# <u>ServiceSearch</u>

Garrett Quintero, Emily Greiman,
Chamal Kayssar, Clara Tran

CSM Bridgman
15 June 2017

# Table of Contents

**Table of Contents** **2**

**1 Introduction** **5**

1.1 Client Description 5

1.2 Product Vision 5

**2 Requirements** **6**

2.1 Functional 6

2.1.1 Website 6

2.1.2 Users 6

2.1.3 Database 7

2.1.4 Google OAuth 7

2.1.5 Server 7

2.2 Non-Functional 7

**3 System Architecture** **9**

3.1 Front-End 9

3.1.1 Events 9

3.1.2 MyEvents 9

3.1.3 Search 9

3.1.4 Admin 9

3.2 BackEnd 9

3.2.1 Database 9

3.2.2 PHP 10

**4 Technical Design** **11**

4.1 Event Management 11

4.1.1 Creation 11

4.1.2 Read 11

4.1.3 Update 11

4.1.4 Deletion 11

4.2 Database 11

4.2.1 Tables 12

4.2.2 Interface 12

4.3 Access Control 12

4.3.1 OAuth Description 12

4.3.2 Credentials 12

4.3.3 User Interface 13

Technical References 13

**5 Decisions** **14**
5.1 Front-end 14
5.2 Back-end 14
5.4 Security 15
5.5 Database vs File System 15

**6 Results** **16**
6.1 Unimplemented Features 16
6.1.1 Emails 16
6.1.2 Event Completion Status 16
6.1.3 Admin Features 16
6.2 Testing 16
6.2.1 16
6.2.2 16
6.3.1 17
6.3.2 17
6.3.3 17
6.3.4 17
6.4 Lessons Learned 17
6.4.1 17
6.4.2 17

**Appendices** **18**

**A. Documentation** **18**
A.1 Pages 18
A.1.1 Home page 18
A.1.2 My Events 18
A.1.3 Events 19
A.1.4 Make Event 19
A.1.5 Individual Events 19
A.1.6 Contact Us / About 20
A.1.7 Administration 20
A.2.1 $_SESSION Variables 20
A.2.2 $_GLOBAL Variables 20
A.2.3 $_SERVER Variables 20
A.3 The Database and Tables 21
A.3.1 Events Table 21
A.3.2 Volunteers Table 21
A.3.3 Users Table 21

A.4 Files to know about 22
A.4.1 Footer and Header 22
A.4.2 Style.css 22
A.4.3 events.sql 22
A.4.4 glogin.php 22
A.4.5 confirmPage.php 22
A.4.6 editFunction.php 23
A.4.7 leavePage.php 23
A.4.8 Section.php 23
A.4.9 latest_events.php 23
A.5 Google oAUTH 23
A.5.1 How it works 23
A.5.2 Set it up 23
**B. Development Environment** **25**

# 1 Introduction

## 1.1 Client Description

The Mines Philanthropy Council is a club here on campus that aims to promote a Mines-centered culture of philanthropy. They play a role in communication with donors and provide a voice for the students about fund allocations. On campus, they can be seen helping fundraisers, projects, and assisting events, such as Philanthropy Days. The council encourages their peers to give back to the community and have reached out to the CS department to help them develop a tool to assist community service events.

The MPC is implementing a website called Service Search. Service Search is a social media-like site that allows people and organizations to post community service events and find people to help run the service. At Colorado School of Mines, there is a disconnect between people who are looking for community service and organizations who need people to work at their service events. The website will bridge the gap between the two parties and provide a service that allows easy and quick connection to community service events and volunteers. Mines organizations, Greek Life, or nearby communities can post events that are in need of volunteers. From there, any Mines student will be able to look for an event and sign up for the community service.

## 1.2 Product Vision

The goal this summer field session was to deliver a fully functional Service Search site that could be deployed at the end of the session. The product will allow for volunteers to look up community events, sign up for events, and drop out of an event if need be. The organization posting the event should be able to post details of what, where, when, and also how many volunteers they need. We will be delivering the product as a website that can be set up on a server. Since it is unknown whether CS majors will continue to monitor the site afterwards, extensive documentation was provided to help hand the project off so that it may be used, expanded, and monitored by Mines students long after we graduate.

# 2 Requirements

## 2.1 Functional

The website itself will have three types of users: clients, volunteers, and admins. Clients will post events that volunteers will sign up for. Administrators will be able to keep the health of the site by being able to see some statistics, remove events, remove users, sign-up a user for an event, or drop them from an event. All information that pertains to the event will be stored in a table on a database for the site to access and display dynamically.

### 2.1.1 Website

- Navigation Bar
    - The navigation bar holds the site together, allowing the flow from one part of the website to the next. This bar lies on every page except the admin pages. It provides a quick link back to the home page and offers the option to either login or logout. It displays the options to see the events, about page, and the events the user is a part of. If the signed in user happens to be an admin, then there is an option to go to the admin pages. Lastly, the option to search for events resides in the navigation bar.
- Home Page
    - The home page provides all information that one would expect a home page to have. It links to the other parts of the site and displays a small list of recently created events. For aesthetic reasons, a slideshow of Mines was added to the top.
- Events Page
    - After some text has been entered into the search field, the search will look through the database for similar event name and tags.
    - No event that has passed will appear.
    - The user will be able to click on an event and will be brought to the event's page.
    - The page will have a restricted view to those who are not logged in. The non-logged in view will not allow a user to create an event nor sign up for one.
    - Side column displays the date and name of 10 events that were recently posted
- The community service event page
    - Non-logged in users cannot volunteer for the event.
    - The page displayed is blank in reality and fills in depending on event id. The page is dynamic and displays information pulled from the database.
- MyEvent Page
    - The page will display the title and date-time of all events that the logged-in user has signed up for as well as created.

### 2.1.2 Users

- Clients
    - The organizations will be able to submit a service event that specify the number of volunteers needed.

    - They will be able to post where and when the event is, and what will be expected of the volunteers.
    - They should also be allowed to upload a logo with size restrictions.
- Volunteers
    - Will be able to sign up for an event.
    - Can back out of an event if need be.
- Administrators
    - Can keep track of stats of the site
    - Can register a Client or volunteer to an event.

#### 2.1.3 Database

The database will hold information of all clients, volunteers, and events. This information is broken up into 3 tables. The first table is called events and it holds information for the event id, creator id, owner, event title, description, tags, picture location, start and end time, location, email, timestamp, and volunteers signed and needed for the project. The events table is updated every time an event is added or edited. The second table is named users, it holds the Google account id, the user's name, the user's email, and whether a user is admin or not. Users is added to when a new user logs into the site. Lastly, the volunteers table holds event id and user id. The table is edited every time someone signs on to an event or signs off an event.

#### 2.1.4 Google OAuth

The google login API was implemented to verify users for events. The button leads the user to the google login and from there Google verifies the user and stores all the sensitive information for us. Google OAuth only returns names, emails, and whether it is a verified account.

#### 2.1.5 Server

The project was developed on an Apache server. The server is fairly simple and widespread, as such it is easy to port from Apache to a different kind of server if need be. This relative simplicity will make handing off the website easy to our client easier.

## 2.2 Non-Functional

The program must be easily extendable. The life cycle of the product may go beyond our team's time here at Mines. As such the program will be written in HTML5, CSS, and PHP as these are fairly common languages. The program was developed using WAMP given all that we needed to use was bundled up in this easy to use package.
The website itself must show polish. For the life cycle of the product, looks and accessibility is a must. The user must easily find projects and post community service opportunities. It must look like a well developed website.
The web service itself is easily accessible and easy to navigate. Taking full advantage of the internet, the service provides users with the information they need to connect to the community. As such the site focuses a lot on UX design and tries to be appeasing to the eye. The service connects volunteers with community service organizers in a convenient way. The success of this

product relies on being a more streamlined process of organizing an event and looking for an event. The design of the site reflects this idea as users can easily find events and sign up for them as well.

# 3 System Architecture

## 3.1 Front-End

The website is linked together by the navigation bar located at the top of the page. The connections are laid out in Figure 3.1.1 where each page can connect back to the Home page as well as to one another. User can also login or logout on any page.

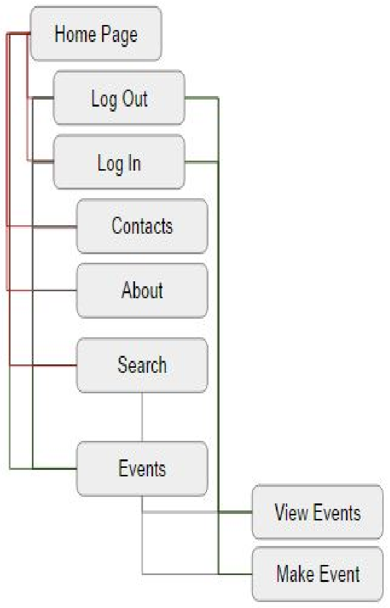


Figure 3.1.1 - Website Page Connections

### 3.1.1 Events

Events page displays an overview of all events that are currently available to volunteer for. Events Page also links to the Make Event page as well as individual event pages.

### 3.1.2 MyEvents

MyEvents page will display all of the events the user has created and volunteered for.

### 3.1.3 Search

Search Page goes hand-in-hand with Events page where events that link to the search term the user inputs are displayed.

### 3.1.4 Admin

Admin page only links back to the Home page.

## 3.2 BackEnd

The website is hosted on a server to allow the user to access it by the URL thesystem.mines.edu/ServiceSearch. The host is a simple stack setup, shown in Figure 3.1.1 below, with Apache, MySQL, PHP, running on a Windows computer.

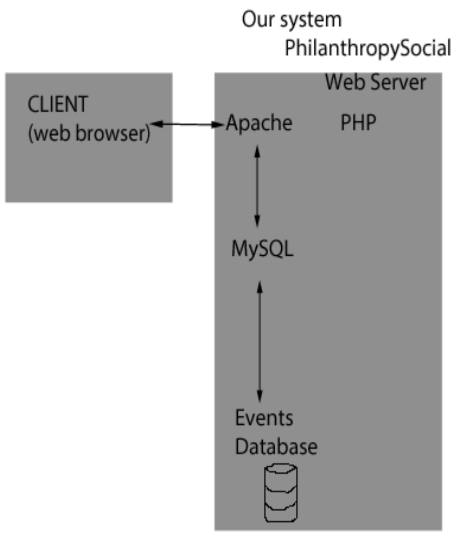


Figure 3.2.1 - User-Server Layout

### 3.2.1 Database

The project used a MySQL database. It holds three tables to store website information: events, volunteers, and users (shown in Figure 3.2.2). The website pulls different pieces of information depending on the page. The database cuts down the need for a complex file system. Most pages dynamically display events according to query pulls. Empty fields are filled with information from the events table.

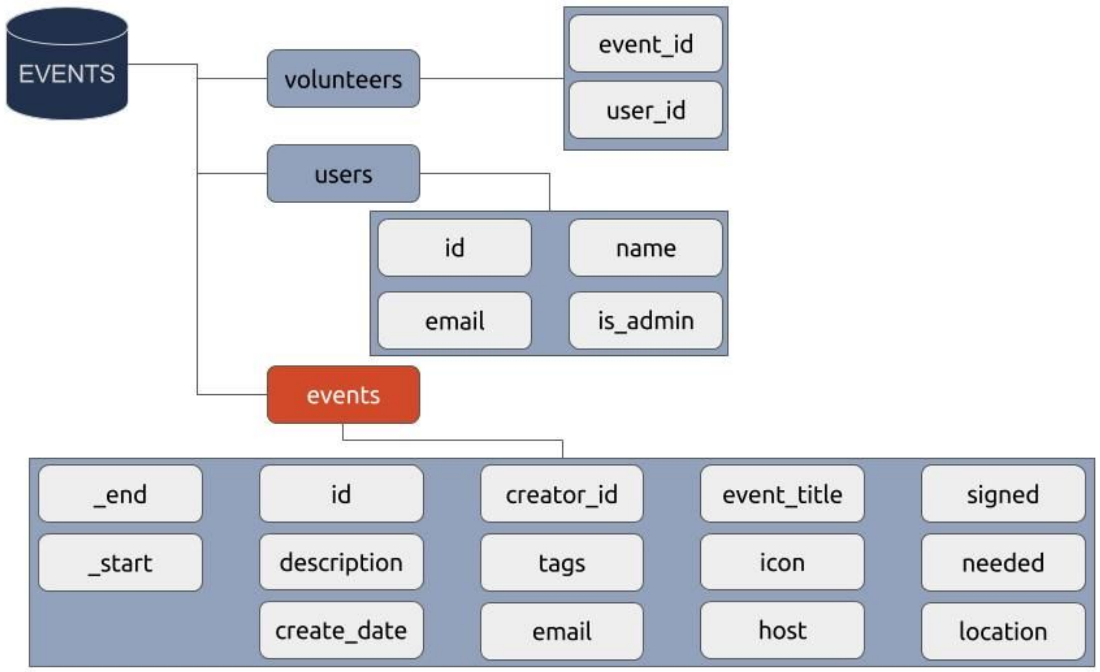


Figure 3.2.2 - Database Tables Layout

### 3.2.2 PHP

PHP can pull information from the database through MySQL commands and is used to integrate the back-end with the front-end. PHP is also used to store information to the database.

# 4 Technical Design

This project implements a website for Mines Philanthropy Council. The website provides a full user interface to a MySQL database. User access control is integrated with Google OAuth, allowing sign in from Google (implemented) or other (unimplemented) accounts.

## 4.1 Event Management

The website provides a full interface to the event database. All primary database interactions (Creation, Reading, Updating, and Deletion) are implemented through web forms and webpages. PHP is used to link the website to the MySQL database.

### 4.1.1 Creation

Event creation is handled via web form. As the user enters information for an event, any incorrect submissions are thoroughly checked through Javascript. Javascript will send an alert, stating what needs to be fixed. This includes: properly ordered start and end dates, image upload is within reasonable size and with accepted image extension (e.g. jpg, jpeg, or png), and a proper number of volunteers is inputted (i.e. greater than zero). Any HTML input type with a red star appearing next to them are required and the system will warn the user when the form is not fully filled out. A PHP script then parses the web form and creates a new entry in the database.

### 4.1.2 Read

Event information is displayed at two levels. An overview page shows basic information on upcoming events as well as recently posted events. A details page displays detailed event information. Both pages pull event information from the database.

### 4.1.3 Update

Appropriate users can edit events. The event update page is accessible from the event details page. A webform similar to the event creation page is filled with current event information. The user can change event fields as necessary. A PHP script updates the event's row in the database. User is able to cancel their edit simply by clicking outside the model or clicking on the 'X' button located at the top-right corner of the model.

### 4.1.4 Deletion

Appropriate users can delete events. The event deletion is accessible from the event details page. A PHP script deletes the event's row in the database.

## 4.2 Database

User Access Control ensures that only appropriate users can make important changes to events or other information. This website integrates Open Authentication (OAuth), which provides a common format for user credentials. This allows integration of sign up and sign in with other web services. This project implements sign in with Google accounts.

### 4.2.1 Tables

As shown in Figure 4.2.1, the database contains 3 tables of interest: Events, Users, and Volunteers. The Events and Users tables store information on events and users, respectively. The Volunteers table stores (event foreign key, user foreign key) pairs that link volunteers to the events they have signed up for.

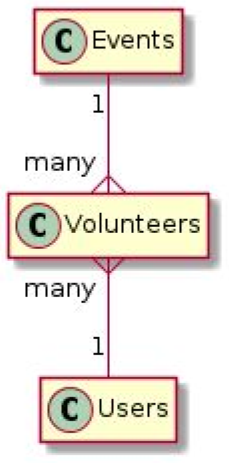


Figure 4.2.1 - Database relationships

### 4.2.2 Interface

All user interactions with the database are through the website.

## 4.3 Access Control

User Access Control ensures that only appropriate users can make important changes to events or other information. This website integrates Open Authentication (OAuth), which provides a common format for user credentials. This allows integration of sign up and sign in with other web services. This project implements sign in with Google accounts.

### 4.3.1 OAuth Description

OAuth 2.0, Open Authentication, is an open source authentication service that allows authentication with credentials from other trusted websites. From the OAuth community website: “OAuth 2.0 is the industry-standard protocol for authorization. … OAuth 2.0 focuses on client developer simplicity while providing specific authorization flows for web applications, desktop applications, mobile phones, and living room devices.” (OAuth 2.0)

OAuth simplifies both users and web administrators. Users can log in with credentials from other trusted and respected websites, such as Google. Web Administrators are spared the hassle of protecting user credentials.

OAuth uses public key cryptography to allow the transmission of irrefutable evidence of log in through a third party service. The user credential is a message digitally signed by the third-party that names the user and demonstrates sign in within a small time window.

### 4.3.2 Credentials

The web server requires several pieces of information from Google to process a log in request. They include: client id, client secret, and redirect URI. All necessary variables are stored in the beginning of glogin/glogin.php in the website directory. Client ID and Client Secret are how Google identifies the website. These codes are obtained by creating an OAuth token in the Administrators Google API Console, which can be accessed at:

https://console.developers.google.com/apis/credentials . The redirect URI tells Google where to send the user after log in.

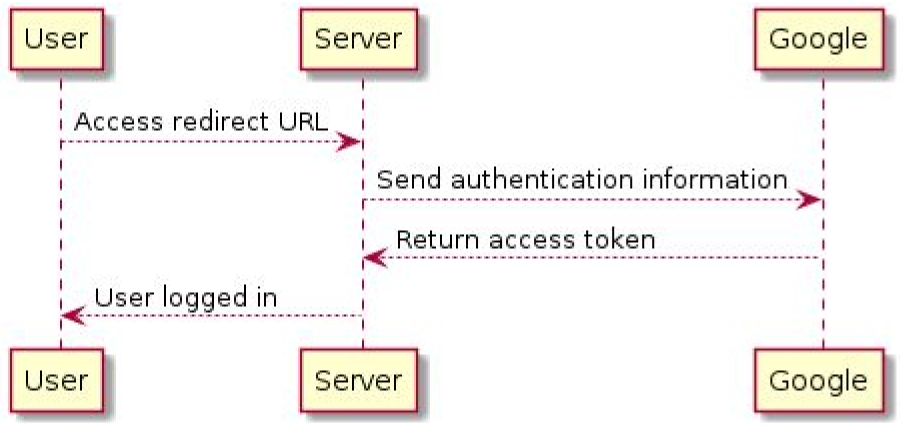


Figure 4.3.1: Action diagram for Google OAuth log in.

### 4.3.3 User Interface

User sign in through OAuth is done via a hyperlink available on all pages.

### Technical References

OAuth 2.0 Community Authors (OAuth 2.0) "OAuth 2.0" https://oauth.net/2/ Retrieved: 6/17/2017

## 5 Decisions

### 5.1 Front-end

When it came to designing the pages that the clients would see, we used HTML, CSS, and JavaScript.There are a lot of resources for each and have proven to be reliable. All three can work together on a single page and offer flexibility in design and aesthetic choices. HTML offers the basic building blocks of the page while CSS creates a pleasant looking site that is user friendly. Finally JavaScript offers functionality that does not directly affect the server side of the website. HTML, CSS, and JavaScript are relatively easy languages to pick and implement. Given the team's lack of experience designing web applications, these three languages helped the process by being simple and well documented on the web.

For CSS in particular, the W3 and Bootstrap framework was used to create the look and style of the web pages. Instead of building everything from scratch, it was much simpler to implement what was included in the already existing frameworks. Once again our limited knowledge on CSS made this a must have in the end.

### 5.2 Back-end

PHP was used to handle the server side of the site. When looking into how to talk to the server ASP.NET was also considered, but ultimately PHP was what the team needed. PHP has been around for years and as a result is heavily documented. Everything the team was about to do had already been done at least once and documented on the web. The website did not require a lot of fancy features and PHP is a simple approach that the team needed. PHP comes together with HTML and CSS rather easily making it a good choice when integrating the front-end with the back-end. Finally, PHP was fairly simple to setup and the development environment was able to run for all team members.

Since we decided to use a database we needed a database management system. The team worked with MySQL. Once again, limited knowledge of making a website lead us to choose this over others. One teammate was familiar with MySQL and it made sense to use his knowledge with the program. MySQL also offers everything we needed. It allowed for data to be stored in tables and it allowed the access and manipulation of the data. Everything we needed to do with databases could be done with MySQL.

The team developed using WAMP mostly due to convenience and since we had decided at the beginning that we would us PHP and MySql. The development environment WAMP setup gave the team an Apache server to work with. Since Apache is an open source cross-platform server the team decided to use it. The work that is being done using the server should have no problem being moved to another server in the future.

## 5.3 Google Authentication

Our client requested for only users who are logged in to create or sign up for an event. It was decided that we would take advantage of the Google authentication. Since this system is fully functional and does not require us to directly handle the usernames and passwords the team decided that this would be the best approach. The only piece of information we handle with the Google authentication is the user's name and email. This allows for us to focus on making sure that the site is fully accessible to everyone and acts more like a way to connect. The feature to sign up and make event requires for logged in users more as a precaution to prevent fake events and volunteers.

## 5.4 Security

The web application is much like a social network site in which only registered members should be able to make events or to sign up for them. Instead of developing a whole database for passwords the team took advantage of the Google authentication to handle the passwords and rely on their security with users' passwords.

To avoid malicious users precautions were made in the code to prevent SQL injection. These precautions include the prevention of putting in SQL statements into the text field. This also lead to the decision that the search will come with pre-made tags as to prevent any injection to begin with. This also simplifies the search event feature and the team will be able to focus more time on other parts of the site. In the HTML forms there is a bit of encryption being used to send the information to the next page.

## 5.5 Database vs File System

The website will have to hold a lot of pages for community service events. However, all the pages will have the same text fields. As a result the team has decided to hold all the information in a database and display the information with a dynamic page view. Since we have a constant look to the events there is no reason to take up space by making each individual event page. With all the information of the even put on a database it allows for other parts of the site to access this information and display it in a dynamic view as well. Since all our data all has a similar structure to it, it makes sense to take advantage of a database. The team makes use of the file system to store the template pages of the database constructed pages. The one piece of data we decided not to place in the database was images for the event. This decision was made based on the fact that storing images on a database is a challenge. It was decided that it was more efficient to store the image in the file system and keep a directory to the image in the database.

# 6 Results

## 6.1 Unimplemented Features

### 6.1.1 Emails

While working on our project, we ran into problems with finding a home for the website.
Since the website would require a server to run on, we weren't able to set up an email service for sending out notifications to volunteers and organizers about the events.
Email reminders were not implemented. The primary reason being that our client set it as a lower priority than several other features and we weren't able to dedicate enough time to developing the needed resources. The email notifications would also require another server within the server we are hosting the website on. Since we were having difficulties setting up the host server, we were unable to set this server up as well.

### 6.1.2 Event Completion Status

In their requirements document, our client asked for the ability for hosts to be able to "rate" the people that signed up to volunteer at their event. We wound up finding this to be a lower priority that we simply didn't have time for. Since we've been learning how to use our technologies while we've been using them, we weren't able to have as much time to dedicate to all of the capabilities that our client originally wanted.
All of the administrator roles are still being developed. Some of the statistics are in place, but many of the management features are still being finished.

### 6.1.3 Admin Features

The admin pages do not hold all of the functionalities requested. The admin can look at some statistics. They cannot however delete an event or remove a user from the site.

## 6.2 Testing

### 6.2.1

Our main method of testing was simply coding the website, then running a trial to see if it was handled properly. Any time a new feature was implemented, at least one member of the team would play with it to make sure it worked as expected.

### 6.2.2

On Thursday June 15th, our clients were given the ability to play with our website and report any problems that crop up. This beta testing stage allowed us to find bugs we could not see without extensive use of the site. This also provided feedback on front end issues with text boxes, buttons, and text fields.

## 6.3 Future Improvements

### 6.3.1

Improved administrative privileges: it would be good for our website to have more advanced administrative powers, such as individual user statistics and event viewing statistics. Also allow a delete function that does not totally remove the user or event from the site.

### 6.3.2

Email capabilities. Sending emails as reminders or as notifications.

### 6.3.3

The design of the website can be improved upon with more attention to detail. There are minor changes with the header and footer that could improve the user interface.

### 6.3.4

Improved tags and search functions.  Our tags aren't very varied at the moment.  It'd be nice to expand them to the ability for users to enter any tags that they want.

## 6.4 Lessons Learned

Throughout this project, there are two major concepts that we kept running into and learning from.

### 6.4.1

The first one was simplicity.  The simpler method is typically the better method for creating the functions and capabilities you need to create. For example, not hard-coding in some of the features for pages.

### 6.4.2

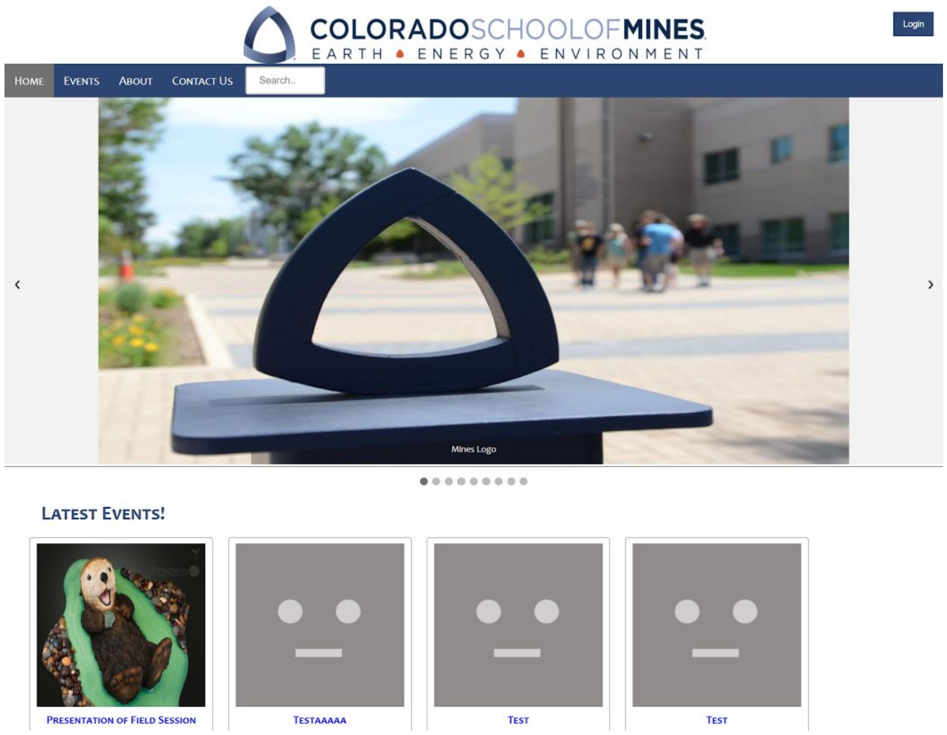
The second was taking time to plan together. We would often form pairs based on what we were going to code that day. Often, however, individually we would focus on one part of the code. This lead to us planning our components on our own and having to redo them to fit together later.  We learned that planning with at least one other person helps to make sure that everything is logical and works together (most of the time).

# Appendices

# A. Documentation

## A.1 Pages

### A.1.1 Home page



The homepage for this website lives within the 'index.php' file in the root directory of the webpage.

This page features the events that were the latest to be uploaded, along with having a navigation bar at the top that leads to the other functions of the webpage. After using the login button to access Google's oAUTH system, administrator users will have access to an extra button reading 'Administration'. This button will only be visible to users marked as administrators within the 'users' table in the 'events' database. For information on how to flag a user as an administrator, please see the "Users" section of this document.

Another navigation bar feature not shown in the image above is the 'My Events' button. When a user is logged in, the button will appear and direct them to the 'My Events' page.

### A.1.2 My Events

The 'My Events' page is designed to show users both what events they have created and what events they have volunteered for. This is simply so that they are able to more easily manage all of the events and information associated with them. From here, users can go to their events to manage different aspects of them. The code for this page is located in the 'events' folder inside of 'MyEventsPage.php'.

### A.1.3 Events

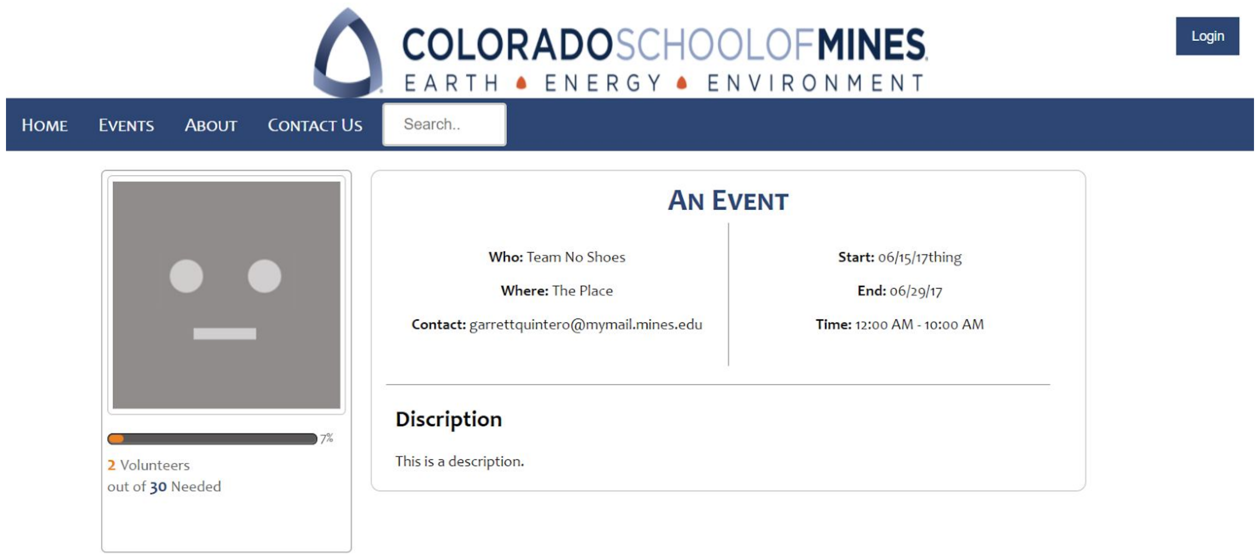
This page is both where the 'Events' button on the navigation bar and the search will lead. This page generates a list of the most recently created events while also creating pages of events. Events that have passed their specified "end time" will automatically be hidden from this page. This page is designed as the "hub" for all of the events that users can see. The code for this page is located in the 'events' folder inside of 'eventPage.php'.

### A.1.4 Make Event

This page is a simple form that adds an entry into the database to represent the event and its details. The file for this page is called 'make_event.php' and passes its information to 'event_making.php' to get the job done. Both of these files are located in the 'events' folder.

### A.1.5 Individual Events

Of all of the pages within this webpage, this may be the most unique. This page generates a unique page tailored to the event id that is passed to it. The links inside of the 'Events' page will automatically be filled in to point to the corred event id. Since the individual pages for the events are dynamically created using the 'TemplateEvent.php' file inside of the 'events' folder, we are able to save storage space on the hosting server by not amassing dozens of very similar files about the events that are being contained. The exception to this rule is held within the 'events/uploads' folder. Images for specific events are held here, all named to reflect the event they are associated with. Here is an example of the layout for an event:



AN EVENT

Who: Team No Shoes
Where: The Place
Contact: garrettquintero@mymail.mines.edu

Start: 06/15/17thing
End: 06/29/17
Time: 12:00 AM - 10:00 AM

Discription

This is a description.

2 Volunteers
out of 30 Needed

### A.1.6 Contact Us / About

This page contains information about the purpose of the website along with information on how to get in touch with coordinators/execs for the Mines Philanthropy Council. Its data can be found under ‘events’ in the ‘aboutUs.php’ file.

### A.1.7 Administration

The administration panel is actually split up into two separate pages. The page you are taken to when you click on the ‘Administration’ link takes you to the ‘Statistics’ page. Here you are able to see some of the statistics about the website and its use. This file is located under the ‘admin’ folder in ‘stats.php’.

The second page is the ‘Volunteer Management’ page. This page contains a list of all of the volunteers that are on the website. This page mainly consists of a large table with a list of volunteers and the ability to add or remove them from an event. The file for this page is also in the ‘admin’ folder and is called ‘volunteer_admin.php’.

## A.2 Useful Variables

### A.2.1 $_SESSION Variables

These variables are accessible to every page that is active. These variables are used to keep track of things like if someone is logged in, who they are, and if they’re an administrator.
The special variables to keep track of are:
$_SESSION[‘login_user’]
$_SESSION[‘access_token’]
$_SESSION[‘email’]
$_SESSION[‘verified’]

### A.2.2 $_GLOBAL Variables

All of our global variables help us when dealing with ‘section.php’ and ‘latestevents.php’. We used the variables to coordinate the look and functionality of these pages.

### A.2.3 $_SERVER Variables

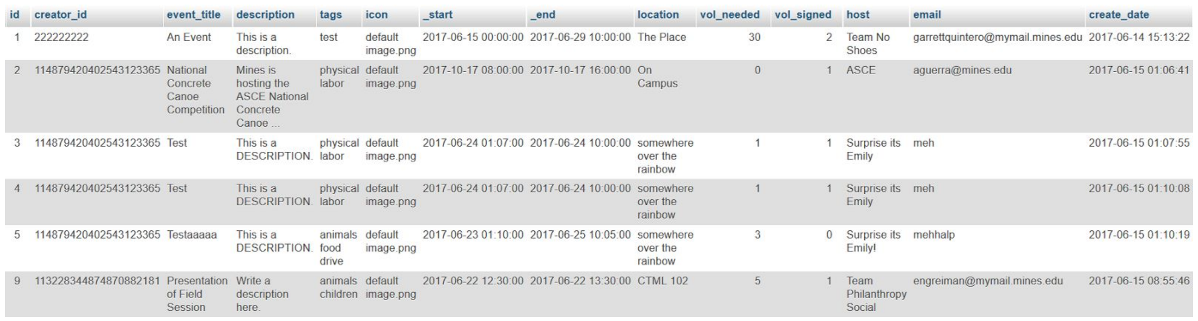
These variables are especially important for coordinating proper URLs from different pages. In several cases, you’ll be sent to the wrong web address if $_SERVER variables are not used.
Important variables to keep track of are:
$_SERVER[‘PHP_SELF’]
$_SERVER[‘REQUEST_METHOD’]
$_SERVER[‘SERVER_NAME’]

## A.3 The Database and Tables

The database for this project is simply called ‘events’.  It has been set up to hold all of the information about the various events submitted to the website, along with keeping track of users and their volunteer sign ups.  All of the details are broken up into the following three tables.

### A.3.1 Events Table

The ‘events’ table contains all of the basic information on an event.  Most of the fields should be rather self-explanatory, so here are the ones that may need explanation.



| id | creator_id | event_title | description | tags | icon | _start | _end | location | vol_needed | vol_signed | host | email | create_date |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 222222222 | An Event | This is a description. | test | default image.png | 2017-06-15 00:00:00 | 2017-06-29 10:00:00 | The Place | 30 | 2 | Team No Shoes | garrettquintero@mymail.mines.edu | 2017-06-14 15:13:22 |
| 2 | 114879420402543123365 | National Concrete Canoe Competition | Mines is hosting the ASCE National Concrete Canoe ... | physical labor | default image.png | 2017-10-17 08:00:00 | 2017-10-17 16:00:00 | On Campus | 0 | 1 | ASCE | aguerra@mines.edu | 2017-06-15 01:06:41 |
| 3 | 114879420402543123365 | Test | This is a DESCRIPTION. | physical labor | default image.png | 2017-06-24 01:07:00 | 2017-06-24 10:00:00 | somewhere over the rainbow | 1 | 1 | Surprise its Emily | meh | 2017-06-15 01:07:55 |
| 4 | 114879420402543123365 | Test | This is a DESCRIPTION. | physical labor | default image.png | 2017-06-24 01:07:00 | 2017-06-24 10:00:00 | somewhere over the rainbow | 1 | 1 | Surprise its Emily | meh | 2017-06-15 01:10:08 |
| 5 | 114879420402543123365 | Testaaaaa | This is a DESCRIPTION. | animals food drive | default image.png | 2017-06-23 01:10:00 | 2017-06-25 10:05:00 | somewhere over the rainbow | 3 | 0 | Surprise its Emily! | mehhalp | 2017-06-15 01:10:19 |
| 9 | 113228344874870882181 | Presentation of Field Session | Write a description here. | animals children | default image.png | 2017-06-22 12:30:00 | 2017-06-22 13:30:00 | CTML 102 | 5 | 1 | Team Philanthropy Social | engreiman@mymail.mines.edu | 2017-06-15 08:55:46 |

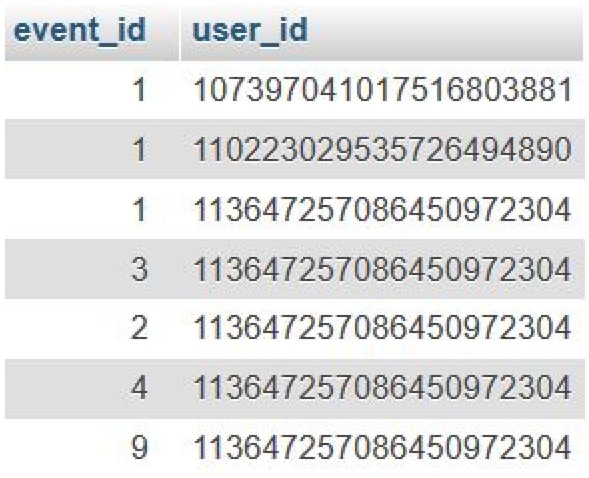
The ‘Id’ column is a self-incrementing column that assigns a unique numerical id to each event submitted to the website database.

The ‘icon’ column contains the path to the image that was uploaded by the user.  The website interprets the text here to use the correct image.

The ‘host’ and ‘creator_id’ columns can be thought of as very similar, the main difference being that the ‘creator_id’ is automatically set to the id of the user currently logged in.  The ‘host’ category is able to be set by the user for different circumstances such as they are creating an event for an organization.

### A.3.2 Volunteers Table

The ‘volunteers’ table is a very simple table.  It is used to keep track of who has signed up for each event as a volunteer by creating an entry that is made of up their ‘user_id’ and the ‘event_id’ for the event they signed up for.

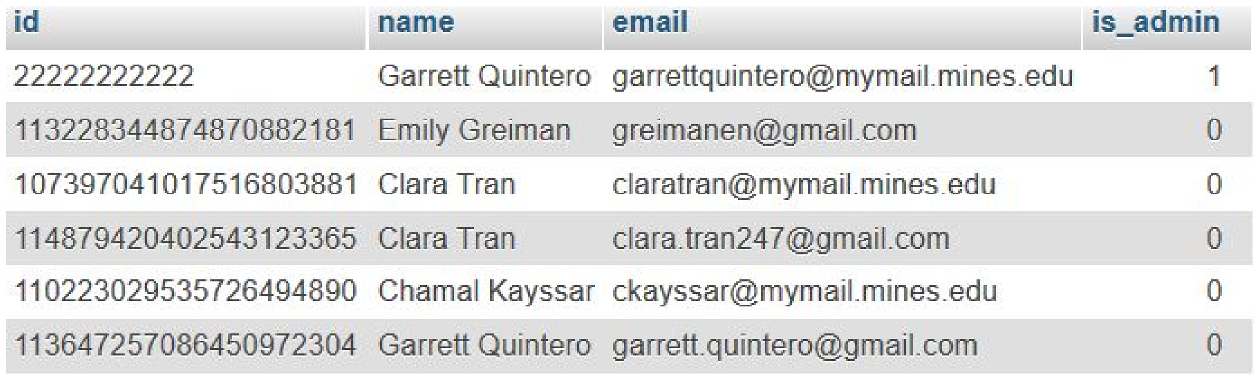


| event_id | user_id |
|---|---|
| 1 | 107397041017516803881 |
| 1 | 110223029535726494890 |
| 1 | 113647257086450972304 |
| 3 | 113647257086450972304 |
| 2 | 113647257086450972304 |
| 4 | 113647257086450972304 |
| 9 | 113647257086450972304 |

### A.3.3 Users Table

The ‘users’ table contains all of the information we collect about users.  The unique columns to be aware of in this table are the ‘id’ column, which keeps track of the unique id that we are given by

Google, and the 'is_admin' column, which we use to indicate if a user has access to the admin pages and privileges of the website.



| id | name | email | is_admin |
|---|---|---|---|
| 22222222222 | Garrett Quintero | garrettquintero@mymail.mines.edu | 1 |
| 113228344874870882181 | Emily Greiman | greimanen@gmail.com | 0 |
| 107397041017516803881 | Clara Tran | claratran@mymail.mines.edu | 0 |
| 114879420402543123365 | Clara Tran | clara.tran247@gmail.com | 0 |
| 110223029535726494890 | Chamal Kayssar | ckayssar@mymail.mines.edu | 0 |
| 113647257086450972304 | Garrett Quintero | garrett.quintero@gmail.com | 0 |

## A.4 Files to know about

### A.4.1 Footer and Header

header.php includes the design layout of the Colorado School of Mines icon and login button that is located at the top of the page. footer.php has the layout of the blue block of credentials located at the bottom of the page.

### A.4.2 Style.css

Style.css is the file that contains all of the common design aspects shared between pages. If there is a recurring theme on the website, its styling is more than likely in here.

### A.4.3 events.sql

events.sql is the file that contains a copy of the database for this webpage. If you are starting up the website for the first time, then you'll need to import this file into a database named 'events'.

### A.4.4 glogin.php

glogin.php is the file that helps to coordinate the process to log into the website. The main aspect of this file to keep in mind is the redirect url. These will more than likely need to be adapted for whatever server is hosting the website.

### A.4.5 confirmPage.php

This page deals with the server update for when a volunteer signs up for an event.
deleteEvent.php
This is the page that is run after the user clicks delete event. It removes the user from
editButton.php
This is the code that affects the edit button. When the button is clicked by the user, they are taken to to this file.

### A.4.6 editFunction.php

Once the user submits the changes from the editButton.php it is sent here. The server deals with all the changes and updates them accordingly.

### A.4.7 leavePage.php

This page deals with the server update when a volunteer withdraws from an event.

### A.4.8 Section.php

For the sake of the code on the eventPage.php, the code for displaying each event was placed into the section.php. This file is called for every event on the event page and filled in dynamically.

### A.4.9 latest_events.php

The homepage holds dynamic display much similar to Section.php on eventPage.php.

## A.5 Google oAUTH

User Access Control ensures that only appropriate users can make important changes to events or other information. This website integrates Open Authentication (OAuth), which provides a common format for user credentials. This allows integration of sign up and sign in with other web services. This project implements sign in with Google accounts.

### A.5.1 How it works

OAuth 2.0, Open Authentication, is an open source authentication service that allows authentication with credentials from other trusted websites. From the OAuth community website: "OAuth 2.0 is the industry-standard protocol for authorization. … OAuth 2.0 focuses on client developer simplicity while providing specific authorization flows for web applications, desktop applications, mobile phones, and living room devices." (OAuth 2.0)
OAuth simplifies both users and web administrators. Users can log in with credentials from other trusted and respected websites, such as Google. Web Administrators are spared the hassle of protecting user credentials.
OAuth uses public key cryptography to allow the transmission of irrefutable evidence of log in through a third party service. The user credential is a message digitally signed by the third-party that names the user and demonstrates sign in within a small time window.

### A.5.2 Set it up

The web server requires several pieces of information from Google to process a log in request. They include: client id, client secret, and redirect URI. All necessary variables are stored in the beginning of glogin/glogin.php in the website directory. Client ID and Client Secret are how Google identifies the website. These codes are obtained by creating an OAuth token in the Administrators Google API Console.

## B. Development Environment

The team used WAMP for development. The package contained all that was needed for the team to work with including Apache and PHP. The team mostly used MyPHPadmin to directly make the database and the team used uniform tables for consistency. All code editing software was handled on either Brackets or Microsoft Expression Web 4. Code was submitted to GitHub to keep up to date with every code change.

When it was time for our clients to begin testing the website to make sure it met their needs, the server was setup to accommodate two websites.  This way, anyone using the ServiceSearch main website would have a consistent experience while we used the ServiceSearch-Development website to update, break, and fix features and bugs.

Our development team is continuing the effort to find a more “permanent” hosting solution for our client, seeing as the current hosting platform is a student’s personal computer.