**3. Competitive Analysis**

After deciding about the project’s theme and its main features, we analyzed other competitive products available today on the market. Following is a table comparing the key features of competitors versus our product. In comparing the products we focused mainly on the high-level functionalities.

|  |  |
| --- | --- |
| Report instantaneously issues | **Infinite Campus, Inc**  User reviews/feedback |
| Large Database | **Campus Explorer, Inc**  User friendly website |
| Search for School Activities | **Data Snapshot**  Easy issue upload/report |
| 24/7 Service/emergency | **Infinite Campus**  More variety of report tools |
| Directions on safety procedures | **Campus Life**  Data on previous reports |

Campus Snapshots vs  Competitors

There are many online school systems, where university campuses around the country can keep up to date and track on environment issues and school activities. However, some competitors do not offer a 24/7 events/issues happening in real time. Competitor’s products are mainly a news page, rather than a tool for detecting and solving problems on campus. Campus Snapshots will offer a better quality of service for students to feel safe, a good living and good study environment keeping the campus clean and safe.

**4. Data Definition**

*This should be reasonably consistent with Milestone 1 but should be expanded as needed and refined as per feedback. Major data items that comprise of sub-data items have to be defined in full (list all its sub-data items, and for images/video list formats, max size etc.). You must use all the data definitions and names consistently in all documents, including GUI text. Focus on data items unique and important to your application and avoid explaining obvious things like Internet,, Browser, Cloud, etc. Be sure to cover ALL items critical to your project and especially those providing a competitive advantage. At this stage data describing user privileges, registration info and main info (raw data, metadata, supporting data) have to be fully defined (as much as it is possible at this stage)*

1. Campus Snapshots — A dynamic web system that publishes information about on campus events and problems, in addition to allowing students to contribute by submitting information about issues happening in real time.
2. Account – user + admin
3. User – A person (student, faculty, staff) that utilizes the web system to make posts about events and/or problems.
4. Admin – A team consisting of members from the university’s facility management, responsible to monitor the activity Campus Snapshots, and solve the problems that are reported.
5. User ID
6. Password
7. Problem — Any event that can disrepute students’ everyday life (*Ex: A building build closed or school transportation system not working)*
8. Event
9. Database — A collection of information that is organized so that it can be easily accessed, managed and updated.
10. Photo – format, size
11. Video –

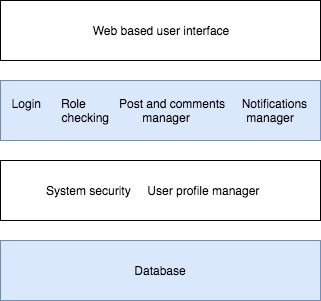
**6. High-level functional requirements**

|  |  |
| --- | --- |
| Priority Level | Functional requirement |
|  | 1. The application shall offer information about events and problems |
| 1  1  1  1  1  1  1  3  1  2  2 | * 1. The general user shall be able to create a new account   2. The general user shall be able to change his/her password   3. The general user shall be able to reset a forgotten password   4. The general user shall be able to login to his/her account   5. The general user shall be able to logout from his/her account   6. The user shall be able to create a new post   7. The user shall be able to upload photos to their post   8. The user shall be able to upload a video to their post   9. The user shall be able to read and add comments to a post   10. The user shall be able to search for an existing post   11. The user shall be able to view the latest posts on his main page |
|  | 1. The application shall offer a notification system |
| 1  2  3  3  3 | 2.1 The admin shall receive an in-app alert and an email when a new problem is reported  2.2 The user shall receive an in-app alert when the status of a reported problem changes  2.3 The user shall be able to opt for receiving notifications via SMS and/or email  2.4 The user shall be able to choose what kind of notifications he/she wants to receive (reported problem updates / upcoming events)  2.5 Calendar |
|  | 1. The admin shall be able to manage the reported problems |
| 1  1 | 3.1 Each reported problem should have a status report (reported, in process, solved)  3.2 The admin shall be able to change the status of a problem |

**8. High-level system architecture and database organization**

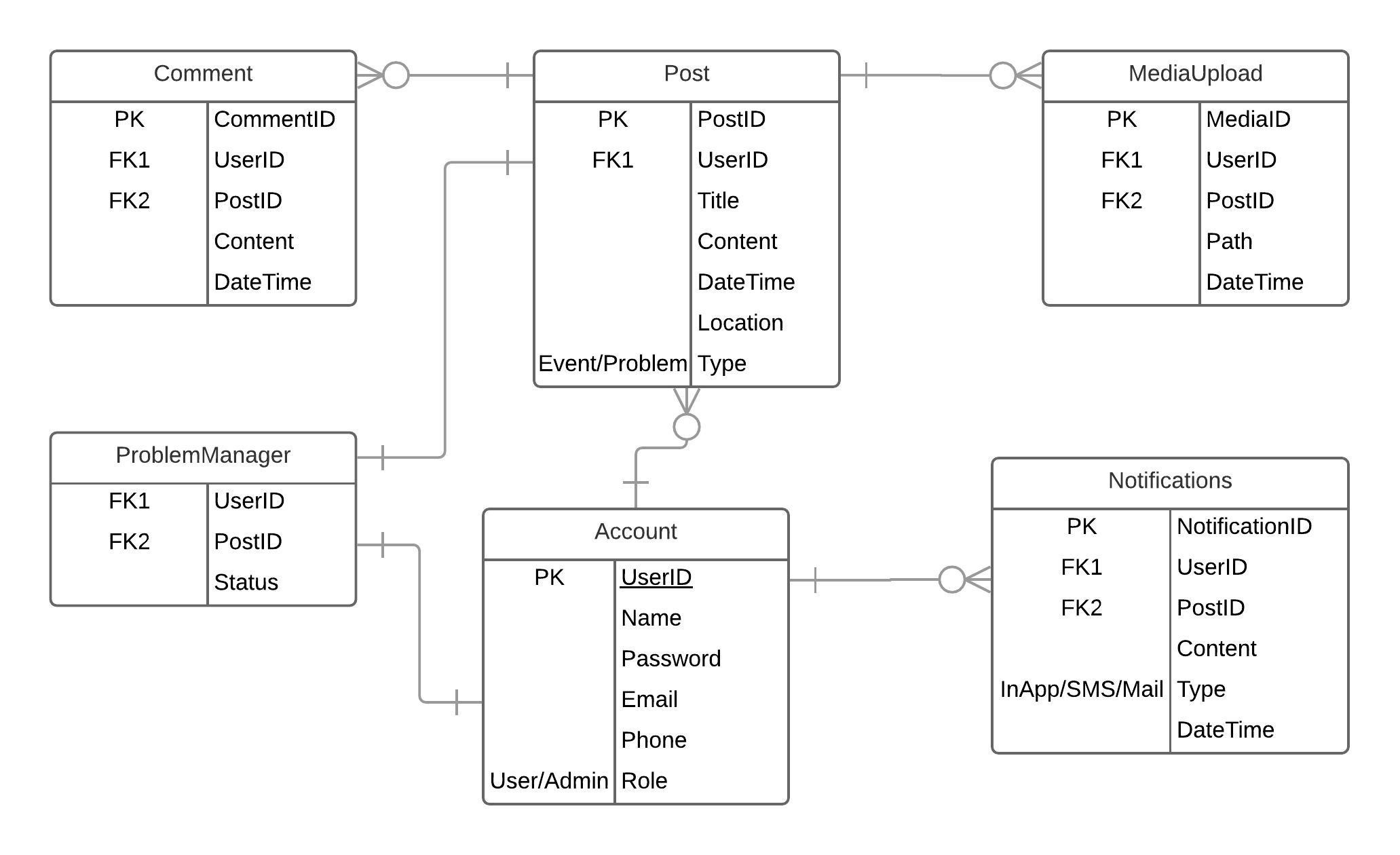
1. High-level Architecture

The high-level architecture of Campus Snapshots is modeled using a layered approach. The top layer is a browser-based user interface. The second layer provides the user interface functionality that is delivered through the web browser. It includes components to allow users to log in to the system, and checking components that ensure that the operations they use are allowed by their role. This layer also includes a post and comments manager that allows users to create a post, upload media, add and read comments, and change the status of a problem. The last component at this layer is a notification manager. The bottom layer is the system database.



1. DB organization

Following is an entity-relationship diagram showing the main entities and their attributes that will be included in the database of Campus Snapshots.



1. Media storage
2. Search /filter architecture and implementation
3. Your own APIs

<config>

<api>

<resources>

<user translate=“title” module=“user”>

<title>User Resource</tittle>

<methods>

<list translate="title" module="user"> <title>Retrieve User</title> </list>

<create translate="title" module="user">

<title>Create User</title> </create>

<info translate="title" module="user">

<title>Retrieve User data</title>

</info>

<update translate="title" module="user">

<title>Update User data</title> </update>

<delete>

<title>Delete User</title>

</delete>

</methods>

<faults module="user"> </faults>

</customer>

</resources>

</api>

1. Describe any significant non-trivial algorithm or process (rating, ranking)

**9. High-level UML diagram**

class diagram

component and deployment diagram

**10. Key risks for project**

1) Skill risks, code implementation and workload completeness are not too much of a issue at the moment all members working to the best of their abilities.

2) We were having some schedule risks. Due to most of us having multiple classes and work, we had trouble finding a time to meet up that works for all of us. We communicate via WhatsApp and so far our communication and commitment been successful.

3)  We do not have any Technical risks.

4) We do not have any Teamwork risks.

5) We do not have any Legal/content risks