1. Title Page

CampSnap

Campsnap is a modern web application.

**Group 7**

**Team #**

Daniel Canter (dcanter2015@fau.edu)

Timothy Giles (tgiles1@fau.edu)

Giselle Lubin (grezende2013@fau.edu)

Omar Oropeza (ooropeza2014@fau.edu)

|  |  |
| --- | --- |
| **History Table** | |
| Date: | Description: |
| July 23rd | Milestone 3 submission |

July, Monday the 23rd

CEN 4010 Principles of Software Engineering Summer 2018

1. **Executive Summary**

The final project our team will be building will be a program to help others report problems they may see on campus. The product name will be CampSnap and it will be primarily targeted at the faculty and students of Florida Atlantic University. It will be an easy to use web application that will make it easier than ever before to report a problem on campus. See a broken sidewalk, water fountain that doesn’t work, or maybe even a lightbulb that’s out and report the problem on the spot on the website. Just take a picture of the issue, give a description and submit it. Your report will be go through three phases once it’s submitted with the phases being

1. Reported

2. Underway

3. Resolved.

If our staff finds that your report wasn’t a necessary one, or that there are multiple reports about the same issue, we may remove your report to help with the avoidance of duplicate claims and resolve the issue faster. CampSnap will be monitored by FAU staff so you can put your reports right in the eyes of the ones you want to see it the most. Enjoy CampSnap on its expected launch in August 2018!

1. **Competitive Analysis**

We believe our major competitors are Facebook groups/subreddits. Right now it is entirely possible to make a Facebook group named “FAU Problems Group” and have all the discussion on the page moderated to focus on fixing problems around campus like our app is based on. However, we believe a product that can be tailored to fit a solution will never be as good as a product that was made from the start to solve that problem.

Core Features

|  |
| --- |
| Any student can report a problem, not reserved for faculty only. |
| A clear visual of where their report is status wise. |
| Easy to use interface. |
| Picture upload. |
| Well optimized application. |

We believe with our layout and focus on doing one thing very well, we will be able to beat our main competition.

1. **Data Definition**

**User:**

Admin=User that has privileges to update the status of reports and able to remove posts that don’t fit our guidelines.

**Report:**

Metadata=Title of report, location of report, location\_id(each part of campus will have an id associated to it)

Description=Brief description of the problem that is being reported.

**Photo:**

Formats=.png, .jpg, .gif, and .gifv all accepted.

Max Size=2mb per upload.

**Location:**

id=As explained earlier, every area on campus will have an id associated with it. This makes it easier to map out DB wise.

1. **Overview Scenarios**

We believe our users will use our application for its intended purpose, to report issues around campus. There is not much other functionality built into the platform as it serves one purpose. Users could try to abuse it to host photos as their will be photo functionality but if this is detected such images/posts will be deleted by our staff immediately. There is also a max size on content that can be uploaded, which as of right now is 2mb per upload so this will also be a deterrent for that. As the intended market is students and faculty we believe their skill level will be perfectly suitable to use our application and we will try and deliver the best possible user interface and UX design possible to achieve this goal.

1. **High-Level Functional Requirements**
2. Photo functionality**(Requirement-Spec: 1**)- We want users to be able to upload a photo, preferably showing the exact problem at hand. Our guidelines will state that photos should be clear and try and show as much of the issue as possible as well as list in the description where on campus the issue is located.
3. User Login**(Requirement-Spec: 1)**- To prevent anonymous users reporting problems, we will require you to be logged in to view and report problems on campus. Only @fau.edu email addresses will be accepted.
4. Admin Privileges**(Requirement-Spec: 1)**- Admins (developers and possibly faculty) will be able to remove posts that don’t fit our criteria or that are duplicates that are already being worked on.
5. Video functionality**(Requirement-Spec: 3)-** We will try and enable video uploads for reports as well but this isn’t a very core part of our philosophy. We believe a photo and a description will suffice for 90% of problems but if our users show us that they want video upload, we will work hard to add the feature.
6. **Non-Functional Requirements**
7. We believe the performance will be stellar as there really isn’t a lot of complex data structures involved in an application like this.
8. Expected load for our application is expected to be fairly low as well as it should really only be used when there is an issue on campus. It is not meant to be a daily opener like a social media app for example.
9. Accessibility: Our application will be as accessible as the users availability of internet or cell service access. It should always be up and running and ready to be used.
10. We will monitor the application daily for faults or any other related issues and try and resolve them as quickly as possible.
11. Storage space for photo uploads shouldn’t be an issue either, as there is a limit on the max file size that can be uploaded and our available storage size is plentiful.
12. **High-level System Architecture**

Frameworks

-Frontend: Bootstrap

Languages

-Javascript

-PHP

Supported Browsers

-Chrome

-Safari

-Firefox

1. High Level Architecture: Code structured in a classic OOP manner. Each time a new post/user/database table is about to be created, it will be created from an object of the corresponding class. I.e. before a user is saved into the database, a new User object is created and the data is pulled from this object and then stored in the DB.

2. DB Organization: The same as from **4. Data Definition-**

User

* id
* first\_name
* last\_name
* username
* password
* admin (Boolean)
* created\_timestamp

Report

* id
* user\_id
* comments\_id
* location\_id
* title
* description
* status
* created\_timestamp

Photo

* id
* report\_id
* photo\_path
* size

Location

* id
* name

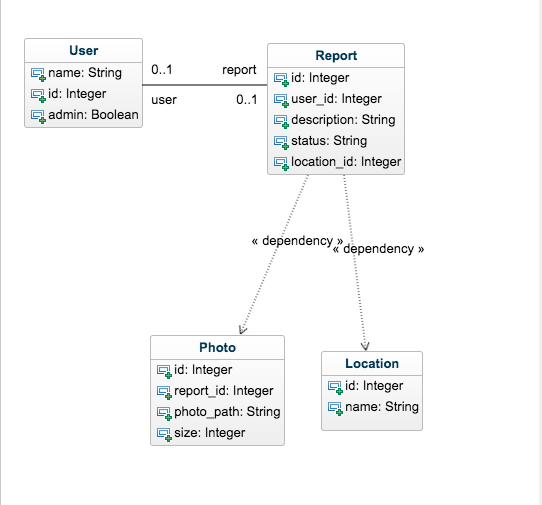
3. Media Storage: Images/video will be stored in the file system and the path of where it is placed will be stored in the DB for retrieval.

4. Search/filter: The main page will have a list of all the current reports by all of our users. I will be filtered by time submitted and that is the only criteria. Search functionality to find a specific report will be filtered by title of the post. So if there was a report with the title “Breezeway lightbulb out”, and a user typed into the search bar “breezeway”, that post would show up (as well as any other reports with that keyword in it).

5. We will not be using any proprietary APIs in our application.

6. The only non-trivial process in our application is the status feature, where we show the users what status a report currently is in. It will be updated immediately once an admin has changed the status of the report.

1. **High-Level UML Diagram**



1. **Key Risks**

1.Key Risks: We believe we have the right skills to get the job done. We all have a grasp on what type of database structure and front end design needs to happen to bring the product to life. None of our team members have any sysadmin experience but as we aren’t actually operating on the servers just essentially loading our files into it, this risk is moot.

2. Schedule Risks: We believe we have the resources and time to finish the product with what we’ve committed.

3. Technical Risks: We believe for the load we predict our application to receive we should have the hardware available to keep the site running. As long as there isn’t any Denial of Service attacks targeted at our platform (unlikely) we believe their shouldn’t be any technical risks.

4. Teamwork Risks: There have been a couple of times where teamwork and communication could be a little better but when crunch time comes, our team is fairly compliant and hard working.

5. Legal/content risks: All the content we will provide is provided for by our users. If any of our users upload content that does not correlate with a problem/issue at FAU then it will be removed immediately. CampSnap does not use any copyrighted code/content/etc.

We plan to resolve any unforeseen risks with a level head and proper reasoning. We will first identify the risk and work as a group to find the best solution given our current resources. We will try our best to not let any risks affect the well-being of our product as well.

|  |  |
| --- | --- |
| **Team** | |
| Name | Role |
| Danny Canter | Frontend Designer |
| Giselle Lubin | Backend Designer |
| Omar Oropeza | Scrum master |
| Timothy Giles | Product Owner |

**Checklist**

* Team decided on means of communication - DONE
* Team found a time slot to meet outside of the class - DONE
* Front and Back End Teams 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 agreed before submissions - DONE