Santa Clara University

Computer Engineering Department

Software Engineering

Alumni Engagement Recording System

By

Ronak Gajrawala

Yutong Li

Santa Clara, California

November 26th, 2018

Table of Contents

Abstract

As of now Santa Clara University does not have a system in which alumni engagement and attendance is recorded for both official and unofficial alumni events. To better understand what aspects of an event affect alumni engagement Santa Clara University needs an online system in which alumni can find events and submit whether or not they will attend as well as any feedback about the event. This report describes a project to create a web interface for alumni events and engagement recording. The basic system is a website where SCU can post official events and approve alumni-submitted events so that alumni can see and interact with these events online. This new system will add a new tool for the SCU Alumni office as well as replace the current event calendar system.

1. Introduction

This report details the various design aspects of the Alumni Engagement Recording System for Santa Clara University. These aspects include the several use-cases of the system, conceptual models, technologies chosen, and architectural design.

This design has three main objectives: create an event calendar with weighted event display, have interactive events for engagement data, and provide a system for SCU administrators to approve and edit events and to view collected data from the interactive events.

The first objective is to create an event calendar for alumni events that displays events with a weighted system in which more important events are displayed first. We plan to achieve this by automatically assigning weights to events based on several factors such as whether the event is official, if the event is recurring, and the type of event. SCU administrators will also be able to modify this weighting system as they seem fit.

The next objective is to have the events on the calendar be interactive to record engagement data. This includes attendance records and satisfaction scores after the event is over.

Lastly, SCU alumni Office will be provided with a system in which they can approve, edit, or remove events on the alumni calendar. This system will also provide access to the data collected from the interactive events.

This report details all the design choices we have made regarding these objectives.

2. Requirements

There are a multitude of requirements that our system must meet to be considered fully-featured and functional. The two sections of requirements are user-interface requirements and administrative control requirements.

The system must provide several functionalities to users. Users must be able to view upcoming alumni events chronologically with significant events shown first. Users must also be able to engage and interact with the events by indicating plans to attend, giving satisfaction scores after events, and requesting email reminders for events. Lastly, this must be achieved without the need for user logins. Users of this system should be able to use all the functionalities by only providing their email address.

The system must also provide a framework for SCU administrators to modify and approve events that show up on the alumni event calendar. Administrators need to be able to change event descriptions and information and approve and post unofficial events that are submitted by SCU alumni. This administrative tool will also need to be able to give all of the data recorded by the interactive events so that administrators can analyze and learn about user engagement.

3. Use-Case

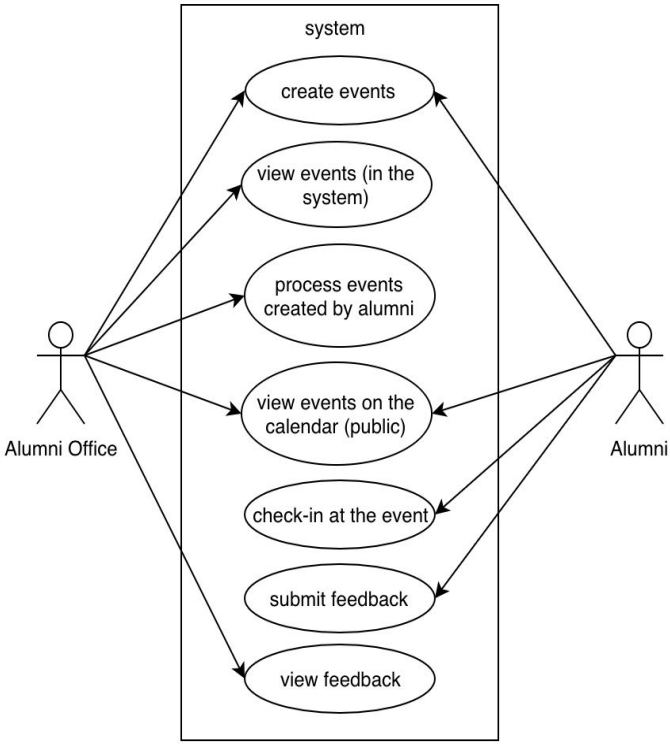

Figure 1: Case diagram of the system

With the system, the Alumni Office is able to create events and approve events created by alumni, while the alumni can use the system to create their own hosted events.

For the events that is created by the admin, it will be on the calendar automatically after creation. For the events created by the alumni, they will be on the calendar if they are approved by the admin.

For all events, the alumni can check-in using the system and submit feedback afterwards.

The admin can then use the feedback collected to decide what kind of events are more popular and therefore hold more of this kind of events.

4. Activity Diagrams

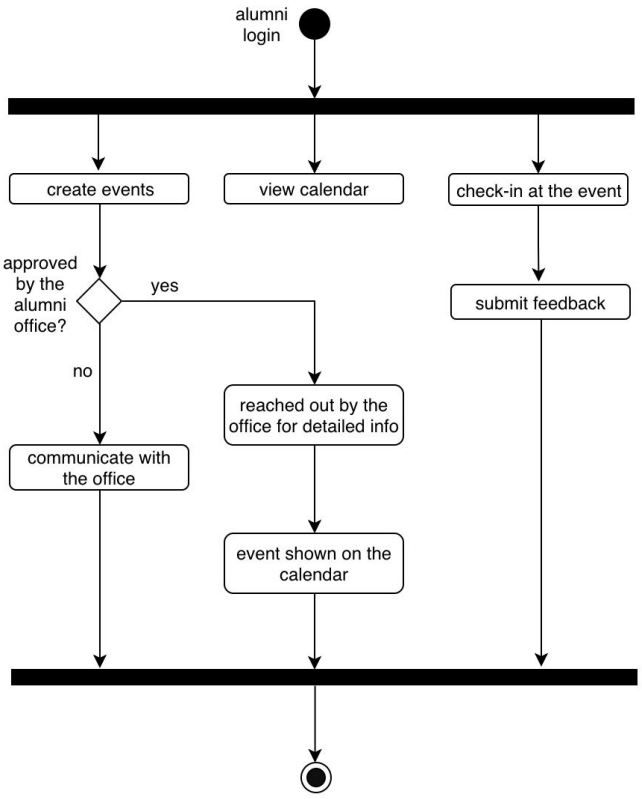

Figure 2: Activity diagram for alumni use

This activity diagram is what the alumni can do with the system.

Through the system, the alumni can create their own events as well as view the event calendar. When the events that the alumni create is approved by the admin, the admin will reach out to the alumni for more detail about the events. After that, the events will be posted on the calendar. If the events are rejected, the alumni can choose to communicate with the office privately to discuss.

When the alumni attend any event, they can check-in at the event using the system, and submit feedback afterwards.

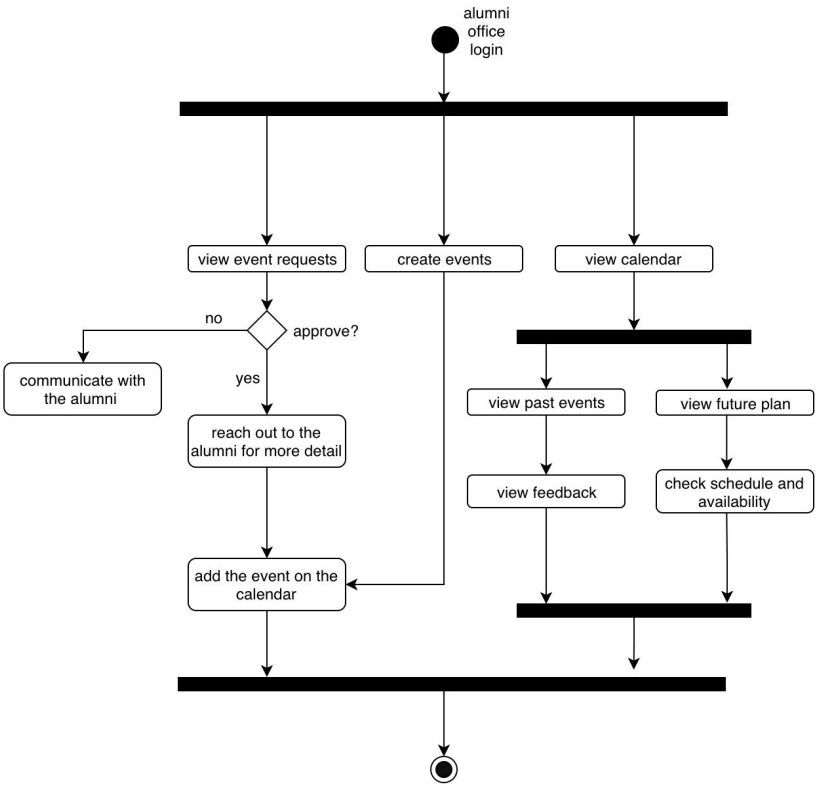

Figure 3: Activity diagram for alumni office use

This activity diagram is what the alumni office can do with the system.

Through the system, the admin can view event requests by the alumni, view the event calendar, and create official events.

When the admin decides to approve certain event, the admin will reach out to the alumni who created it for more information about the event. When both sides agree on all details about the event, the admin can then add the event on the calendar. When the admin decides to reject an event, they can communicate with the alumni privately.

By viewing feedback of the past events on the calendar, the admin can learn what kind of events is more popular or immersive and then hold more of these events in the future. By view the calendar for future plan, the admin can check availability and therefore avoid conflict while adding events to the system.

5. Technologies Used

Since we are utilizing the Design Center to host our website, we do not have many technology choices to choose from. We used a SQL database and MySQL for database management since these are the only database technologies that the Design Center utilizes. For the backend of the server, we were forced to use the Apache server and PHP backend language that are already configured to work for the Design Center. We incorporated a PHP framework, i.e. Laravel, to make backend development more structured and easier to implement. The backend serves as an internal API for the frontend to query. The frontend of the website is a single-page application implemented using a JavaScript framework, namely React. In addition, we used the Bootstrap framework to help style our website. Finally, we used Webpack as our frontend asset manager to simplify the frontend development process.

6. Architectural Diagram

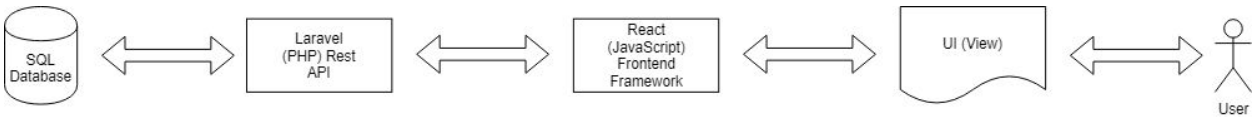

Figure 4: Architectural diagram

The architecture is linear and follows the “high cohesion, low coupling” model. Each component of the architecture only interacts with two adjacent components at the most, and simply modifies and passes data from one component to another.

From client-side to server-side, actors interact with the user interface (UI) or webview. These interactions get translated into events that React, the Javascript UI framework uses to make requests to the API based on how the user interacted with the UI. The PHP API then runs some additional validation on the request from the front-end and fetches and/or updates data from the database based on the action that the user wanted to perform. The result is then passed back to the front-end framework, which modifies the UI to let the actor know that their request has executed and whether the action was successful or not.

7. Design Rationale

We were forced to use SQL, MySQL, Apache, and PHP to implement the backend because other technologies are not supported by the Design Center. We were using the Laravel PHP framework to significantly simplify and bolster backend development, keep our backend code clean and easy to understand, and ensure that our internal API is REST-compliant. We were implementing an internal REST API and SPA frontend so that the initial load time for our website is fast, as many users will probably load the website over a mobile connection weekly or biweekly. We chose to use the Bootstrap frontend framework to make our site's UI mobile-friendly without having to waste time writing our own mobile-friendly code and styles. Finally, we decided to use Webpack as our asset pipeline manager so we can write modular and loosely-coupled code without having to worry about writing HTML code to reference all of the source files we created.

8. Description of System Implemented

The system implemented uses a website that displays and allows alumni to create alumni events in a certain time period and offers moderation capabilities for posted alumni events to the Alumni Office. Events that have been created with confirmed emails and approval from an administrator can be viewed in the calendar view by anybody. Only administrators can view events that have not been approved in the calendar view. Events are created by filling out the event creation form and filling in all the inputs. For non-administrative users, a confirmation email will be sent to the inputted contact email with a link to confirm the event and email. When administrators create events while being logged into the site, the event will automatically be confirmed and approved. Administrators can log in with a pre-configured email and password that can be set up using the Design Center's PHPMyAdmin server. Once logged in, administrators can also view reports. All site visitors can check-in to events that are currently in progress by clicking on the event in the calendar and filling out the check-in form.

9. Testing

Our plan for testing was to have four testing phases: manual functional testing, automated functional testing, pen testing, and security testing. To complete our initial site on time, we could not allocate resources to write automated functional tests, so we decided to manually test parts of the site that were required for the initial demo. These included the sign-in, event submission, calendar, and reporting pages. Manual testing worked fine because we were only testing four parts of the site. Eventually, we wanted to move on to automated functional testing of the entire site for our final demo. We decided to use Jest, a testing framework that worked well with React, since it was easily to implement. However, Jest was not that useful for testing the API itself. As a result, we wanted to run manual security testing on the backend to make sure that our site was secure and could securely respond to improper requests. We went through the code to make sure that the logic was correct and ran manual pen testing to make sure our site was secure. We used examples from the Open Web Application Security Project (OWASP) to test the top ten most common vulnerabilities. As a result, we made sure that our site was bug-free for the final demo.

10. Difficulties Encountered

Unfortunately, one of our teammates had to leave the team in the middle of the project because of his illness, which left the remaining two of us more workload. As a result, we need to re-assign the remaining tasks and adjust the corresponding deadline. We made a decision to have one focus on back-end and the other focus on front-end.

It had been hard for both of us to achieve efficient workflow along the project. The main problem we had is that we were unable to update code with less overhead. We were using Github to store our code. Each of the teammates had their own branch and the file they upload would be on their individual branch first. The file need to be approved by the administrator in order to be added to the master branch. Besides, whenever the master branch was modified, we need to clone it to stay updated.

We also found incompatibility between technologies. There are discrepancies between PHP and Javascript objects. The difference is mostly in syntax, as Javascript is not an Object Oriented language, but it does have objects, python on the other hand is very object oriented. The syntax is quite different, Javascript uses the C notation and uses curly brackets for closure and blocks, while python just uses spacing. We used PHP and Javascript a lot during the project, and it took some time to adjust to the language that we were currently using from the former one.

We were not experienced with some technologies that we were using, especially React, which is a Javascript library for building user interfaces. We decided to use React because it is a simple yet powerful UI library and its architecture extends beyond HTML. The team spent some time together to learn and get used to React.

11. Suggested Changes

Tags that the user can add to the event, so the user can view events on not only calendar in chronological order, but also by the type of the events. For example, the user is viewing an event that has tag that says “food”. When the user clicks this tag, the user can view all future events that also have the tag “food”.

A search bar for specific event(s) so the users can look for the events that they are interested in by name, time, type of the events or other characteristics.

Improve user interface. This is very broad, but we think the aspect that we need to work on the most is on simplifying the main page. Currently our default main page is the calendar of the current month, so there might be maybe unused space. In order to use the space more efficiently, it would be better to have a page of the events in the form of table, but not in the form of calendar.

12. Lessons Learned

We learned most of the lessons the hard way mainly because we did not consider the risk of missing any teammate.

The project would go much more smoothly if we had more detailed planning, especially the alternatives if anything unexpected happens. While planning for the project, it might seem unworthy to spend time on thinking about Plan-B because we believe that things will go as planned and even if there is any change, we can handle the matter as the circumstances require then.

Our skill in time management is insufficient so that we sometimes missed the deadlines. We did not have an overall plan for the entire project. Instead, we started to plan for the next few weeks when we finished the current tasks. For any project, it is necessary to plan for the entire thing at the first place, so we can have a general idea of the project. It is important to follow the plan so we can meet the deadlines.

We also learn about the importance of work allocation in this project. Doing a team project does not necessarily require to split the work evenly within a team. It is helpful for both the project and the individuals if we can play to our own strengths. At the beginning of the project, when we still had three teammates, two of us who have experience with back-end were responsible for back-end while the other one who has experience with front-end was working on front-end. In the middle of the project, we lost the teammate who was working on the front-end. Since Ronak is more experienced in back-end than Yutong, we assigned Yutong to work on front-end, which worked out fine.

13. Table of Figures

14. Appendices

Installation guide for Front-End Development:

- Install node/npm. nvm recommended.
- Change working directory to src (e.g. run cd src)
- Run npm i to install dependencies for front-end development.
- Run npm start to run build script. The script will automatically watch for new changes.

User manual:

Intention	API Routes
Get events	GET /v1/event/all
Get specific event	GET /v1/event
Create event	POST /v1/event/create
Check in for event	POST /v1/event/checkin
Confirm event (email confirmation)	GET /v1/event/confirm
[admin] Approve/publish/verify event	GET /v1/event/approve
[admin] Get reports for all events	GET /v1/report/all
[admin] Get report for single event	GET /v1/report
[admin] Log in	POST /v1/admin/login
[admin] Log out	GET /v1/admin/logout
[admin] Verify alumni status	GET /v1/admin/alumni