CEN 4010: Principles of Software Engineering

Fall 2019

**Milestone 3**

**More Detailed Requirements, Architecture and a Vertical Software Prototype**

**Campus Live**

**Code Nxt**

Team 3

Team Members

|  |  |
| --- | --- |
| Sharon Freystaetter | [sfreystaette2015@fau.edu](mailto:sfreystaette2015@fau.edu) |
| Jeff Campbell | [jeffcampbell2016@fau.edu](mailto:jeffcampbell2016@fau.edu) |
| Francois Joseph | [josephf2014@fau.edu](mailto:josephf2014@fau.edu) |
| Kevin Lewitzke | [klewitzke2017@fau.edu](mailto:klewitzke2017@fau.edu) |

Document Versions

|  |  |  |
| --- | --- | --- |
| **No.** | **Revision** | **Date** |
| 1.0 | Initial version | 11/04/2019 |

# 

# **Executive Summary**

Campus Live is a new simplistic way for students and administration can easily communicate with each other regarding issues around campus. College campuses tend to be very large and hard to maintain. Hiring maintenance crews to continuously perform checks would not be cost effective or fully accommodate the students who go to class every day or sit in the library for hours studying. The goal for Campus Live is to close the gap for students to easily provide feedback to administration about the condition of the campus.

Campus Live is a friendly and easy to use website that all students will be able to access with their school mandated email and login. On the website, students are able to report anything they see on campus. If there is a broken lamp post, students can easily take a picture and post the issue for the proper administration to take action and fix the issue faster than before. It can capture the small nuances that only a student who spends their days traversing every region of campus can notice.

Campus Live does not only allow for students to find faults around campus, they will also be able to comment on things they enjoy about campus. For example, maybe there was an activity hosted last week to promote moral on campus, students are able to post their experiences and thoughts about the event to the Campus Live website so that administration can have instantaneous feedback on what best benefits the students.

# 

# **Competitive Analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| **Campus Live features** | **MyFAU features** | **Facilities module by ModoLabs features** | **Campus Eye features** |
| Ability to report campus issues | No | Yes | Yes |
| View existing campus issues reported by other users | No | No | No |
| Ability to view the status of existing campus reports | No | No | No |
| Ability to report upcoming campus events | No | No | No |
| View upcoming campus events | Yes | No | No |

Campus Live, Facilities by ModoLbs, and Campus Eye all allow the user to report a problem on campus but only Campus Live also allows the user to view other reported problems and check the status of reported on-campus problems. Administration is alerted to potential problems reported on Campus Live and can act quickly to resolve any issues that have been reported. Campus Live streamlines the “identify, report, and fix the problem” process so that campus life is overall a much safer and enjoyable experience.

Both Campus Live and MyFAU provide the ability for users to view upcoming campus events, but only Campus Live gives all users the ability to report upcoming campus events. Now students and teachers can advertise some of the smaller events, fundraisers, and gatherings on campus that are not officially advertised through MyFAU.

# **Data Definition**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Comment | Additional information added to an entry after it is created. Each comment contains, at a minimum, the following attributes: description, date added, and user ID. |
| Entry | A record added to the system containing University-related information. Each entry contains, at a minimum, the following attributes: type, title, description, date added, user ID, priority, and status. |
| Event | A system entry-type that describes events on campus which would be of interest to students, faculty, and staff. |
| Faculty/Staff | A system user-type belonging to faculty and staff at the University. |
| Issue | A system entry-type that describes a problem at the University that requires maintenance attention. |
| Maintenance | A system user-type belonging to maintenance personnel at the University. |
| Photo | An image file which may be attached to issue and event system entries and comments. |
| Report | Summary or detail document that describes system entries. |
| Status | An attribute that describes the current status of issue and event system entries. For issues, the possible values are Open, Work In Progress, On Hold, Additional Information Needed, and Resolved. For events, the possible values are Upcoming, Canceled, and Completed. |
| Student | A system user-type belonging to students at the University. |
| System Administrator | A system user-type belonging to individuals at the University who have complete control over the configuration and operation of the system. |
| User | A person who uses the system. |

# **Overview, Scenarios and Use Cases**

Campus Live serves to primary purposes: to provide a means for reporting maintenance issues on campus, and to provide a means for communicating upcoming events to the University community, including students, faculty, and staff. Both general aspects of the application can be accessed by all students, faculty, and staff, as users of the application. However, each user’s specific ability to view or edit information about issues or events may be limited by his or her role in the system. Main use cases of the application are explained below.

## **Use Case #1: Report an Issue**

A student logs on to the Campus Live service to report a problem. The student selects the option that says “Report a Campus Problem” from the menu. The student is taken to the “Report a Problem” page. On this page, the student types a textual representation of the problem, including its location on campus. The student then selects a button to upload an image of the problem. The student selects the image from their device. The student then selects a button to submit their report.

## **Use Case #2: Look Up an Issue**

A student identifies a problem on campus. The student logs on to the Campus Live service, with the intention of seeing if the problem has already been reported. The student selects the option that says “View Reported Problems” from the menu. The student is taken to the “View Problems” page. On this page, the student is presented with a list of problems that have already been reported. The student selects the link to the problem that seems most like the problem he or she has identified. The student confirms the problem that he or she identified has already been reported, using the textual description and photo.

## **Use Case #3: Check the Status of an Issue**

A student has already reported a problem on campus, and is curious as to whether or not it has been resolved. The student logs on to the Campus Live service, and selects the option that says “View Reported Problems” from the menu. The student is taken to the “View Problems” page. On this page, the student is presented with a list of problems that have been reported recently, and he or she is also able to search for problems that are not listed by default. The student searches for a problem, using its ID number or another attribute. Once the student locates the problem, he or she can view its current status and other details.

## **Use Case #4: Create an Event**

A student is having a small event on campus, and wants to notify others who may be interested in attending the event. The student logs on to the Campus Live service, and selects the option that says “Add Event” from the menu. The student is taken to the “Add Event” page. On this page, the student selects the date and time of the event, and enters a textual representation of the event, including why others should attend. The student then selects an option to upload a photo that represents the event.

## **Use Case #5: Look Up Events**

A student has logged into the Campus Live service to view upcoming events that are happening around campus. The student selects the option that says “View Campus Events” from the menu. The student is taken to the “View Campus Events” page, where he or she is presented with a list of events on campus. The student selects the link to an event that is of interest. The student is then provided with details about the event, and even a photo that represents the event.

# **Functional Requirements**

|  |  |  |
| --- | --- | --- |
| **Req #** | **Priority** | **Requirement** |
| 1.1 | 1 | All user types shall be able to enter new issues. |
| 1.1.1 | 1 | A title for the problem is entered into a text box. |
| 1.1.2 | 1 | The location of the problem is entered into a text box. |
| 1.1.3 | 2 | A photo of the problem may be uploaded by clicking the Browse button, which gives the system access to the photos on the user’s device. |
| 1.1.4 | 1 | A detailed description may be entered into a text box. |
| 1.1.5 | 1 | The problem is assigned a unique ID number and stored in the system when the user clicks the Submit button. |
| 1.2 | 1 | All user types shall be able to enter new events. |
| 1.2.1 | 1 | The date and time of the event are selected. |
| 1.2.2 | 1 | The location of the event is entered into a text box. |
| 1.2.3 | 1 | A title for the event is entered into a text box. |
| 1.2.4 | 1 | A detailed description of the event is entered into a text box. |
| 1.2.5 | 2 | Photos representing the event may be uploaded by clicking the Browse button, which gives the system access to the photos on the user’s device. |
| 1.2.6 | 1 | The event is assigned a unique ID number and stored in the system when the user clicks the Submit button. |
| 2.1 | 2 | All user types shall be able to create comments and add photos to existing issues. |
| 2.1.1 | 2 | A comment is entered into a textbox on the Problem Detail page. |
| 2.1.2 | 2 | A photo may be uploaded by clicking the Browse button, which gives the system access to the photos on the user’s device. |
| 2.1.3 | 2 | The comment and photo are assigned unique ID numbers and stored in the system when the user clicks the Submit button. |
| 2.2 | 2 | All user types shall be able to create comments and add photos to existing events. |
| 2.2.1 | 2 | A comment is entered into a textbox on the Event Detail page. |
| 2.2.2 | 2 | A photo may be uploaded by clicking the Browse button, which gives the system access to the photos on the user’s device. |
| 2.2.3 | 2 | The comment and photo are assigned unique ID numbers and stored in the system when the user clicks the Submit button. |
| 4 | 1 | Students shall be allowed to view existing entries. |
| 4.1 | 1 | By clicking on the ‘View Reported Problems’ link from the Campus Live home page, the Student is taken to the Reported Problems page. |
| 4.2 | 1 | By clicking on the ‘View Campus Events’ link from the Campus Live home page, the user can view existing events. |
| 5 | 1 | Faculty/Staff, Maintenance, and System Administrators shall be allowed to view and edit existing entries. |
| 5.1 | 1 | By clicking on the ‘View Reported Problems’ link from the Campus Live home page these users can view existing entries. |
| 5.2 | 1 | By clicking on the ‘View Campus Events’ link from the Campus Live home page these users can view existing events. |
| 5.3 | 1 | On the Problem Detail page these users have the ability to edit any item on this page by clicking on the edit button. |
| 5.4 | 1 | On the Event Detail page these users have the ability to edit any item on the page by clicking on the edit button |
| 7 | 2 | Only System Administrators shall be allowed to add or modify system users. |
| 7.1 | 2 | The System Administrator adds a user from the ‘Add User’ page. |
| 7.1.1 | 3 | A profile photo representing the user may be uploaded by clicking the ‘Add Profile Picture’ link, which gives the system access to the photos on the user’s device. |
| 7.1.2 | 2 | The user’s email address is entered into a text box. |
| 7.1.3 | 2 | A user name for the user is entered into a text box. |
| 7.1.4 | 2 | A default password for the user is entered into a password field. |
| 7.1.5 | 2 | The user is assigned a unique ID number and stored in the system when the System Administrator clicks the Submit button. |
| 7.2 | 2 | The System Administrator modifies a user from the ‘User Detail’ page. |
| 7.2.1 | 2 | On the ‘User Detail’ page, the details about the user are displayed in text boxes. |
| 7.2.2 | 2 | The System Administrator modifies any values in the text boxes. |
| 7.2.3 | 2 | The changes are saved when the System Administrator clicks the Submit button. |
| 9 | 2 | The system shall not permit issue entries which have been closed (resolved) for at least 30 days to be re-opened. |
| 10 | 3 | A history of all edits made to comments, entries, photos, and users shall be recorded and available to System Administrators. |
| 10.1 | 3 | These entities shall be displayed to System Administrators as links, or with a textual ‘detail’ link adjacent to the entity. |
| 10.2 | 3 | Clicking the link takes the System Administrator to a page that displays the history of changes to the entity. |
| 11 | 1 | Each system entry shall be classified as either an Issue or an Event. |
| 12 | 2 | Each system entry shall be classified as either High, Medium, or Low priority. |
| 13.1 | 1 | Each issue-type system entry shall have a status of Open, Work In Progress, On Hold, Additional Information Needed, or Resolved. |
| 13.2 | 1 | Each event-type system entry shall have a status of Upcoming, Canceled, or Completed. |
| 14 | 3 | Maintenance users and System Administrators shall be able to generate on-demand reports. The content and layout of each report shall be specified in the Report Specification Document. |
| 15 | 2 | A mobile version of the web application shall be presented to the user when he or she is accessing the system from a mobile device or tablet. The mobile version shall have all functionality of the desktop version, with layout optimized for viewing on a smaller screen. |
| 16 | 1 | All users shall have the ability to search for existing entries, using any or all of the criteria available for each entry type. |
| 16.1 | 1 | The following text boxes are displayed on the ‘View Reported Problems’ page: Date Reported, Reported By, Location, and Description/Keyword. |
| 16.2 | 1 | The user enters values into one or more of the text boxes. |
| 16.3 | 1 | Search values entered by the user are not case-sensitive. |
| 16.4 | 1 | A wildcard character at the beginning and end of each search value shall be assumed by the system. (For example, a search for ‘smit’ shall match an entry containing ‘John Smith’.) |
| 16.5 | 1 | The search is performed when the user clicks the Apply Filter button. |
| 16.6 | 1 | Empty text boxes shall not restrict the search results. |
| 17 | 2 | All users shall have the ability to view in one place all of their own entries, comments, and photos. These entities are displayed on the user’s profile page. |

# **Non-functional Requirements**

|  |  |
| --- | --- |
| **Req #** | **Requirement** |
| 1 | Campus Live shall be compatible with the current, non-beta versions of Mozilla Firefox, Safari, Google Chrome, and Microsoft Edge web browsers, including the versions of these browsers designed for mobile and tablet devices. |
| 2 | The user’s operating system shall not be relevant in the operation or performance of the web application. |
| 3 | All users shall be able to report an issue, look up existing issues, create an event, and look up existing events, without any prior training. |
| 4 | The system should have the latest up-to-date security features for encrypting and securing all users information and data. |
| 5 | The system should comply with all outside legal and regulatory agencies to the best of its ability. |

# 

# **System Architecture and Database Organization**

## High-Level Architecture

The user interface for the web application will be created using HTML5, cascading style sheets (CSS), and jQuery. The Bootstrap framework, which is free and open-source, will supplement CSS, and enhance the attractiveness of the user interface.

Back-end services will be provided by the LAMP Stack, hosted by the Florida Atlantic University College of Engineering and Computer Science. The LAMP Stack contains Linux, Apache, MySQL, and PHP. All data will be stored in a MySQL database. Data will be transmitted between the web application and database using the PHP database connection abstraction library, PHP Data Objects, or PDO. The PDO class contains a data access layer that is vendor-neutral, so other relational databases may easily be used in place of MySQL in the future.

## Database Organization

**Table: Users**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Data Type** | **Null** | **Key** | **Extra** |
| znum | int(8) | No | Primary Key |  |
| type | char(1) | No |  | A : System Administrator  F : Faculty/Staff  M : Maintenance personnel  S : Student |
| first | varchar(50) | No |  |  |
| last | varchar(50) | No |  |  |

## **Table : Events**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Data type** | **Null** | **Key** | **Default** | **Extra** |
| event\_id | int(11) | No | Primary Key | null | auto increment |
| event\_name | varchar(255) | No |  |  |  |
| event\_description | text | Yes |  | null |  |
| event\_date | date | No |  |  |  |
| event\_time | time | Yes |  | null |  |
| event\_location | varchar(255) | Yes |  | null |  |
| created\_by | int(11) | No | FK references Users(znum) |  |  |
| event\_status | int(11) | No |  | 1 | 1 : upcoming  2 : canceled  3 : completed |
| photo\_URL | varchar(255) | Yes |  | null |  |

## 

## **Table : Issues**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Data type** | **Null** | **Key** | **Default** | **Extra** |
| entry\_id | int(11) | No | Primary Key |  | auto increment |
| reported\_by | int(11) | No | FK references Users(znum) |  |  |
| date\_time\_reported | datetime | No |  | now() |  |
| title | varchar(255) | No |  |  |  |
| location | varchar(255) | No |  |  |  |
| description | text | Yes |  | null |  |
| priority | tinyint(4) | No |  | 3 | Rank from 1 - 5 |
| status | tinyint(4) | No |  | 1 | 1 : open  2 : work in progress  3 : on hold  4 : additional info needed  5 : resolved |
| photo\_URL | varchar(255) | Yes |  | null |  |

## **Table : Comments**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Data type** | **Null** | **Key** | **Default** | **Extra** |
| comment\_id | int(11) | No | Primary Key |  | auto increment |
| entry\_id | int(11) | Yes | FK references Issues(entry\_id) | null |  |
| event\_id | int(11) | Yes | FK references Events(event\_id) | null |  |
| entered\_by | int(11) | No | FK references Users(znum) |  |  |
| comment\_date\_time | datetime | No |  | now() |  |
| description | text | No |  |  |  |
| photo\_URL | varchar(255) | Yes |  | null |  |

## Media Storage

The only media used will be static images. The images will be stored in the Linux file system, and file names will be referenced as values in the database. Images will be uploaded from the user’s device to the Linux file system using standard PHP file upload functions.

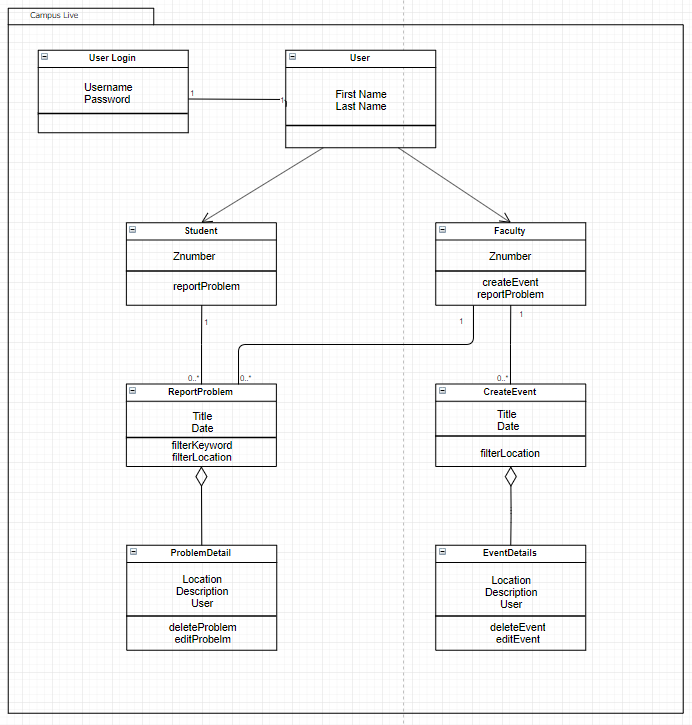
## Search/Filter Architecture

All search will be performed using the InnoDB database engine in MySQL. Search and filter criteria will be formed into a SQL query, which will be executed on the database using PHP. The result set will be parsed using PHP and displayed to the user using HTML. All fields which may be used as search or filter criteria will be indexed. The InnoDB database engine uses a B+Tree (or clustered b-tree) structure for its indexes. This type of index is ideal when searching with wildcard characters.

## APIs & Significant Algorithms or Processes

No APIs or significant algorithms or processes will be created.

# **UML Diagram**



# **Project Key Risks**

**Schedule risk:** Our team has committed a breadth of features, which may be too ambitious for the hard project deadline of Dec 2, 2019. We are mitigating this risk by prioritizing requirements, and completing features in order of priority. The front-end and back-end Team Leads are staying focused on completing their respective tasks. The Project Owner is reviewing and re-prioritizing requirements as the project deadline approaches. The ScrumMaster is ensuring communication is ongoing.

# **Team**

|  |  |
| --- | --- |
| **Team Member** | **Role** |
| Jeff Campbell | ScrumMaster |
| Sharon Freystaetter | Product Owner |
| Francois Joseph | Back-end Lead Developer |
| Kevin Lewitzke | Front-end Lead Developer |