for the information in the table, there really is not a major competitor in this field since most campuses handle this information on their own. The closest campus app has to be the FAU Mobile App while the closest global app is the WAZE app. Neither of these have any sort of reporting system implemented or have any major way to update user sightings automatically. Our web system allows students to report issues and events to a database instantaneously and updates on the map. You can check on administration marked statuses to view the progress of how the issue is being handled or what event is currently going on. You can also read the information provided by administration and/or students to see what the issue or event going on is. Finally, there is a way to search the campus buildings as the FAU Mobile App does but with the power to see what is specifically happening in that building. This is not a standardized industry allowing for even basic implementation to have massive benefits with the use of students participating in spreading the word.

**Data Definition:**

* *Campus Snapshots*: Name of the product
* Students: An access point attending a campus that can create markers and upload photos with them
* Administration: An access point working at a campus that has the same roles as Students along with the ability to mark status updates on the marker and/or delete them
* Users: Refers to both Students and Administration as a whole
* Campus: The location currently being monitored by the web system
* Web System: A storage database map that holds all markers on a certain campus
* Snapshot: An instance in time where a marker has occurred and has been updated to the web system
* Marker: An issue or event created by the users
  + Location will refer to the closest building to the marker
  + Image upload can be in PNG/JPG format and at least 480p
* User Privileges
  + Students
    - Create markers
    - Upload photos attached to markers
    - Add a comment on marker
    - Edit created markers/comments
    - Report an issue on specific marker
  + Administration
    - All Student privileges
    - Update status on markers
    - Delete markers
* Registration Information
  + Users
    - Valid Campus Email Address
    - Username (6+ characters long)
    - Password (8+ characters long)
* Main Information
  + Raw Data
    - Information entered by users
  + Metadata
    - File format and size of images (PNG/JPG; 480p+)
    - Marker Data
      * Event ID (11-bit integer)
      * Event Name (20 characters)
      * Event Location (25 characters)
      * Event Time (8 characters; will be switched to date format)
      * Event Description (255 characters)
  + Supporting Data
    - User Login Information (email/username/password)
    - Location listing

**Overview, Scenarios, and Use Cases:**

Users create an account. Students log in and can view current events or issues happening around campus. Students may create a new marker and report an issue. Administration then sees the new marker and provides status update on that issue based on their progress. If there is a duplicate reporting then Administration can remove the duplicate report. Students can also create a new marker for an event they are hosting. Students can report any inappropriate posts and provide a reason. Administration has privileges to remove any inappropriate posts if policies are violated.

**High-Level Functional Requirements:**

1. Users should be able to create an account if they do not already have one.
   1. A valid email address is required to create an account.
   2. A username must be at least 6 characters long.
   3. A password must be at least 8 characters long.
2. Users will be identified by their campus provided email.
   1. Any duplicate emails shall not be permitted to create an account.
3. Users should be able to attach a photo to their reporting and creation of marker.
   1. The picture must be a PNG or JPG file format, all other types will not be taken.
   2. The picture must be at least 480p so the image can have decent quality.
4. Users should be able to comment on existing markers.
   1. Comments will be limited to 100 characters.
   2. Students can only post one comment per marker and zero on markers created by that student.
5. Users should be able to report inappropriate posts.
   1. Students that report inappropriate posts must be reviewed by Administration.
6. Administration should be able to delete/modify markers and comments.
   1. Any deleted markers/comments will be completely removed from the web system.
7. Administration should be able to provide status updates of reported issues.
   1. Status updates will either be in the form of comments or its own issue resolver system.
   2. If in the form of comments, previous status updates will be deleted from the system.

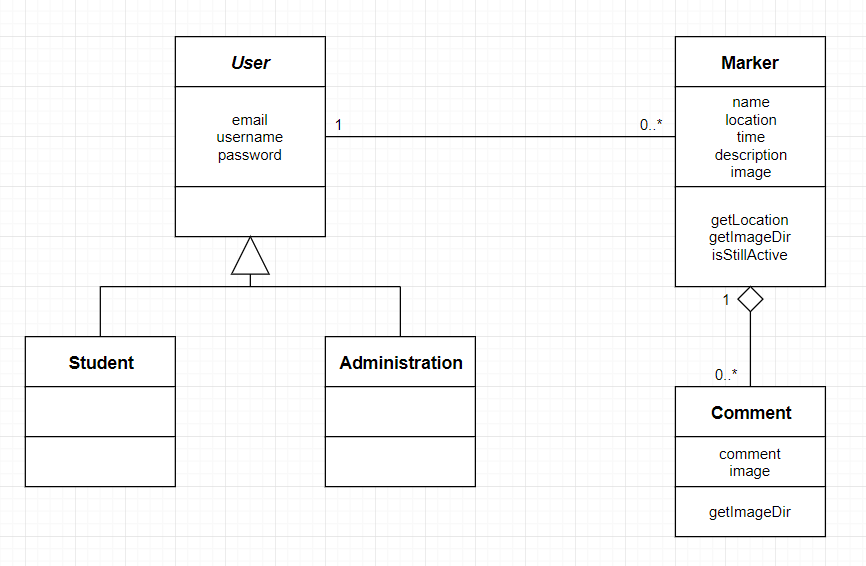
**Non-Functional Requirements:**

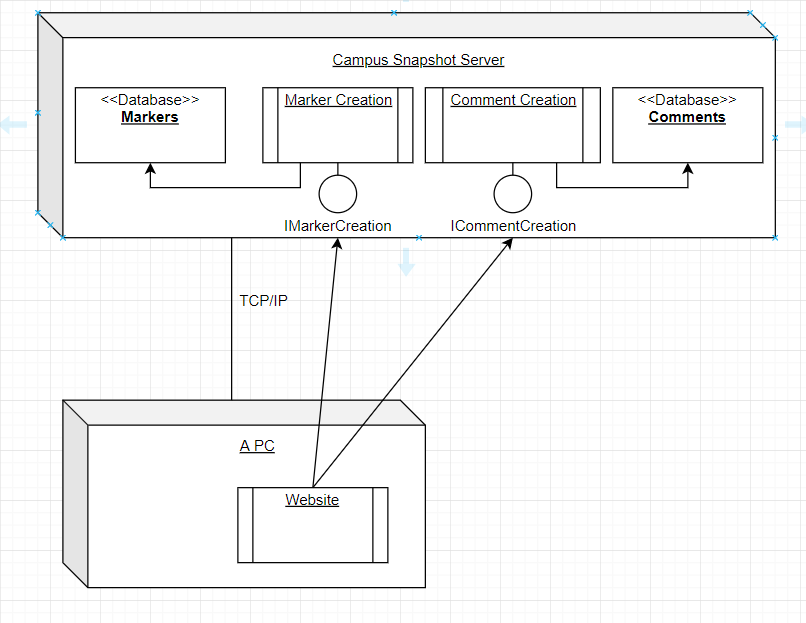
1. Students should be limited to a specific number of posts per day to avoid spamming the system.
2. Users can only have a single account.
3. Posts should indicate whether it has been modified by Student or Administration.
4. Users should be able to use the system without instructions.
5. JavaScript, SQL, HTML, and CSS will be the main focus languages used to develop *Campus Snapshots*.
6. Markers can be sorted/displayed by location on campus.

**High-Level System Architecture and Database Organization:**

* Languages
  + JavaScript
  + HTML/CSS
  + SQL
* Tools
  + GitHub
  + Google Docs
  + Trello
  + Discord
* APIs
  + Google Maps API (Dependent on best functionality or just campus map)
  + Custom APIs
    - Marker Creation
      * Creates markers based on User email
    - Comment Creation
      * Creates comments based on User email and Marker name
* Browsers
  + Google Chrome
  + Mozilla Firefox
* Database Organization
  + Users
    - Email Address (Unique)
    - Username (Unique)
    - Password
  + Markers
    - Email Address (Users)
    - Marker Name
    - Marker Location
    - Marker Time (Can be null)
    - Marker Description
    - Image
  + Comments
    - Marker Name (Markers)
    - Email Address (Users; independent of Markers email address)
    - Comment
    - Image
  + Filtering/Searching
    - Simple Comparison search
    - Students can only filter markers by location
    - Administration can filter markers/comments by location or email address
* Images
  + PNG or JPG format
  + At least 480p
  + File System with DB reference format
* Prioritization
  + Most recent events will be prioritized first in the marker display

**High-Level UML Diagrams**





**Key Risks To Project**

* **Skill Risks**
  + None at the current moment
* **Schedule Risks**
  + **Potential schedule risks**
    - There is a small potential that some features might not make it into the final product
    - Due to current workflow speeds, however, this issue is very minute and probably will not affect the final product release
* **Technical Risks**
  + **Implementation of Visual Marker Location**
    - See Google Maps API under Legal/Content Risks
    - Best workaround is to use static image of campus with dynamic markers
* **Teamwork Risks**
  + No teamwork issues
* **Legal/Content Risks**
  + **Google Maps API**
    - **User agreements and Costs**
      * Google Maps API is free to use but does have some user agreements and costs depending on what is used
      * The easiest way around this issue is to just use an image instead of the API, depends on the path taken once we get to that stage of front-end programming

**Team:**

* **Aaron Haim**
  + **Scrum and GitHub Master**
  + **Back End Programmer**
* **Daniel Fradlin**
  + **Front End Lead Programmer**
* **Richard Urena**
  + **Back End Lead Programmer**
* **Yiyang Liu**
  + **Product Owner**
  + **Front End Programmer**

**Checklist:**

* **Basic Means of Communications - DONE**
* **Time Slot to Meet - DONE**
* **Front and Back End Leads Chosen - DONE**
* **GitHub Master Chosen - DONE**
* **Team Ready/Able to Use Chosen Frameworks - DONE**
* **Member’s Skills Defined and Known - DONE**
* **Team Lead Ensures Members Agree/Understand M1 - DONE**