Orlando Math Circle Design

An event calendar system with a native app-like experience in the browser.

Orlando Math Circle

Math & Creativity | Service & diversity

Group 4

Richard Bent

Luisa Cardona

Samantha Perez

Christopher Polynice

Cody Traywick

Sponsors

Sheina Rodriguez

Matthew Villegas

Table of Contents

1. Executive Summary

STEM has emerged as the leading area of study in today's world. Orlando Math Circle has taken on the challenge to encourage and educate future generations of the importance of mathematics to allow for a better tomorrow. Many children are discouraged by mathematics because they find it to be too difficult, however, this problem usually stems from a lack of motivation and poor teaching methods. OMC seeks to solve this problem by allowing children of all ages and backgrounds, especially minorities and children, to receive resources that may not be available outside of their school. To bring this together, OMC needs to have a fully functional, easy-to-use application to support their needs. Our group will fill in this role to improve and add necessary features to an existing mobile calendar event system that was not in a usable form.

The application shall provide a registration process for volunteers that ensures that they complete all necessary training before granting them the role, allow volunteers to select jobs and be able to switch jobs when necessary, and extensibility for high schoolers to take over in the development process. These requirements will be met with a volunteer dashboard that allows volunteers to see their upcoming jobs and the ability to swap shifts with other volunteers along with thorough documentation allowing students to familiarize themselves with the development process and techniques used.

A main concern from our clients is that the parents using the application may not be native English speakers and have a low technological proficiency while also using outdated devices and software. Our group must take this into consideration and develop to allow for accessibility for all types of users. When dealing with minors and personal identifiable information, we must comply with COPPA laws requiring many security hurdles. Our group is striving to improve the user experience by making it as straightforward as possible while listening to feedback from the clients and students themselves.

2. Project Overview

2.1 Project Description

This project aims to bring forth a solution for aspiring high school students who also share an interest in computer science concepts to engage with the Orlando Math Circle (OMC) in storyboarding math engagement. Currently, the Orlando Math Circle is in the process of developing a web application that will serve to solve the dynamically changing needs of both the instructors and the students, keeping in tandem with the innovative nature of technology. The previous Senior Design group took the first step in drafting a solution for the OMC by meeting with sponsors, coordinating meetings amongst the teams and students, and forming the framework that would serve as the base for all subsequent iterations of the web application.

For the next two semesters, this group will improve upon the application further by ensuring that the involvement of students is closely integrated throughout the development and instructional process, satisfying the newly drafted requirements from the sponsors, and improving upon an already robust solution that the previous team implemented. In accomplishing this, it's important to understand the context and the sponsors of OMC in further detail.

The Orlando Math Circle (OMC) is a non-profit organization who strives to create an inclusive, diverse, and passionate community of mathematicians who apply their skillsets of mathematics in activities outside of school. Geared towards individuals looking to further their knowledge and applications in mathematics, the program aims to help students discover potential career paths, whether they are interested in becoming mathematics educators or are going into a field that utilizes heavy mathematics. While the previous Senior Design group looked to create the framework and ideas that the OMC currently uses now, we are tasked now with improving the quality of the web application and strengthening its use through the collaboration of students currently in the OMC program.

There are methods that we have considered in looking to realize their goal. For starters – which will be expanded on more in the *Requirements* section – involvement of students could be coordinated bi-weekly using Zoom as the medium. Through here, we would discuss ideas and functionalities from the current

application that the students enjoy, functionalities that are more involving and/or difficult to grasp, and features that would need to be improved upon considerably. In this process, we would also facilitate active learning by explaining the efforts and coding changes that would need to be made to satisfy said changes. An additional idea would be to deconstruct the backend framework and think about whether data accessed from the database could be structured more efficiently to ease developmental goals and get additional features for free. Lastly, the frontend system could encapsulate many of the existing features onto a home page that would improve accessibility and streamline use of the application in a more practical way.

Because the requirements and desired features of the application are student-focused, design criteria for the look and appearance of the application are more open than the functionality and utility. With a focus on students for the use of the application, the constraints are based on the students and sponsors, and will probably involve negotiation in developing the fine lines between the needs and potential wants of the application.

2.2 Goals and Objectives

Our main goal of this project is to improve an already existing application to a usable state for OMC. Our clients have expressed multiple times that developing for is a must while also extensibility involving the high schoolers in the process. The application that they are using right now makes everybody's life within OMC, from the clients to students and volunteers, much harder than it needs to be. That application is also closed source and the developers have since moved on from developing for OMC. Obviously, this provides many hurdles for OMC because there are no updates occurring for the application that can provide bug fixes and new features.

An application may seem to perform the intended services required but end up being more of a nuisance to users. This is the case with OMC and our group will work diligently with the clients as well as students and volunteers to ensure that the product satisfies all their needs. Many of the technical solutions to these issues can be found further in the document inside of their respective sections.

Testing the current application was a top priority for our group to learn what features are working and what needs to be fixed. Our experience with starting fresh

on the application, reading through the documentation, and setting up our development environments will aid us in explaining the process to the students and help us work through any technical difficulties that arise during the setup process.

Below are the concise goals and objectives the team has agreed on:

- Improve the existing application for OMC to be self-sufficient
- Include volunteers and high schoolers in the process
- Improve the UX for users (volunteers especially)
- Collaborate with volunteers and users to find solutions to issues
- Build a testing suite to conduct unit testing for the app
- Fix issues with the UI that could potentially cause confusion

2.3 Broader Impacts

The Orlando Math Circle is attempting to create a solution in the realm of mathematics that no other math group has. In creating an application that facilitates learning of mathematical and programming concepts, OMC would have the ability to outsource the application to other mathematical groups that are interested in creating a similar system. While the impacts of the application would be more localized, constant exposure and feedback about the application could have the chance to travel outside Central Florida.

Furthermore, the OMC would experience immediate benefits such as increased attendance and engagement: because the application would present the events and information to the user through the convenience of mobile and web frameworks, students would feel more inclined to attend events, especially with the inclusion of the previous Senior Design group's credit system. Our proposed solution would improve upon the current design that has been storyboarded to bring important information that users would need to stay current on events.

Perhaps one of the biggest impacts the web application would bring to OMC is the inclusion of under-represented students who are from lower-income families. Since OMC is a non-profit, the usage of the application would bring about learning concepts that other corporations would charge a premium for, to students in an online environment at virtually no cost. Since the OMC is also trying to coach students on mathematical concepts applied in the world of computer science,

the application can extend itself and apply the scope from individuals a part of the OMC to any student that wants to explore mathematics and computer science.

2.4 Statements of Motivation

2.4.1 Christopher Polynice

As an aspiring software developer, I've always enjoyed producing solutions that will impact the public in a way that is life changing. Growing up as the middle child and the only male in the family, my mother always said to me that if I wanted to change lives, I should join the workforce as a doctor. While I loved the idea of impacting lives, I wasn't too fond of the healthcare field and the material did not interest me as much. In my high school, it had an Engineering program I was a member of that exposed me to differing fields of Engineering, and the one that stuck out to me the most was Computer Engineering, more specifically, Computer Science.

Throughout my journey in undergrad, I realized that my passion for Computer Science comes from the aftermath: when a software solution is drafted that eases lives of individuals, it alludes back to the goals of my mother, who said that changing lives is best realized through joining the healthcare field. I feel like I've met her in the middle – computer science topics are intensive enough to the point where they provide a challenge for me to rise to and rewarding enough in that I've been a part of technological history where I've changed lives through the logic and solutions I've created. For the OMC, I feel that my passion and drive align with their goals in enriching the lives of students through a software solution and, given that math is one of my favorite subjects, I think that I'd be an asset not only to my team but to the sponsors of OMC.

2.4.2 Richard Bent

Growing up, math and sciences always intrigued me. Solving problems was always interesting and satisfying for me. When I watch the pitch for the Orlando math circle project in class that one day I immediately knew that this is the project I wanted to be on. When I was young, I was a part of programs just like the Orlando math circle which led to me being in the field I am now. The exposure I received when I was young, opened doors to the current career path I am on now. Being able to give back to the community while fulfilling

my senior design project and solving an interesting problem seemed like a dream came true.

What technically motivated me about this project is the tech stack. I look forward to working with Vue since I will increase my frontend development skills. I have experience in React and Angular which are similar frontend frameworks. I have been working as a full stack developer for almost 2 years now, working for 2 fortune 500 companies Leidos and IBM, so getting more experience in front end development interest me. This project will allow me to not only develop my developer skills but also other skills necessary for software developers such as writing documentation and explaining code since we will be working with high schoolers, I will need to explain It to them.

2.4.3 Cody Traywick

I have always been interested in the software side of computers and stumbled across web development/software engineering within the past year. While I always thought I was set on becoming a cybersecurity analyst, software engineering has piqued my interest and made me consider a career in software development. When I heard the pitch from OMC, I was amazed at how they were working with children of all ages and backgrounds to get them interested in math. There is usually a negative connotation behind math, especially in school, but it is great to see that OMC is trying to make math engaging and fun.

I see this project as an opportunity to expand my skillset as not only a web developer but as a person. I understand we will be working with many volunteers and children with similar interests, so this project is a great opportunity to be surrounded by an amazing group of people to develop an application to make their lives easier. At the end of the day, I want to see OMC happy with what we produce and the overall impact that our group makes on the students involved with OMC.

2.4.4 Luisa Cardona

My main motivation for participating in this project was working with a non-profit since it has always been my wish to run my own, so when I saw the presentation by Orlando Math Circle it felt like a great fit. As a Hispanic woman in STEM, I understand the need for representation which is why partnering with OMC was so important for me, I'm excited to provide a better way to connect with people and help the organization engage more children and minorities in STEM. I believe that

both my background and passion for Computer Science will be great tools during this process.

Additionally, I've accumulated some experience in project management and software development while working on both personal and professional projects, which has caused me to develop a sense of commitment towards improving user experience while increasing functionality. I believe Orlando Math Circle's objectives was pretty aligned with this which is why I was drawn towards this project. I'm excited to be part of an engaging, diverse, and dynamic team and mostly to expand my knowledge and skillset as a software developer throughout this process with the hopes of delivering a product that exceeds expectations.

2.4.5 Samantha Perez

Since I spent my entire childhood wanting to become a teacher and loving math, hearing the pitch for this project felt like fate. I took further interest in this project because it reminded me of what initially ignited my passion for computer science. When I was given a project in both middle and high school that had the potential to be utilized or displayed on a website, I would put in the extra time to incorporate it as one, typically through website template sites. This initially sparked my interest in UI/UX design and by discovering HTML/CSS later in high school, I became invested in the world of website development. I am excited to learn many new technologies and expand my skill set in web application development throughout this project.

As a woman in STEM who has navigated her entire educational career without much guidance from others, I was drawn to how Orlando Math Circle is built on providing resources and opportunities to students who may not have been given supplemental material that would stimulate their interests. In fact, I was hesitant to go through with majoring in computer science due to both the worries of facing gender bias/inequality along the way and the fear that I didn't have what it takes to prosper in this field. Girls in STEM should not only be encouraged to pursue their interests but be reminded that they have exactly what it takes to succeed in such a male-dominated industry. I have always loved helping others and working with kids, so I am hoping to gain some valuable experience working with clients and contribute to making STEM more engaging and inclusive for students.

2.5 Legal, Ethical, and Privacy Issues

In constructing a web application that will bring about events and learning of mathematics and computer science to students, there are issues that can arise when dealing with a task such as this one, especially when considering the involvement of minors. To circumvent this, individuals who are looking to join the OMC web application as an adult will have to undergo a background check that will verify that the person has not been subjected of criminal activity.

In implementing this, the OMC can ensure that the application caters and is conducive to a learning environment that aligns with their mission statement and goals. Users that are authorized to use the OMC application will need to adhere to a minor code of conduct that will outline rules and regulations that must be followed. For new users who wish to enroll in the web application, they will also be subjected to an online, one hour training video that will present content and assess the users based on the understanding presented.

The application, on the developmental side, is developed in a way that is compliant with the Children's Online Privacy Protection Act (COPPA). For instance, the application places restrictions on the content and information that students may share, an example being if a student wished to change their Avatar picture. While users on the administrative side can upload and manipulate data piped into the application, students must select from a list of pre-existing avatars and do not have the ability to source images elsewhere.

Customization of profiles and other information is also limited for students to ensure that information is protected and kept private throughout. Regardless of how the application functions, in working with students, conducting workflow in a way where collaboration is encouraged while protecting sensitive information that students may input is of most importance, and should be presented as a notice or modal that will encompass all COPPA compliant regulations for new users, both student and admin, to accept.

3. Specifications and Requirements

This project is a continuation of a previous group's efforts to replace an old, closed source native application and build a responsive web application that OMC can maintain in the future. The main goal of this project is to build upon the existing frameworks and educate members of OMC and their volunteers how to provide extensibility and eventually becoming autonomous.

The sponsors have stated that high schoolers must be included in this process as the requirement with the highest priority. This requires that all members taking part in this project must undergo a background check and fill out additional paperwork, regarding child abuse among other topics, since we are working with minors. Volunteering is a core ideal at OMC, so they want to make the experience for them as smooth as possible. This includes having the functionality for volunteers to be able to be able to communicate with others when they will not be able to complete a shift that they signed up to attend along with the ability to switch "shifts" within the application when this situation occurs. Testing of the existing application, finalizing the PayPal payment methods, and updating account information on an annual basis tops out the primary objectives for this project.

Secondary goals for this application include email notifications for events, secondary payment systems as a redundancy (Square), volunteer awards based on hours, and administrative changes such as a balance on accounts and modification of event times. Reach goals focus on integration of message boards (social media and within the app) and point systems from volunteering to level up to entice students to be more active in the OMC community.

3.1 Operational Requirements

3.1.1 Primary Goals

1. High schoolers must be included in the process of this application.
2. The application shall be deployed to Azure and linked to the OMC non-profit account.
3. The application shall provide a framework for extensibility by high school programmers.
4. The application shall port over existing accounts from the previous application.

5. The application shall allow volunteers to read/virtually sign Volunteer Handbook agreement (.PDF).
6. The application shall provide support for annual membership payments through PayPal at multiple levels to be paid starting January 1st for the parent account.
7. The application shall enforce that parents update student details annually for each school year.
8. The application shall allow membership paying students access to certain events.
9. The application shall allow non-membership access and allow admins to approve on a case-by-case basis.
10. The application shall allow the ability for volunteers to swap “shifts” and events with each other inside the application.
11. The application shall be extensively tested and allow for users to report bugs to be fixed.

3.1.2 Stretch Goals

1. The application shall provide automated email notifications at certain time intervals for event timelines.
2. The application shall provide an alternate payment method (Square).
3. The application shall enable accounts to have a balance/credit to apply towards memberships or other events that require payment.
4. The application shall allow admins to modify prep times along with manually verifying/adding hours for volunteers when necessary.
5. The application shall allow admins to modify event times after an event is added.
6. The application shall keep track of volunteer hours that students accumulate and calculate which awards they have earned based on those hours.
7. The application should keep track of students’ graduation year and change account type once they have graduated.

3.1.3 Homerun Goals

1. The application shall provide a point system that allows students to earn points on their account when completing volunteering sessions, attending classes, and answering challenge questions (see 2).
2. The application shall keep track of donors in the database.

3. The application shall support a messaging system between volunteers upon admin approval.
4. The application shall allow student submissions for questions of the week upon admin approval.
5. The application shall provide a framework for Google class integration.
6. The application shall allow avatar creation tied to a point system upon admin approval.

3.2 Design Requirements

3.2.1 Primary Goals

1. The application shall include a Volunteer dashboard to display the user's hours, upcoming shifts, and standing on the leaderboard.
2. The application shall be thoroughly inspected for visual inconsistencies.
3. The application shall display a social media feed from Facebook posts from the OMC page.
4. The application shall display notifications for time modifications and upcoming events.

3.2.2 Homerun Goals

1. The application shall provide a framework for admins to create math challenge questions, based on the student's age, to be answered by students to gain points.
2. The application shall include a leaderboard based on the point system developed by OMC.

3.3 Document Requirements

1. The documentation from the previous Senior Design team shall be updated to include new features and modifications implemented.
2. A curriculum shall be created including basic exercises, tutorials, and trainings pertaining the application for OMC high schoolers.
3. (Homerun) The application shall be containerized with Docker so it can be easily modified or used for educational purposes in the future.

4. Division of Labor

For the current semester (Spring 2021), we will initially spend time getting accustomed to the current application provided by the previous senior design team and simultaneously familiarize ourselves with the relevant technologies, libraries, and frameworks. While working on our design document, we will tackle many of the more attainable requirements so the app can enter a working state before May as well as work with the high schoolers to design mockups for the volunteer dashboard. By the end of the semester, we plan to have both the design mockups and curriculum plan we will utilize throughout the summer finalized.

In the summer, we plan on implementing our devised solutions to the larger scope requirements and continue to incorporate client and high school volunteer feedback into the application. Much of the summer term will be spent on acceptance testing and fixing any and all issues with the application's UI/UX. The team will work together to continue to contribute to the application's hosted documentation as well as collect resources and create exercises for the high school volunteer curriculum plan as we implement future features. By the end of Senior Design 2, we will have a final documentation deliverable that encompasses every component to the OMC application so future developers can easily pick up where we left off.

4.1 Christopher

For the second iteration of the Orlando Math Circle (OMC) application, one of the features that was most important to the sponsors was the functionality of swapping events and shifts. To understand why this feature is integral to the application, an explanation of the current system is warranted.

Currently, when using the application, the users and/or volunteers that provide services to students can configure them using events. When an event initiates creation, fields need to be filled in such as the title, date, coordinator, and fees. After the event is created, the event is appended to the calendar and attendees that are registered to attend the event get notified of its creation. However, there may be certain instances where an event that is scheduled to be carried out by one may need to be switched to another instructor.

Additionally, if timing constraints appear where the event may need to be rescheduled, the way the system is now, the event would have to be deleted and a new one would need creation with the updated credentials. While the solution accomplishes that need for the users on the OMC side, the workaround as it stands is inefficient and could prove cumbersome further down the line as more users join the OMC and the application outsources further past Orlando.

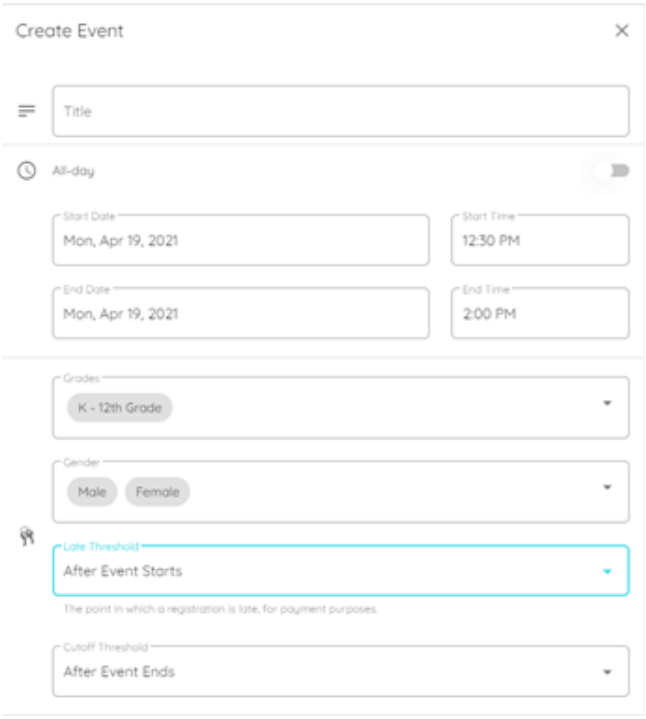


Figure 1. A snapshot of the event creation form, in the current state.

The new solution with event editing and swapping would compartmentalize the solution into one area where it could happen all at once. For the proposed solution, the events could always be placed in a state where the information could always be editable, especially the initiator of the events. Alongside other solutions that were proposed from the group, the solution would work itself into the modified permissions of the application, stating that faculty of OMC and visitors/volunteers of OMC would have read-only access.

As one of the backend developers for the second rendition of the OMC application, I am tasked with creating the logic that will handle the swapping of shifts. Through analyzing the current code, I can figure out ways to improve the flow of data

throughout the backend through refinements of the pipeline and configuration of how the data speaks to one another in the OMC application.

In readying for the task, the current progress I have made towards this ranges from familiarizing myself with NestJS, the current backend server that OMC uses to handle all the data supported to easing the developmental preparation of running the application in developer mode.
While I will not spend too much time on the current steps taken to run the application, it summarizes to configuring the frontend of the application and the backend of the application for them to run asynchronously. By simplifying the process, it can free up more time for ap plication debugging and future-proof the application for the high school students.

4.2 Richard

Richard will be working on the backend of the application as well as supporting frontend developers. On the backend Richard is tasked with integrating the points system. The point system, the point system allows students to earn and accumulate rewards based on volunteer hours. Rewards can be earned through volunteering sessions, attending classes, and answering challenge questions. Admins will be able to assign then points to each student as appropriate. In order to complete this class Richard will need to utilize the modules using nestJS and add the points to the database.

In addition to implementing the point system Richard will be using his full stack development skills to assist on necessary task on the frontend application. Also one of the project's homeroom goals is to integrate google classroom. If the group decides to proceed with the home run goal then Richard will have to task of spear heading that effort and doing research to successfully integrate google classroom.

4.3 Cody

Cody is taking over the lead position for the backend and overall quality assurance of the application. He has been tasked to implement support for PayPal to ensure that transactions are able to be processed for the multi-tier memberships of Orlando Math Circle. This includes setting up a system to allow students the option to upgrade their membership whenever they choose.

Currently, PayPal buttons are integrated into the application but are not fully functional so Cody will need to update the buttons and test their functionality to make sure there are not any issues as he deals with real world money. The sponsors have requested an account balance for users of the app, so he will research for solutions to this request and determine ways to implement a balance and integrate it with PayPal.

With all the new features being added, there will need to be updates to the database to allow for new variables that arise with new implementations of features. Cody will have the responsibility of updating the database and ensuring that these variables provide the correct information when accessed by API calls.
The volunteer dashboard will need an API to be able to access the necessary information to display on the page. Cody will assist in developing this API to give the frontend access to the variables such as volunteer hours, volunteers, and points.

4.4 Luisa

For this project Luisa will be taking over the role of Project Manager and Frontend Developer. Currently, she's managing the project's documentation and storage in OneDrive, updating and organizing Trello cards, and developing a realistic working schedule for the project to assure deliverables are being turned in on time and team continues to make progress to ensure we deliver a product that meets all of the customer's requirements. As the Project Manager, Luisa will also become the primary point of contact with the sponsors and the professors, generate meeting agendas, and gather questions from the team to be addressed with the client.

In terms of Frontend, Luisa will be focusing on creating a high schooler approved dashboard for volunteers to view hours, shifts, and awards. This volunteer dashboard will provide a unique experience to the user where they can see their current status, standing on the volunteer leaderboard, and it will provide a space where they can swap shifts with other volunteers if needed. OMC has also mentioned wanting to add a messaging feature in the future which could be hosted in this area of the application. Therefore, providing a centralized location for the volunteers to perform all activities in the app. Granted that the user experience for this application was developed by the previous Senior Design team, we will be following their design standards to provide the user a seamless and engaging experience.

Additionally, Luisa will be working closely with Samantha to create a curriculum for high schoolers which will contain a series of exercises, tutorials, and trainings on the technologies implemented in the application so they can begin to become familiarized with the application for a smooth transition once OMC takes over the project.

4.5 Samantha

Samantha will be taking the lead on frontend development and involving high schoolers in the development process. As the team will be heavily focusing on the user experience and providing a final product in a fully usable state, much time will be spent on fixing bugs discovered during usability testing and additional minor tweaks to the UI. She will be working with Luisa on the volunteer dashboard and designing ways to implement the features the high schoolers wish to see on the page, along with implementing the functionality behind the features that the dashboard will entail.

Samantha will also construct documentation regarding the application's technologies and framework that is both understandable and engaging for the students. By the end of Senior Design 2, she will ensure that the documentation encapsulates all the necessary components surrounding the application so the students can take the application into their own hands and continue to improve it and add features with ease. Part of that documentation will include the high school volunteer curriculum plan. She will work with the team to devise an effective method for providing resources and exercises for the students to gain familiarity with the relative languages and frameworks.

5. Current Application Status

The previous Senior Design team developed an event calendar system with an app-like experience on a mobile browser with the following features:

- Event management and check-in system
- Email notifications for events and newsletters
- Extended usability through an admin panel
- Open source, documented, and extensible

Although most original requirements were completed, there are fundamental parts that are still missing for the application to be fully functional based on the needs provided by OMC. In this iteration of the project, we will be focusing on addressing any existing bugs and implementing new features we believe will add value to the user experience and increase the functionality of the application. With continuity in mind, all new features and modifications will follow the current design standards to provide the user a seamless experience. Figure 2 has been included below for reference.

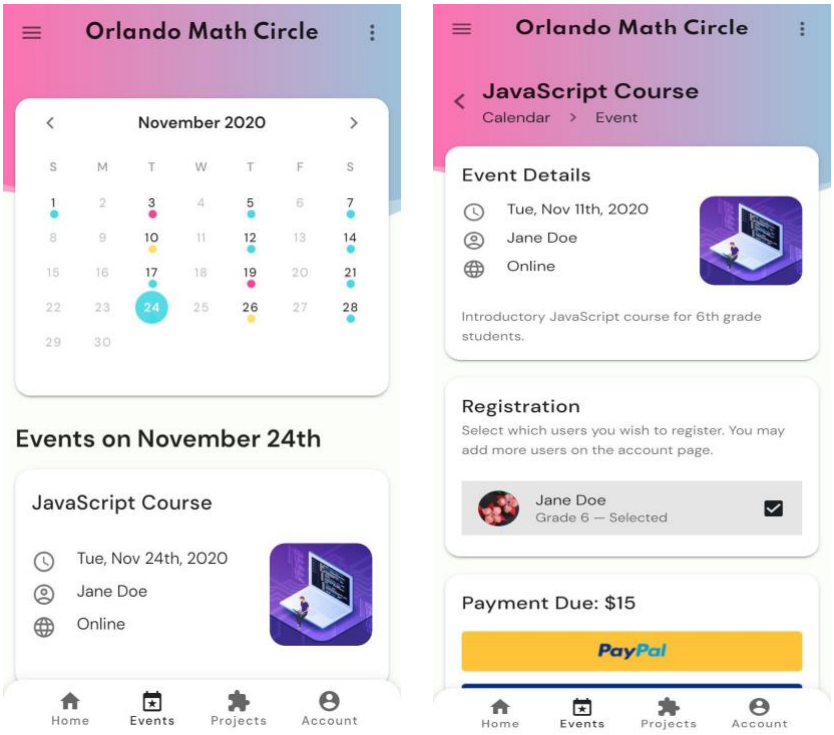


Figure 2. Mobile App UI.

5.1 Infrastructure

The current infrastructure of the application has been broken down in this section, all of these decisions were made by the previous Senior Design team and we will continue to abide by the set standards. This will require a period of familiarization and learning of any unknown technologies by the developers. However, this time has been taken into account in the making of the project schedule and should not impose any delay.

5.1.1 Frontend

- Vue.js through the Nuxt.js framework
- Server-Side Rendered (SSR)
- Single-Page Application (SPA)
- Vuetify component library
- Separation of concerns in single-file components

5.1.2 Backend

- PostgreSQL database with MikroORM
- Node.js and the Nest.js (Express Framework)
- Modular, class-based, and opinionated
- TypeScript and custom third-party interfaces
- Automatic crash recovery with PM2, a NPM library for app process management and restarts

5.1.3 Hosting

- Azure

5.1.4 Networking

- NGINX reverse proxy server
- Intercepts requests and forwards them to the frontend or backend
- Caches and serves static assets
- Contains Let’s Encrypt SSL certificate credentials and renewal flow
- Frontend routed to main domain
- Backend to subdomain api.orlandomathcircle.org

5.2 Deployment

During the transition of the project between teams, the application was deployed to Azure by OMC's in house developer, we were also granted access to the documentation and source code developed by the previous team. During the review process we decided to launch the application to the general public as part of our Senior Design project, so that members of the organization could begin to use it, become familiarized with the app, and provide us feedback of any bugs/issues encountered.

However, during the process we discovered that a couple of elemental features in the application are not ready and need to be addressed prior to launch. After several meetings with the client, as well as a member of the previous Senior Design team, we narrowed down the minimum requirements that would make the application functional for OMC, it is important to note that all other features will be not available given that they will either require further inspections or will be modified during the second phase of the project.

5.2.1 Requirements

The requirements agreed upon for an initial deployment and release to the general public have been grouped in the following categories: account registration, event management, and attendance tracking. It is fundamental to note all three segments will require thorough testing prior to launch.

- ***Account registration:*** the registration page must allow users to sign up without encountering any errors. Registration and Verification emails must be sent out during this process. The landing page, registration page, and log in page, must all work for this requirement to be met.
- ***Event management:*** users must be able to register to all available events. Only admin accounts can create and manage events in the application. Event reminders must be sent via email at registration and at the following times prior to the event: one week, one day, one hour, and 15 minutes before.
- ***Attendance tracking:*** as of right now the application marks all event registered users as 'attended' automatically. This needs to be changed so only users who attend the events get accredited points for attending.

5.3 Known Issues

Our team was given an application that was in an unusable state by the clients, forcing them to rely on the system that was already in place. The clients expressed their difficulties with the current system such as having to keep track of everybody's information manually on spreadsheets and volunteers having a hard time with changing shift times with other volunteers.

Since the application could not be used in the current state, our team decided to fix the minimum amount of the application necessary so that the clients could start using the new application as soon as possible. After several meetings with the sponsors, we came to an agreement that account registration, event management, and tracking attendance of members and volunteers at those events must all be running near perfectly to launch the application to the users.

As expected, when inheriting a codebase from another source, there will be certain bugs and issues that need to be addressed to provide a seamless experience for the users. Our group has tested and will continue to test the application to remedy any issues that arise. We have categorized the encountered bugs in the following sections.

5.3.1 Visual Bugs

As you open the application, you are greeted with a screen that contains the usual Log In or Sign Up buttons, however, that is not apparent right away because you must scroll down to get to those buttons. This concerned our sponsors especially since parents and students may not know that they need to scroll down to be able to log in to the application. Simple modifications can be made to the CSS styling to ensure that the web page does not extend past the edge of the screen.

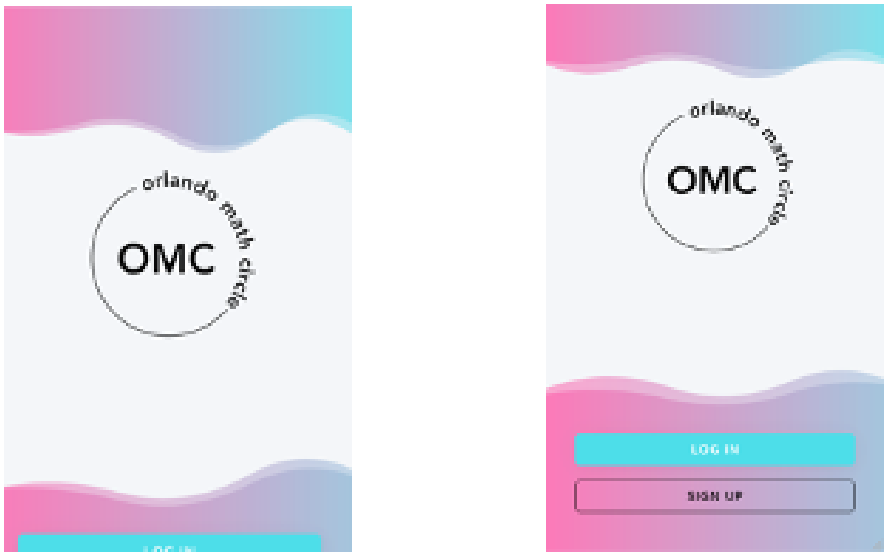


Figure 3. Login Visual Bug.

When opening the admin panel, there can be a visual bug where the graph that shows the number of users added that month is stretched out extending past the field of view. There are also several contrast issues between light and dark mode that need to be addressed. All of these will be fixed through CSS after pinpointing the issues within the source code of the application.

Original Bug

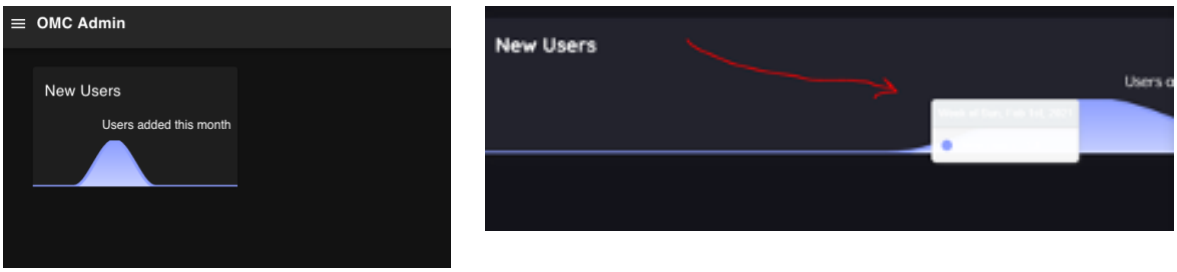


Figure 4. Admin Visual Bug.

Similar to the first visual bug on the sign in page, a user may have to scroll down when choosing an avatar for their profile picture to submit their changes. This may not be apparent at first and was another concern that our sponsors mentioned in

one of our first meetings. Since users of all ages and backgrounds, parents and children, will be using the app we want to make the application as accessible and user friendly as possible. This specific issue can be fixed by creating a static div with the submit and cancel buttons that is always shown no matter where you are on the popup window.

Figure 5. Avatar Selection Bug.

A bug was reported that can trap an administrator inside the admin panel inside the registration page. To remedy this bug, the user must close the application or log out of their account if possible, refreshing the page was not enough. It is unclear if this was a browser or device issue and could be a list of possibilities. More testing is required at this time to attempt to replicate the bug and find a solution.

5.3.2 Logical Bugs

Another major concern from the sponsors is that some children may not have the funds necessary to participate in membership/event fees. This is remedied by a form, such as a reduced lunch form for OMC, granting them reduced to no fees depending on the student's situation. The current method for requesting a reduced

lunch form is by going to the parent's profile and submitting a reduced lunch form with their information sending it to the database for an admin to review and either approve or deny their request.

As of right now, admins can see the request as pending and should be able to approve it by clicking on the edit button, reviewing the information, and clicking approve. However, when trying to access the reduced lunch request, it disappears and cannot be approved by admins. Unless they manually approve it inside the database, which defeats the whole purpose of the application. This is an issue with the frontend and will be fixed by displaying the request and its information when the edit button is clicked.

Before clicking edit, we can see that there are two forms in the pending status in the Figure below. After clicking edit, the form disappears not allowing administrators to accept or deny form.

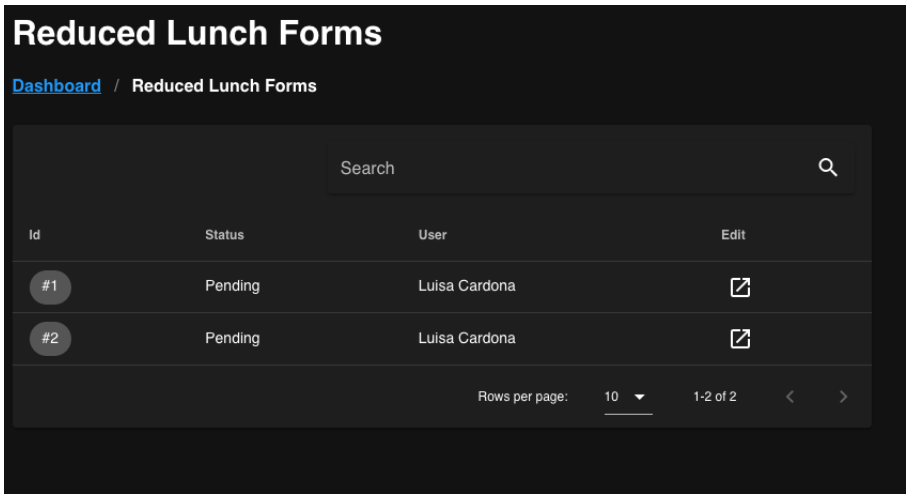


Figure 6. Reduced Lunch Form Bug.

A big appeal of Orlando Math Circle is the nature of a tight knit group of like-minded individuals that have a passion for mathematics and giving back their time to help others. Volunteering is at the heart of the organization and the sponsors have voiced that there should be a straightforward process to become a volunteer. There is a process that volunteers must complete before becoming a volunteer such as signing forms, completing training videos, and even possibly performing a background check if they are teaching at an event. As of right now, the only way

to become a volunteer in the application is to change a user's role to volunteer in the admin panel.

Our team will implement a system for registering as a volunteer through a volunteer dashboard while tracking stages the candidate has completed. If they complete all necessary steps for the level of volunteering that they wish to become, then they will gain the volunteer role allowing them to register for volunteer jobs, track their volunteer hours, and gain rewards for their hours.

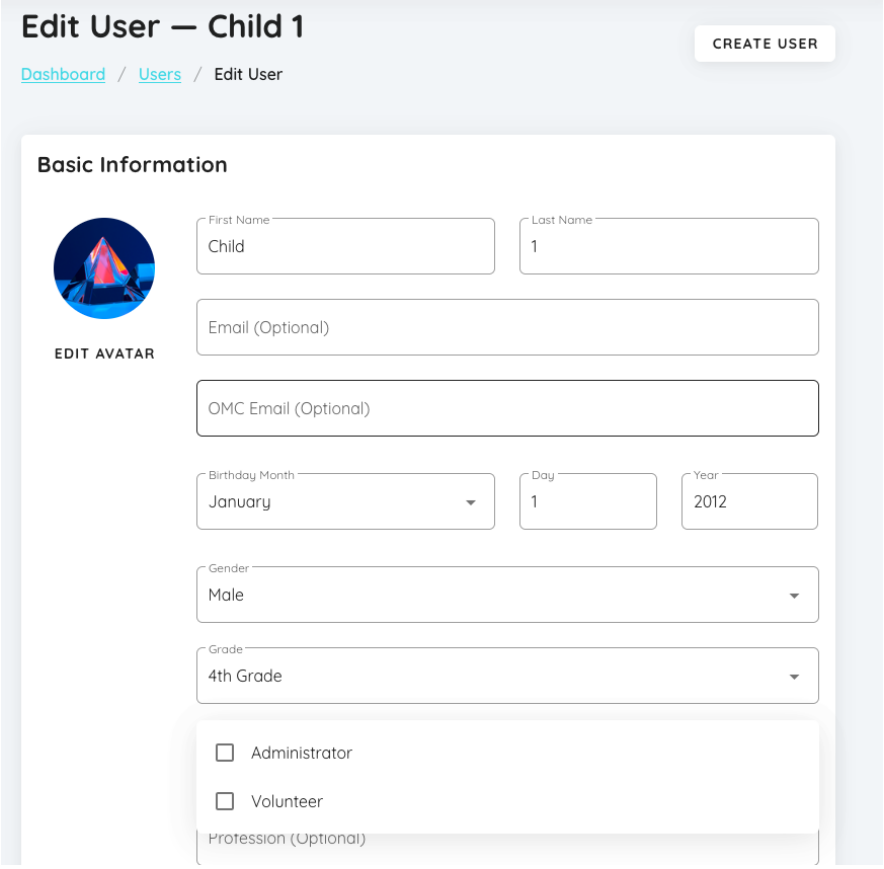


Figure 7. Only way to receive volunteer role.

Students under the age of 18 are required to have their parent sign up user account and then add the students as a dependent "child" account like a Netflix account system. A major security concern from our team is that these accounts do not have their own username and password meaning that they need their parent's username and password to be able to log in. This gives the child full access to their

parent's account and can even allow them to log in to other services that their parents use if they were to use the same username and password for those other services.

We have proposed the following two solutions to address this issue:

1. Require the parents to make a username and password for these accounts (not very secure).
2. Require the students to create a username and password on their first login.

Our team will have to ensure that the existing logic does not require a child to log in to the parent account before being able to access their account.

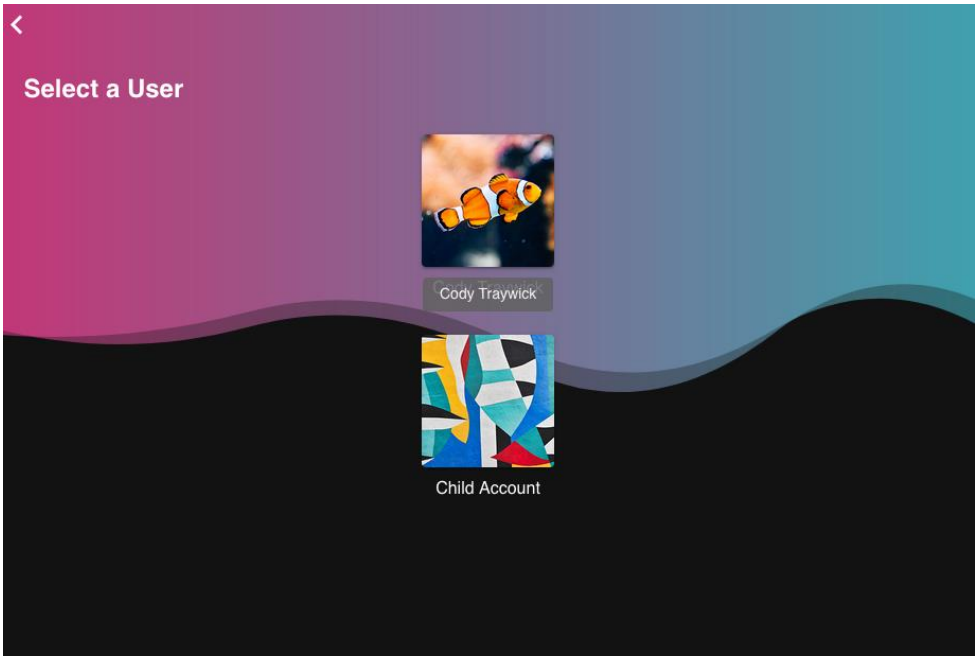


Figure 8. Showing Netflix account selection after parent login.

It is evident that student volunteers will have volunteer hours that came from a different event or organization and that the application will have to account for this if the volunteers want those hours to apply towards the many awards that they can receive from volunteering. The current state handles this situation by allowing users to submit a work request form that takes in user input for the type of work, hours, and a description. The admins should be able to go to the admin panel and either accept or deny these hours, but these hours are automatically approved.

Figure 9. Denying a work request.

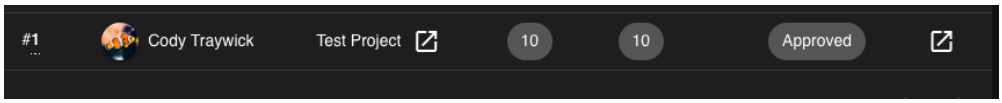


Figure 10. Showing the automatic approval.

Our team has tested it and the only way to “deny” these hours was to edit the request and set it to zero hours manually. This could turn into an issue if a volunteer were to just spam the max number of hours and cheating the system without any approval from an admin.

The current solution is also tedious for admins because instead of either accepting or denying the request, they must manually set the hours for that request to zero. A solution to this issue would be to add a pending state rather than automatically accepting the hours and ensuring that the deny feature works.

When a parent is trying to add children to their account, a form module pops up and they can fill out the form with their child’s information. Once that form is submitted, the child account is created, and the parent can switch to the child’s account for the child to use. A bug inside the current application does not reset the form after submitting so the information from the child that was previously used to create an account is still inside of the form. While not a major issue, this can inconvenience parents that have multiple children since they must delete all the information manually before inputting in their other child’s information.

Refreshing or navigating away from the page resets the form but will be tedious and not straightforward to users. Our team can fix this by resetting the form after submitting it so that it is empty when opened for the second time.

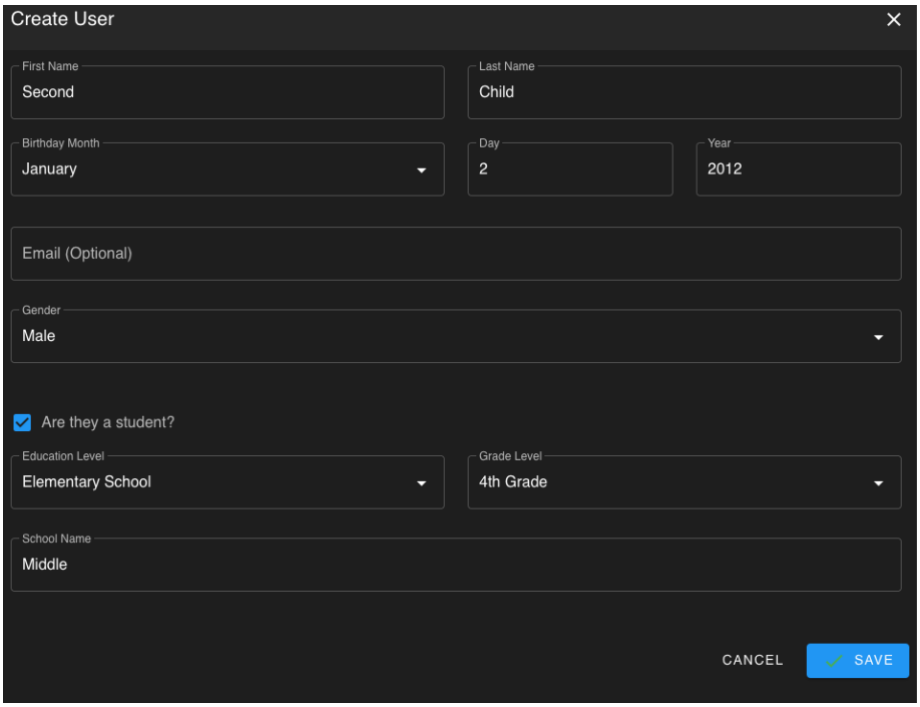


Figure 11. Completed form from previous child when creating a new child account.

6. Potential Solutions

6.1 Brainstorming

In creating a solution that will satisfy both the Orlando Math Circle and the team, outlining a plan with ideas that can improve the web application already present is of most importance.

Christopher Polynice

- For registering with the OMC application, the sponsors stated that there will be a PDF document that new users must sign and verify. This process has been thought of by emailing the document to the user, signing the document, and emailing it back to the respective parties. We could simplify this by presenting a dialog with the information on there and authorize with users checking a box and signing there instead.
- If a parent were to create an account for themselves, they could have the option to create a secondary account for their child, registering as a student, that would automatically sync permissions and restrictions for the student as specified by the parent.
- The system could also dynamically determine the status of the user just by asking for the birth date of the student – through an internal algorithm, we could set it where the status (minor, student, adult, parent) would dynamically change as the age of the user changes.
- For volunteers that wish to help students but do not want to register with the OMC application itself, we can create an option that would allow the user to continue as a guest. As a guest user, there would be restrictions on the type of information the user could see and would only serve to observe or tutor students as they continue through the event.
- To ease user interaction, the backend of the application can handle populating required information throughout the app, from account creation with predefined credentials to speed up use of the application to events that would only need a start time, end time, and date.
- Events that would be created for courses through the OMC can be categorized by content and interest to the user. For example, an event that would be an hour long that would focus on a course in computer science concepts could be placed in a coding category and that category would populate the events and recurring ones.

Richard Bent

- For the application to be extensively tested we can use automated testing such as Jenkins to make testing easier
- Use Vuex for state management so It is easier to debug and change the application state when developing.
- We should give the user account security features such as, change password, two-factor authentication and more.
- Functionality to swap shift can be implemented by having volunteers initially place request, then other volunteers may go in the app and view available swap request and decide if they would like the swap or not

Cody Traywick

- For the ability to swap shifts, we can add a swapFrom (event), a swapTo (event), and a swapWith (person) variable inside the database for each user that can hold the events that they want to swap. If a user's swapFrom and swapTo variables match (mirror) the user that they want to swap with, then those events can be switched, and the application can perform the switch requested.
- If a user tagged as a volunteer hasn't filled out the Volunteer Handbook agreement, we can incorporate the document into the homepage of the volunteer to make them sign the agreement. The database can create a flag variable such as signedVolunteerAgreement that can be set to true or false marking if they have signed the Volunteer Handbook agreement. This could also be used as a check to make sure that volunteers can't register for events if they haven't signed the agreement.
- An email account can be created to send out these automated messages when a specific occasion triggers such as an event being a certain time away. Nodemailer allows a Node.js application to send emails. If certain criteria are met based on the timing desired by the sponsors, then the application can send an email to all participants of the event to notify that the event is coming up in that specific amount of time.
- A graduation date can be added to the database for students that can keep track of the status of the student. Once the student graduates, the backend can update the account type of the student.
- A balance variable can be added to each account in the database that hold a numeric data type (float and double are awful for precision needed for

money). This can keep track of any extra money or credit rewarded to the account that can be used toward payments. This balance can be used to subtract this amount from the transaction amount before processing payment.

- Just like PayPal, OMC would have to register with Square through an eCommerce API application. Once registered we could implement the API calls when payments are required.

Luisa Cardona

- In order to provide a framework for extensibility to high school programmers, any additional features we add to the application created by the previous senior design group need to be developed with modularity in mind for an easier transition.
- Zoom sessions can be set up with the high schoolers during which they can do some shadowing with individual team members so they can become more familiar and involved with the process.
- A TypeScript bootcamp for the team could be very beneficial as a refresher since most of us have not used this programming language in a while.
- Volunteers must be required to read and sign the Volunteer Handbook Agreement (.PDF) as part of the registration process. If their account has been migrated over, then they must read and sign at first log on.
- A Trello board should be implemented among the Senior Design Team, the Sponsors, the Client, and the High School students that will be involved in the process to easily assign activities and track progress.
- The application should include developer documentation so it may be modified or used for educational purposes in the future by the sponsor.
- A volunteer dashboard could be added to the application, this would provide a centralized location where we can host the availability to swap shifts, implement a leaderboard, house all documents that need to be signed to become a volunteer, and in the future a messaging feature could be added so the users can interact with each other.
- If we implement a dashboard such as the one mentioned above, it will provide a hub for volunteers to view all available events, leaderboard standings, upcoming events for which they have registered, as well as any pending actions required to complete their user account. I believe this would be an engaging and interactive space for the volunteers that could increase involvement and participation in events and the application itself.

Samantha Perez

- As most academic school years begin in mid to late August, the application can prompt parent accounts to update student details on August 1st of each year. In the case of students being held back or skipping a grade level, enforcing parents to update details right before the school year would be the most suitable time.
- To further incorporate high schoolers in the process, we can utilize Visual Studio's Live Share feature during occasional meetings to let them learn and give live input as we code. Not every resource we encounter will be included in the project's documentation, so we can create an organized folder on Google Drive of both relevant and valuable but possibly tangential resources we discover along the way to share with the students. In case the high schoolers come across resources on their own, they'd be able to share them with us as well if desired.
- Regarding both the volunteer point system and homerun goal of incorporating a messaging system for the volunteers, we can look into implementing a friend request system within the application. The high schoolers will be able to view their friends' profiles and see their collective hours, badges, and any additional applicable information or statistics about the user. Rather than having to manually search for volunteers each time, users would have a more convenient mechanism of notifying their friends and fellow volunteers about upcoming events or shift openings.
- It's important to remember that the above method would need to comply with COPPA regulations and ensure each student's personal information is not visible to other users.

6.2 Project Block Diagram

A main feature being implemented in version 2.0 of the OMC Web App includes allowing volunteers the ability to swap "shifts" meaning time slots for events that they have signed up to volunteer. Another high priority feature is making sure that parents or students (members of OMC) have updated their information on an annual basis along with paying their yearly dues. The model below shows how these cases are encountered and what steps should be taken after a decision has been made.

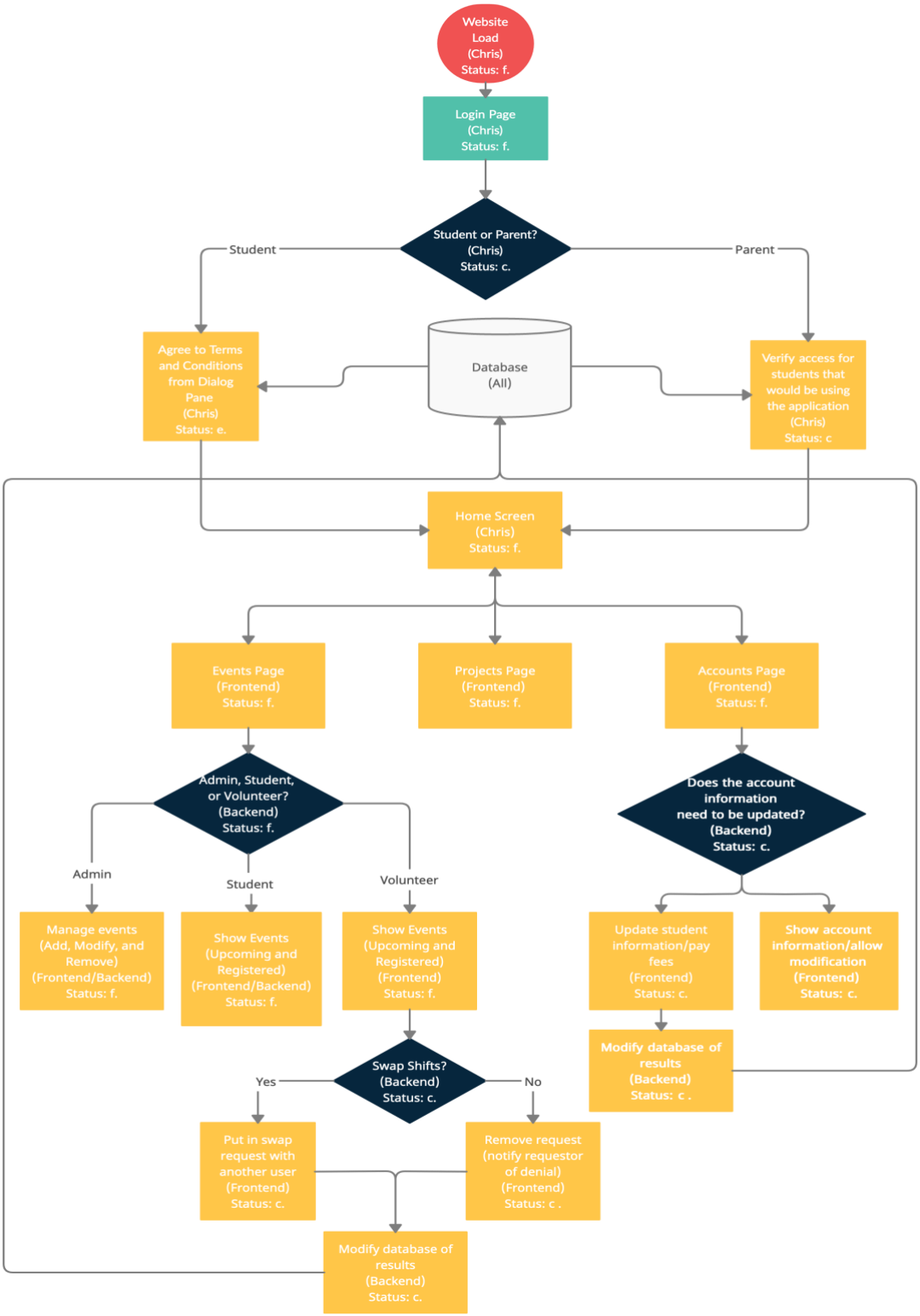


Figure 12. Block Diagram.

Legend: ◆ – decision ■ – action to be performed.

7. Research

7.1 Backend

7.1.1 Docker

Since the major request from our sponsors was the need to incorporate high school students in the process and help teach them how to develop pages to keep the application self-sufficient, the development needs to be a seamless and easy as possible. Most high school students do not have the technical knowledge of finding all of the dependencies for the application, installing them on their machine, and configuring them to be able to run the application. No matter how easy and instructional the documentation is, it will take a lot of focus to follow no matter what skill level of the developer. The last team provided thorough documentation but there were a few mistakes that when followed word for word, the application would not start since it was configured improperly. A high school student would most likely not be able to trace these errors in syntax back to the documentation and become discouraged quickly.

An easy solution to the problem above is to containerize the application especially the initialization process. The application requires four tools just to be able to run and two more that are recommended to use when developing. Navigating to six different websites to find the right downloads for your system can be streamlined down to one or two assuming the developer would need to download Docker to be able to run the Docker image. DataGrip requires you to make an account to download their product since it is not open-source software, so the developer would need to sign up for a student account or purchase the application before using it.

To create a docker image we need to create a Dockerfile first. The following instructions are from the Node.js documentation. [1]
Navigate to your terminal and the folder where your application will reside.

Create the Dockerfile using the following command:

```
touch Dockerfile
```

Open the Dockerfile in a text editor and type the following to support node version 14:

```
From node:14
```

The following code builds the directory to hold the source code of the application inside the Docker image:

```
WORKDIR /usr/src/app
```

Now we need to install the dependencies to be able to run the application. We will need a package.json and package-lock.json to hold the meta data for the application. The following makes sure we generate these and install npm:

```
COPY package*.json ./
RUN npm install
```

To copy the application source code, we will use the following command:

```
COPY ..
```

We want to be able to see the application running, so we will expose the port 8080 to the outside:

```
EXPOSE 8080
```

As always, we need to use node server.js to start the server. To do this in the image we need to use the following command:

```
CMD [ "node", "server.js" ]
```

Now, we need a .dockerignore file to prevent the logs from copying over into the image and plaguing it with information that is not needed within the image. To create the .dockerignore file:

```
touch .dockerignore
```

Inside the dockerignore file type the following:

```
node_modules
npm-debug.log
```

To finally build the docker image all we need to do is navigate to the directory with the Dockerfile and run the following command:

```
Docker build -t <your username>/omc-web-app
```

Obviously, we will need to update the dependencies later to completely match the application. This will be done after more research allows us to pinpoint exactly what can and cannot be done within Docker when creating images.

7.1.2 NodeJS

The backbone of our application is NodeJS. We are using NodeJS on Both frontend and backend of our application. So, what is NodeJS? NodeJS is an open source, cross platform Java run-time environment. It gives us the opportunity to run JavaScript code on the server side rather than on the web client Itself. JavaScript is a popular language used in web development, so It makes It easier for developers who are already familiar with It be able to write server-side code. NodeJS is a popular choice for application development because of JavaScript, speed, asynchronous programing, scalability, no buffering, and developer community. [28]

JavaScript is dynamically typed single-threaded interpreted languages for the Web. You can use JavaScript by going in your browser and opening the developer tools. JavaScript being single threaded means that the code runs line by line, so you can run scripts on the web page without the need of It being compiled like a language like C or Java.

What makes NodeJS so fast is its non-blocking properties, which will be explained later, and the V8 execution engine that It uses. Every web browser needs a JavaScript interpreter, Mozilla's Firefox uses Spider Money. V8 is Google's open-source high performance JavaScript and Web-assembly engine. V8 optimizes code using just in time compilation to convert the code into machine code. Since

every browser uses a different java script interpreter ECMA standardizes JavaScript's features.

The most popular feature of NodeJS is the opportunity for asynchronous programing. Asynchronous programing is when your code does no need to wait for the previous code to finish execution before the next one begins. Especially in web development this can prove to be useful. For example, let's say you are loading a static web page that consist of picture and text. Images are large files thus It may take some time to render on the browser. [30]

When loading the web page and the browser comes across an image It will stop the rest of the page from loading until that image finishes. This is inconvenient from an user experience standpoint. That was an example of sequential programming. NodeJS solves this problem with a non-blocking model. NodeJS has something called the event loop. The event loop is constantly listening to events. It then uses its multiple threads to process the events at the same time.

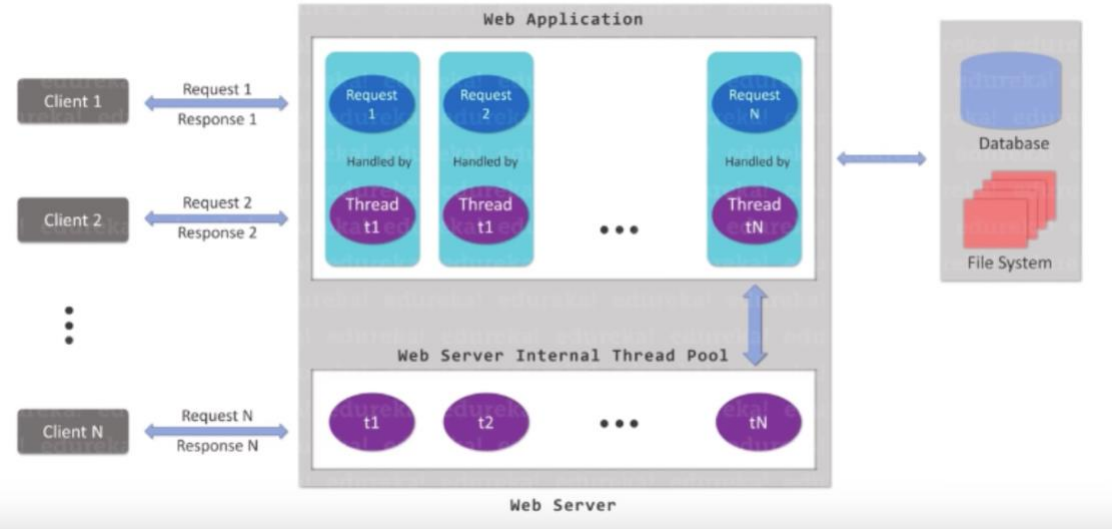


Figure 13. NodeJS's blocking architecture. [27]

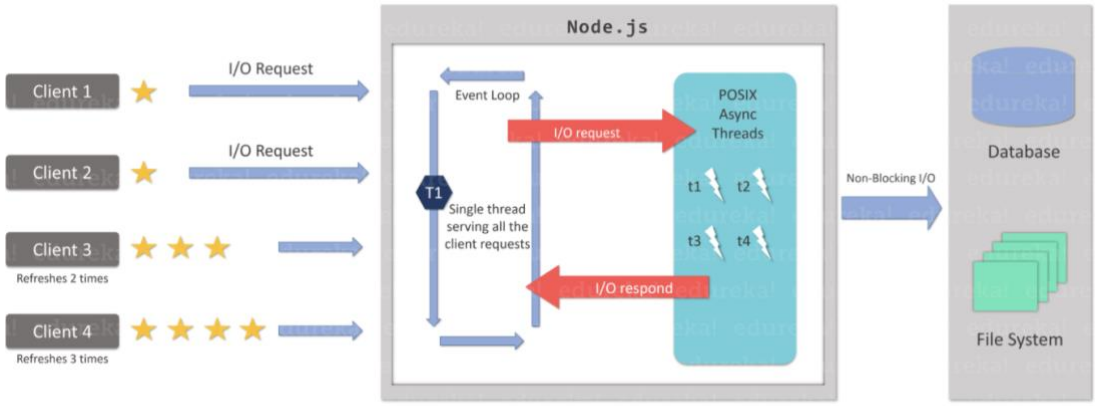


Figure 14. NodeJS's non-blocking architecture. [27]

NodeJS's thriving developer community makes It one of the most popular selections for development. NPM which stands for Node package manager is a collection of modules, which is like libraries you might find with other languages, made from developers which are all open sources. This make It easy to find and install preexisting code from problems that was already solved by open-source developers.

To install npm packages:

```
npm start <package-name>
```

7.1.3 Insomnia

Insomnia is an application that allows you conveniently test your API endpoints. It is similar to Postman which is another popular HTTP REST client. In insomnia you can organize, run, and debug APIs. In insomnia you are able to switch between environments using separate custom variables such as custom headers, bodies and cookies. to make your HTTP request. Postman give the ability to use the response values of other HTTP request as well as add in code snippets from the language of your choice. Insomnia's feature to save HTTP request into a workspace you to easily collaborate between team members when testing and debugging APIs.

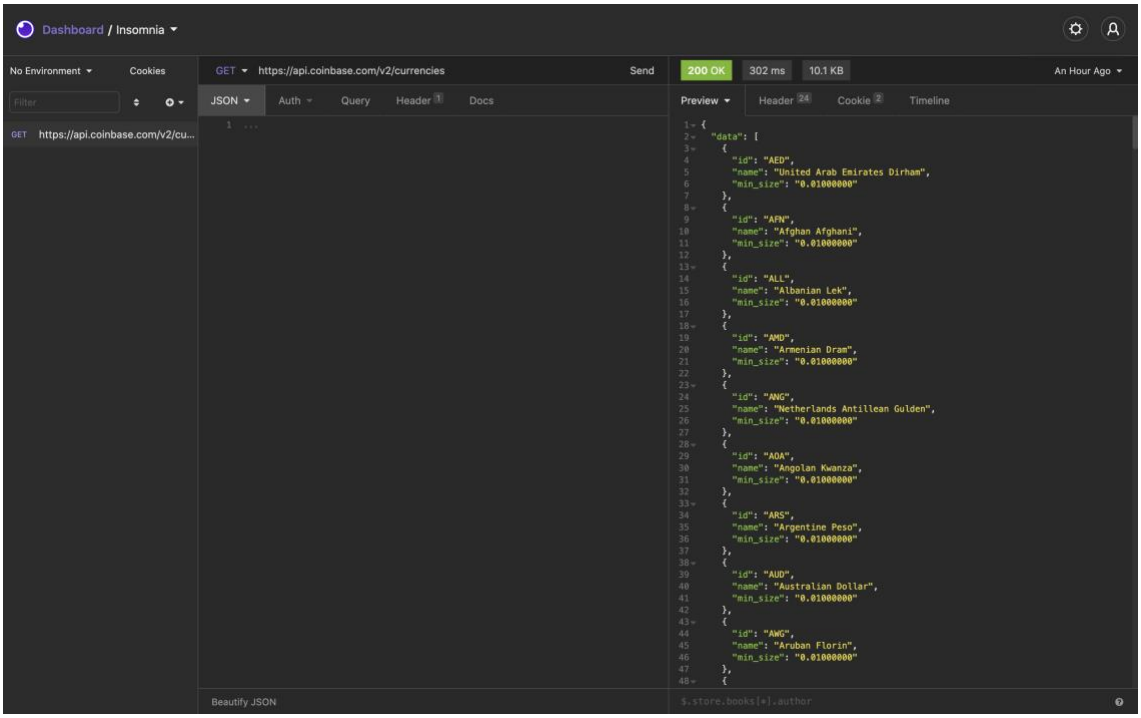


Figure 15. Insomnia dashboard.

7.1.4 DataGrip

DataGrip, a developer database management environment, helps to create, query, and manage databases that are developed either locally, on a cloud, or on a server. The software is powerful in that it supports other SQL-based databases such as MySQL, PostgreSQL, Microsoft SQL Server, Oracle, and more. In using DataGrip, the documentation chunks the information into sections that outline connecting to the data source, querying the data source, viewing the results of the data source, and importing/exporting the data for further manipulation.

7.1.4.1 Connecting to the Data Source

The DataGrip documentation explains that before searching through the database, a connection to the data source must be established [4]. The previous Senior Design group utilized PostgreSQL for database usage and the current Senior Design group intends to keep it. Conveniently, DataGrip supports PostgreSQL and provides steps in integrating it throughout the data source. They are as follows:

- Download the driver files that allow DataGrip to interact with the database. Because they are not integrated with DataGrip upon installation, the size of the file can stay minimized, and drivers can stay up to date.
- Select an authentication method that will be used to allow access into the database.
- Specify the details of the database connection.
- Test connection.

7.1.4.2 Querying the Data Source

As the heart of DataGrip, querying of the data is performed by executing statements that contain results returned to the user. There are various methods that DataGrip allows one to utilize to run and verify command output ranging from file launches that contain commands for execution to designing run configurations that can be launched as well. The methods will be expanded on below as follows:

Running Statements from a File
In file format .sql, DataGrip supports the opening of these files. When launched inside the application, options are available that allow single statements to be executed if clicked or multiple statements to be selected if highlighted.

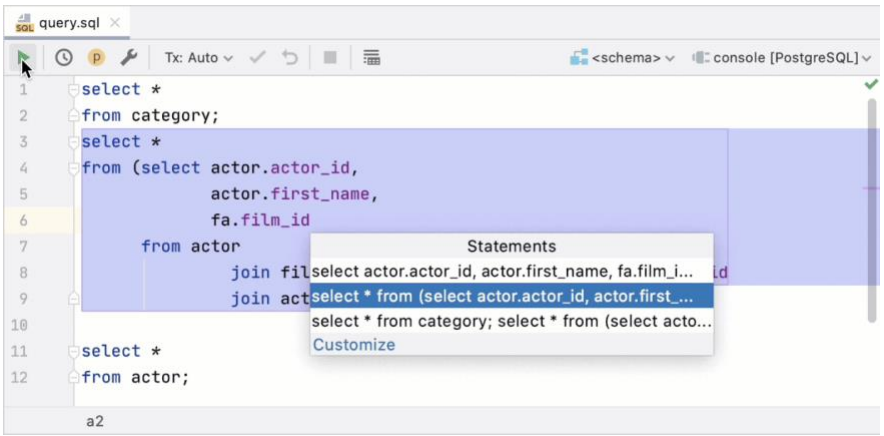


Figure 16. Example of statements that can be ran together when highlighted.

Running Statements from a Hard Drive
If the location of the .sql file is specified to DataGrip, DataGrip then allows one the opportunity to execute the list of commands from the file.

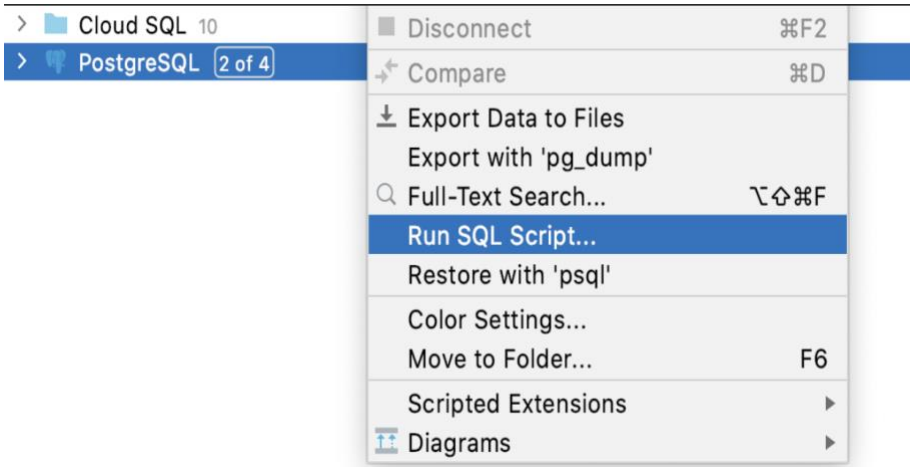


Figure 17. Pictural depiction of launching an SQL Script from the hard drive.

Running an SQL File for Multiple Data Sources

With the file selected, one can use the file and launch it with multiple data sources by changing the run configurations of each of the data sources created.

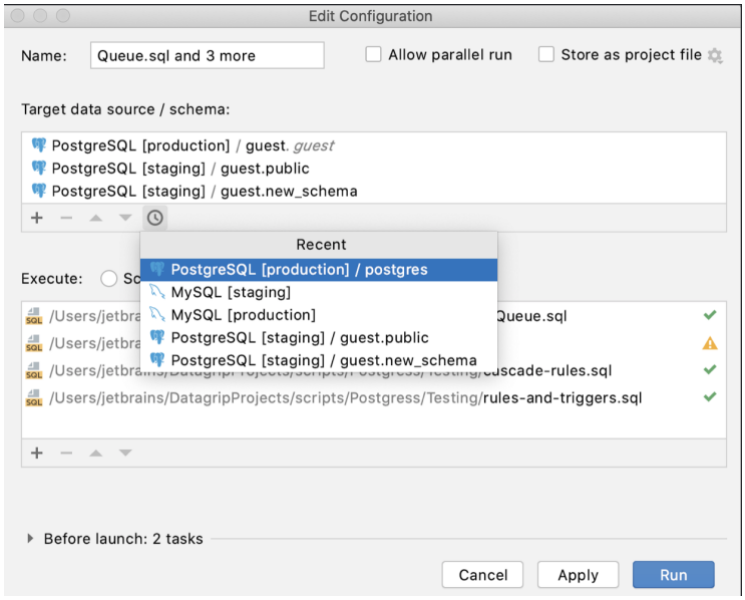


Figure 18. Pictural depiction of changing the configuration of a data source to use a .sql file for command processing.

Using Run Configurations to Launch SQL Files

Through the data source's configurations, upon launch, DataGrip allows the ability to run SQL files. After adding a new configuration and selecting Database Script, the data source is now configured to run commands.

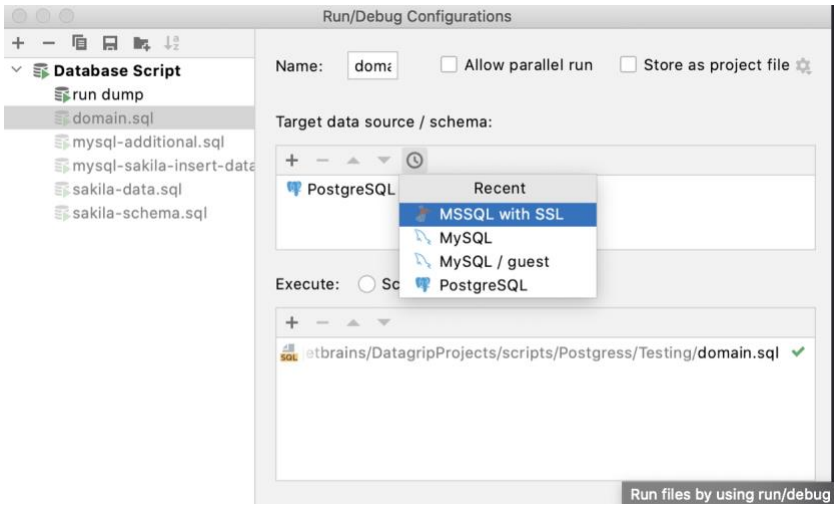


Figure 19. Snapshot of the Run/Debug Configurations dialog, showing the area where the SQL files are connected to the source.

7.1.4.3 Viewing Results of Data

Of course, working with SQL commands and querying data sources would not be as effective without an efficient method to display the data to the user. DataGrip understands this and allows for customization of the outputs of the data for the user. For one, the results of the query are presented in-place shown on the same tab. This behavior can be changed to present the results in another tab for each command ran or to split the results seen in a split-screen like view.

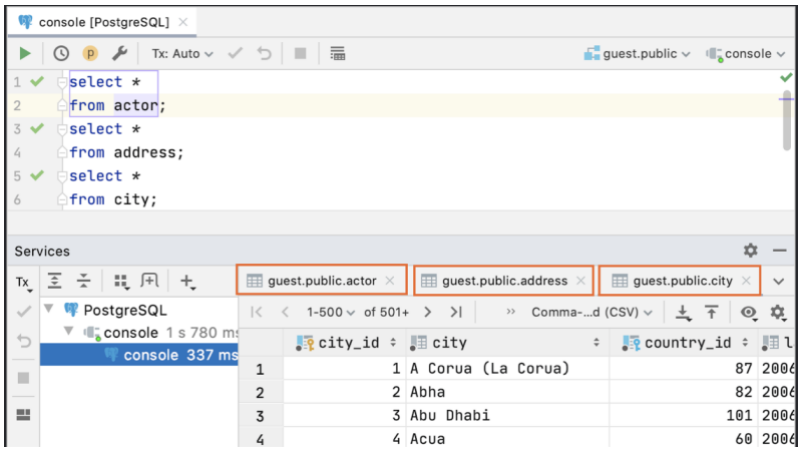


Figure 20. Results of the query presented above shown in new tabs.

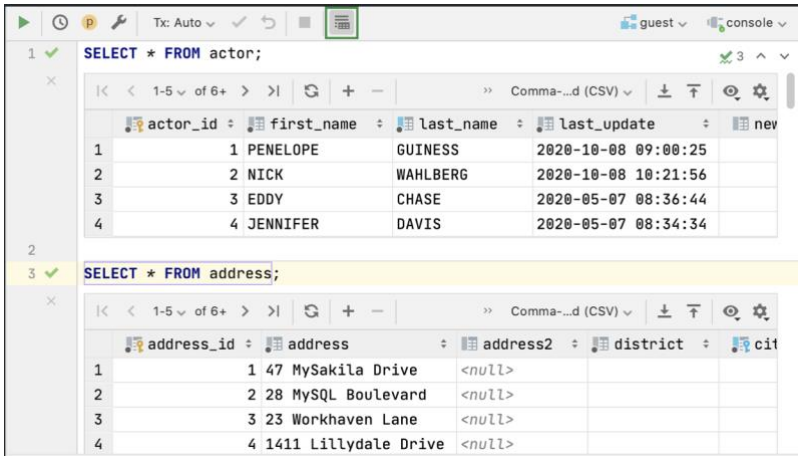


Figure 21. The query results shown in a split-screen view.

If one wishes to denote the results of the query in titles that promote brevity, the user has the ability to change the titles of the tabs to better reflect the purpose of the results. Values in the result set can also be edited, compared, or extracted to further assess the results of the query. Additionally, if one wishes, operations can be performed on the result set such as sorting, removing, or adding to the data.

7.1.4.4 Importing/Exporting SQL Data

As well as supporting importing .sql files, DataGrip also allows the importing and exporting of comma-delimited (CSV) files to the database. DataGrip can export many more file types such as TXT, JSON, XML, and more. The process that is taken when exporting data is as follows: first the data structure table is exported, then the data from the table is exported, and the reason why this format is followed is because it allows for both the table structure and data to be manipulated as needed.

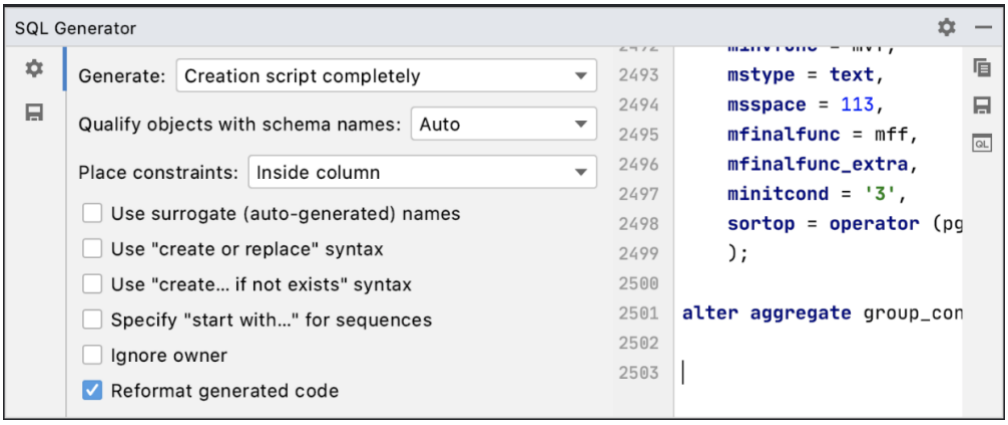


Figure 22. Dialog prompt of a SQL generator, allowing the data to be imported and/or exported.

Current Application Usage

The following client-server model below should outline the minimum specifications that the Orlando Math Circle application will need to abide by to satisfy the requests of the users and volunteers.

Figure 23. Simple Client-Server Model that describes the interaction between the clients (users, volunteers, parents) and the database (fetching the appropriate records to display to the user).

- *User:* As the controller of the application, the user is tasked with performing requests and queries to the application, ranging from searching open events to tasks that may be picked up as a volunteer in the Volunteer Dashboard.
- *Application Database:* Essentially what DataGrip would represent, the hub that handles the user querying and requests, processing an internal set of commands based on permissions that each user contains. For example, the Volunteer Dashboard would be a database that volunteers could perform operations on but minors who are not volunteers would not be able to place queries inside of that page.
- *Application (User List):* The application bridges the connection between the User and the Database, returning the results of the commands and operations placed in a modern and stylish user interface.

While the User and the Database could communicate with each other directly, in theory the Orlando Math Circle application is developed to simplify that task to the user. Through logic and methods crafted to handle the requests and database software as robust as DataGrip that has differing methods to manipulate the data, the application will need a frontend that makes use of these abilities cleanly and efficiently.

7.2 Frontend

7.2.1 UX

UX will be a primary focus throughout the development and usability testing process. Usability and accessibility, which directly correlate to the success of the application, are essential conditions to give users the most intuitive and enjoyable experience. OMC would like to ultimately amplify student and volunteer involvement in the organization through this application. If users have a bad experience working with the app's features, they will be less likely to register for events, sign up for courses, refer family and friends to the organization, etc.

While the students and young adult users may be more familiar with modern technology, many parent users may not be the most computer literate. Maintaining a simple interface that will allow users of all ages to navigate throughout the application without guidance is necessary. As some users may be non-native English speakers or have disabilities, our team will commit to ensure the app is accessible for all users as well.

7.2.1.1 Usability

Usability refers the user-friendliness of a product or website and the measurement of how easy the product is to use in practical application. The aspect of usability is approached from both a design standpoint and by taking the user's point of view into account. The website's features combined with the user's intentions with such features is what determines its level of usability. The three main outcomes of a usable interface are said to be as follows [6]:

1. Upon first contact with the application, it should be easy for the user to gain instant familiarity with the UI and easily navigate throughout, engaging in functional components at ease and relatively quickly.
2. It should be easy for users to complete all practical intentions they have with the application. Whether the user intended to only use one feature or several upon the same visit, the UI should help guide them in completing objectives.
3. Upon subsequent visits, it should be easy for users to recall the user interface and have no trouble continuing usage of the application. Usable interfaces have good design when users can absorb the UI on

first visit and then use the desired functionality of the application just as easily.

The current state of the application has dark mode enabled, a feature that both prolongs battery life and is an increasingly popular choice for application UI aesthetics. It uses a simple and uniform color scheme to enhance the user experience as well. Buttons, forms, and additional functional components are consistent and intuitive across the entire application. The team will continue to uphold these current standards as more features are added in the upcoming iterative development cycle. Figure 24 below displays the tab under the account page where users can fill out important forms that will remain tied to their account.

Currently, the only form available to be sent to administrators is the reduced lunch form. Users can easily upload documents by importing content from their camera roll, taking real-time images or videos of physical forms, or by uploading from their mobile devices' files by clicking the 'Browse…' option. Having several convenient and straightforward options for users to upload content increases usability.

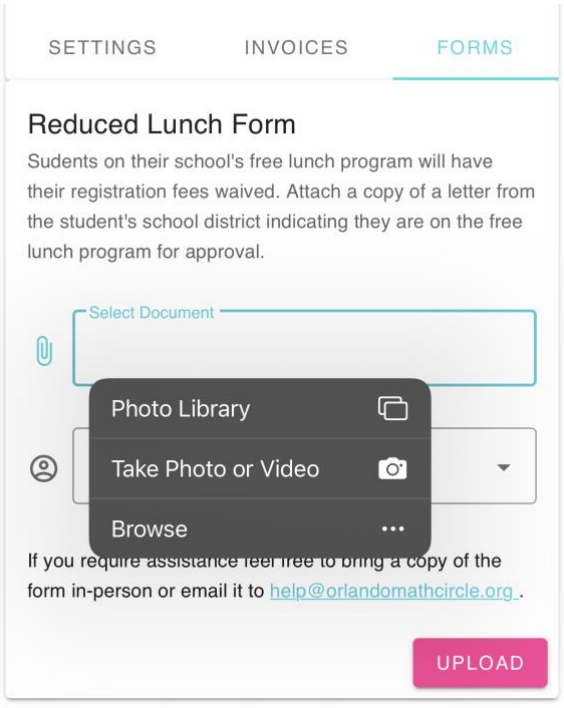


Figure 24. Usability example within application.

7.2.1.2 Accessibility

Accessibility is arguably just as essential of a factor to consider when developing an application as usability is. In fact, the two concepts often overlap with one another. Websites that are accessible ensure that all potential users, including those with disabilities, will be able to easily achieve their objectives within the website. Official guidelines are in place to improve accessibility for people with cognitive, visual, auditory, language, learning, neurological, speech and physical disabilities. Ultimately, accessible design improves the overall user experience and provides equal access and equal opportunity for all users.

Though oftentimes unintentional, the interface language of a website or application can confuse users who do not speak English as their first language. We will carefully analyze any text associated with the application's features to ensure all instructions are simple and standardized across the entire framework. For users with partial sight or color-blindness, it is important to not rely on color as a means of conveying information. We can adhere to this guideline by making sure certain colors do not have meanings assigned to them that can affect the way information is perceived by such users. For example, forms that require fields should not only be labeled with red text but include an icon next to it indicating the field is required.

There are several sets of official accessibility guidelines and standards in existence for UX development. The Web Content Accessibility Guidelines (WCAG) developed by the Web Accessibility Initiative (WAI) of the World Wide Web Consortium (W3C) are a series of step-by-step guidelines to improve web accessibility. On top of extensive technical specifications that categorize guidelines into levels of conformance, the documentation under W3C includes practical user stories that give examples of how users with certain disabilities or cognitive impairments encounter difficulty with non-accessible UI elements, along with how their user experience can be improved.

Each user story provides a list of exactly which other UX techniques for accessibility the user would benefit from as well. The four core principles of WCAG that build the foundation for achieving accessibility claim that content must be Perceivable, Operable, Understandable, and Robust (POUR) [8].

- ***Perceivable*** – In terms of perceivability, content must be presented to the user in ways that can be perceived through multiple senses. Text alternatives, audio descriptions, and sign language interpretations for audio content are common methods to achieving perceivable content. From descriptions for graphical content to labels for UI components, text alternatives for non-text content can be beneficial to visually impaired users who need descriptions read aloud and those who use voice recognition software for functionality to navigate a website. Similarly, captions and descriptions for audio content are essential for users that are hard of hearing. Text should have the capability of being resized up to 400% without any loss of information [7]. Furthermore, audio volume should be adjustable to prevent unnecessary distraction or frustration.

- ***Operable*** – Users must be able to navigate and operate UI components without hassle. For content to be operable it must typically provide functionality from a keyboard. Whatever can be done with a mouse should be capable of being done with a keyboard. Like the previous principle, users with visual or motor impairments may rely on voice recognition software to navigate a website and bring focus to UI elements that must be keyboard operable to provide functionality. Content should be presented to users in a manner where they can read and write, understand instructions, and complete tasks at their own pace. In the event of a session expiring, the website should re-authenticate the user without losing their data and preferably return them to where they were when their session timed out. Developers should be mindful when implementing animations and moving content to prevent the risk of physical reactions such as seizures. Content must also be organized and straightforward to allow for intuitive navigation by the user.

- ***Understandable*** – Content and features that are understandable are easily comprehended by the user. The use of complex language not only makes the user experience more difficult for users with disabilities or cognitive impairments, but for non-native speakers as well. Definitions should be provided for abbreviations, technical terms, unconventional phrases, etc. to support screen readers and comprehension for non-native speakers. Navigation mechanisms should be consistent and occur in the same place on every page. Making a predictable user interface reduces the amount of time the user spends on understanding a website's functionality and navigation components. When implementing forms and other interactive

components, developers should incorporate detailed descriptions and error messages to help the user avoid making mistakes.

- ***Robust*** – For content to be robust, it must be compatible with a wide range of browsers and assistive technologies. Such browsers and technologies typically function and interpret content through standards like HTML, so any non-standard components, such as scripted elements and controls, should exist alongside an alternative standardized version that can be rendered. This allows older browsers and less advanced assistive technologies to process content that would comprise of fewer features but still provide functionality.

While WCAG is a series of accessibility guidelines for web content, Section 508 is a set of standards for accessibility through federal law, published by the US Access Board. Federal government websites, whether used by federal employees or the general public, are required by law to be 508 compliant. Section 508 declares that web content meets accessibility standards when it is provided with the same effectiveness and ease for all users. Most websites for personal and commercial use are not required to be Section 508 compliant but will attempt to meet many of its standards to make content more accessible.

Alongside WCAG exists Authoring Tool Accessibility Guidelines (ATAG) and User Agent Accessibility Guidelines (UAAG). ATAG corresponds with making the authoring tools, the software that enables users to create digital content, themselves accessible so all users have equal opportunity to create web content. The user agents in UAAG refer to web browsers, screen readers, and any other application that renders web content. Together, the three sets of guidelines aim to encourage and assist developers with making their web content accessible for all users. As illustrated in the following figure provided by the W3C, the essential and individual components under each subset of standards work jointly to provide an effective, accessible user experience [9].

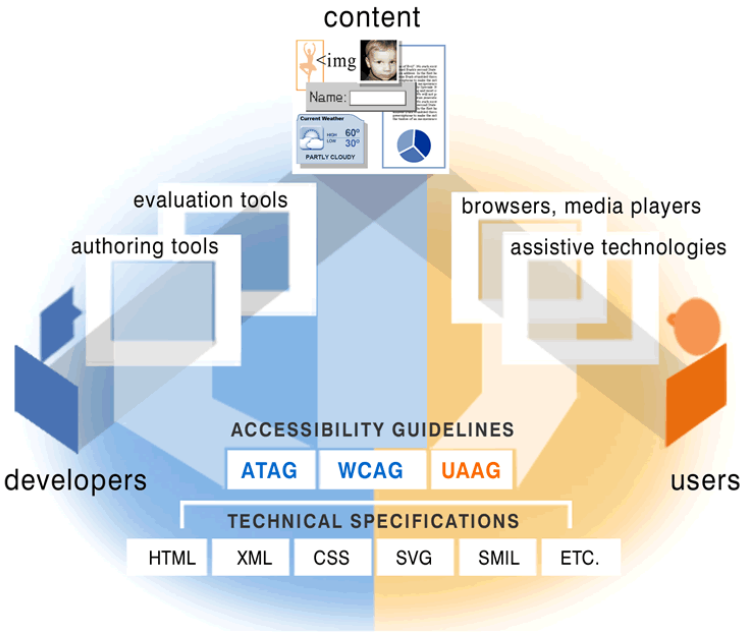


Figure 25. Essential components of web accessibility.

7.2.2 Vue.js

The frontend of the application uses Vue.js, the widely emerging progressive and lightweight JavaScript framework used to build user interfaces and single page applications. As our team does not have any prior experience with the Vue library, we will be doing extensive research to develop and deliver the most suitable UI components for the OMC application. Compared to React and Angular, Vue is considered the easier of the three frontend frameworks for beginners to start out with. Due to its detailed documentation, developers with a basic knowledge of HTML, CSS, and JavaScript will have no trouble working with the library, making it a suitable frontend technology for the high school students to get hands on experience with.

7.2.2.1 Vue Directives

Vue employs a simplistic component-based structure that allows for decomposition of an application into smaller, reusable pieces that can then be used to construct a more complex structure. Within Vue instances are templates that allow for data binding to the DOM and is where the component's HTML is defined. Vue's template syntax consists of text interpolations and directives, which are convenient

Vue-specific attributes that reactively manipulate DOM elements. Table 1 details helpful directives listed under Vue's API documentation [10].

Directive	Details
v-text	Updates an element's textContent as an alternative approach to mustache interpolation (double curly braces).
v-html	Updates an element's innerHTML and renders it as plain HTML instead of through Vue's template compiler. As a result, scoped styles will not apply to content inside this directive in single-file components. It should only be used on trusted content not provided by the user as dynamically rendering arbitrary HTML could lead to XSS attacks.
v-show	Toggles the element's display property, where such element will be rendered and remain in the DOM regardless.
v-if	Conditionally toggle's the element's display property, where such element and any contained components will be destroyed and re-rendered during toggles. It is recommended not to use v-if and v-for together, though v-if does hold higher priority.
v-else	Serves as an "else block" and must immediately follow a v-if or v-else-if element.
v-else-if	Serves as an "else if block" and must immediately follow a v-if element. It can be chained multiple times suitable for conditional statements if desired.
v-for	Loops through data in an array or object. It uses a special alias in expression syntax where an alias, the current element in the iteration, can alternatively be specified for index referencing in arrays and key referencing for objects. The directive works on values that implement the iterable protocol, the opportunity for JavaScript objects to customize their iteration behavior, as well.
v-on	Triggers an event listener. Event types are indicated through method names, inline statements, or modifiers. Such modifiers can be chained and are listed as follows (NOTE: may or may not remove the list of modifiers from this section): • .stop – calls event.stopPropogation(). • .prevent – calls event.preventDefault(). • .capture – adds the event listener in capture mode. • .self – triggers the event handler only if the event was dispatched from the current element

	• .once – triggers the event handler just once • .passive – attaches a DOM event with { passive: true } Due to its frequent usage, the directive uses @ as a convenient shorthand symbol. For example, v-on:click.once and @click.once are equivalent.
v-bind	Dynamically binds one or more attributes or binds a component prop to an expression. Additional value types, such as Arrays and Objects, are supported when binding is used on the class or style attribute. The directive includes a .camel modifier that converts an attribute name in kebab-case into camelCase. Due to its frequent usage, the directive uses : as a convenient shorthand symbol. For example, v-bind:[key] and :[key] are equivalent.
v-model	Creates a two-way data binding between form input, select, and textarea elements or between two components. It includes the following modifiers (NOTE: might remove): • .lazy – listen to change events rather than input • .number – cast valid input string to numbers • .trim – trim input
v-slot	Creates a slot instance that is expected to receive props, where slots in Vue serve as distribution outlets for content. The directive writes the scope of a slot directly onto a component or template tag. Due to its frequent usage, the directive uses # as a convenient shorthand symbol. For example, v-slot:item and #item are equivalent.
v-pre	Skips the compilation process for the current element and all its children. It is used to display raw mustache tags and to speed up compilation by skipping large numbers of nodes with no directives.
v-cloak	Hides the element until the component instance has finished compiling.
v-once	Renders the element and component only once. Often used to optimize update performance, the component and all its children will be treated as static content and skipped over on any re-renders that follow.

Table 1. Vue Directives.

7.2.2.2 Vue Lifecycle

The process of a Vue instance going through a series of initialization steps that creates and updates the DOM is known as the instance's lifecycle [11]. Vue is a powerful framework in binding data, making views reactive, and internally updating everything through its lifecycle process. Vue's lifecycle events allow for the ability to grab HTTP requests from within. Figure 26 displays a diagram of the Vue lifecycle process and the relation between hooks.

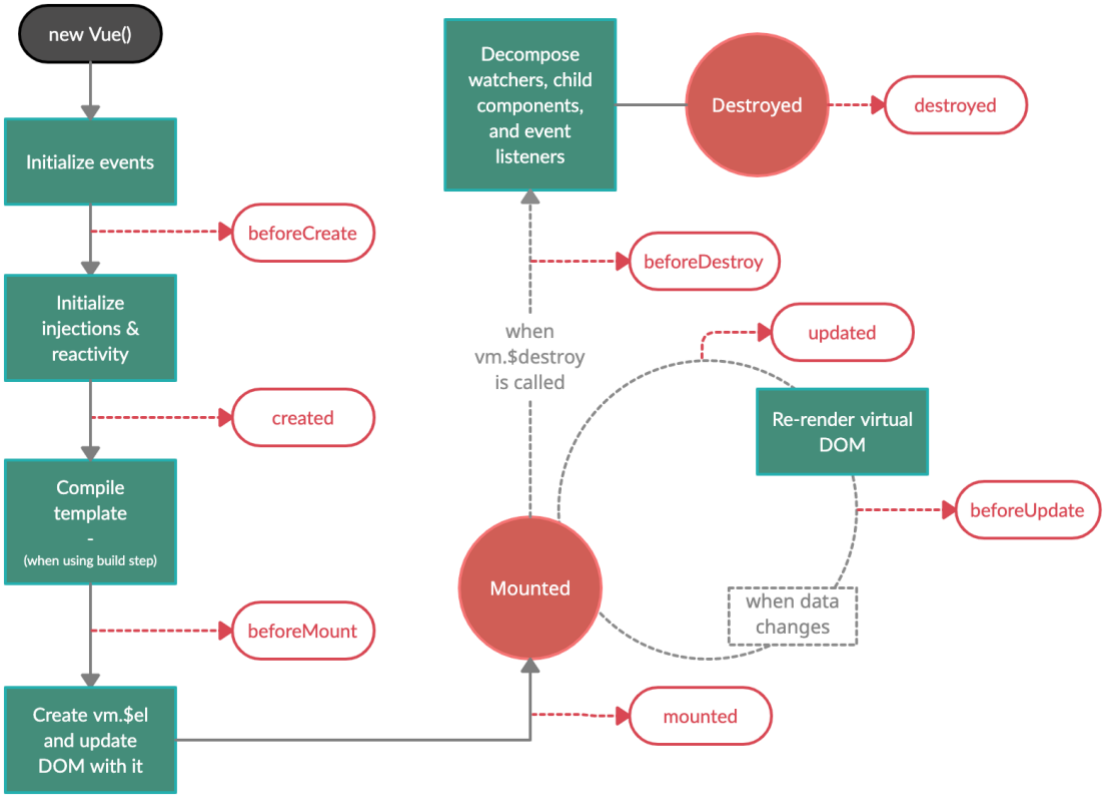


Figure 26. Diagram of the Vue instance lifecycle process.

Throughout the lifecycle process, a series of functions known as lifecycle hooks are run. These hooks are automatically bound to components, allowing for convenient management of a component's state and methods, and are listed in sequential order of execution as follows:

1. Before Create
2. Created
3. Before Mount
4. Mounted
5. Before Update
6. Updated
7. Before Destroy
8. Destroyed

7.2.2.3 Vue Frameworks

Although Vue is the newest of the popular frontend frameworks, there are a collection of frameworks and libraries compatible with Vue to help streamline the development process, provide consistent styling, and boilerplate code within a Vue project. While not required, such frameworks substantially benefit developers by limiting the amount of work needed to be done and making the experience more engaging overall. The variety of frameworks provide flexibility and additional project structure to developers, assisting with improving modularity among code. Common Vue frameworks are listed and described as follows:

- ***Nuxt.js*** is an intuitive application framework based on Vue, Node.js, and Babel.js, the JavaScript transcompiler made to run modern JavaScript on older browsers. It offers convenient features, such as server-side rendering and automatic generated routes, designed to help organize components and allow developers to focus directly on their code by abstracting the complex configuration details that surround server and client code distribution. Furthermore, the framework has Vuex, a state management library practical for scalability, built into it to easily store components and the state of the application in a central location.

- ***Vuetify*** is a UI framework built with Material Design, the design system developed by Google which provides helpful design principles and advanced interactive components to development teams. The framework is built from the ground up and is tailored to deliver concise and semantic components, ultimately simplifying the development process and enhancing the user experience. Vuetify automatically performs tree shaking, the term used for eliminating dead code and, more specifically, is the process of removing unused modules to reduce bundle size and simplify the build process. Vuetify comes with a large support community for developer questions and issues and even releases patches weekly, while most other frameworks patch bugs bi-weekly [12].

- ***BootstrapVue*** is the component library based on Bootstrap, the most popular CSS framework for building highly responsive sites and web applications that simplifies the use and encourages the reusability of UI elements such as forms, tables, buttons, and navbars. BootstrapVue is modular and provides responsive layout through mobile-first design. Mobile-first design is the ideology of starting the design process from the mobile end due to its limitations and bandwidth restrictions to prioritize content and ensure the user experience is universally seamless across every device. As the OMC application is intended to be primarily used on mobile devices and provide a native-like experience, utilizing mobile-first frameworks is essential for the development process. Additionally, BootstrapVue automatically adds markup suitable for accessibility that, when implemented correctly, satisfies WCAG and Section 508 accessibility standards [13]. The framework also includes a Nuxt.js module to allow for easy integration within a Nuxt.js project.

- ***Buefy*** is a UI component library built on top of Vue and Bulma, a modern CSS framework based on Flexbox that limits the need for CSS knowledge by providing an extensive range of built-in responsive and modular features. Buefy's principles are fixated on providing simplicity and keeping everything lightweight [14].

- ***Element UI*** is a component library and UI toolkit for web. Although Element is a Vue 2.0 based library, it offers compatibility with React and Angular and focuses on unified visual styling across a website with its design disciplines centered around consistency, efficiency, and controllability.

- ***Quasar*** is another framework that automatically performs tree shaking. Out of all Vue frameworks, Quasar is the most performance-focused of the bunch. It eliminates the need of additional libraries like Bootstrap and even offers a UMD (Unified Module Definition) that allows developer to incorporate Quasar into an existing project through the addition of a CSS and JS HTML tag [15].

7.2.3 TypeScript

TypeScript is a superset of JavaScript (JavaScript plus extra features) allowing all JavaScript code to run inside of a TypeScript file. [5] This has drawn appeal from many major corporations and small developers alike because existing JavaScript code preserves the runtime behavior of JavaScript. This means that it can run as expected while they transition their code to reap the rewards of TypeScript. The previous team decided to use TypeScript to add a layer of standardization to the application. Standardization is extremely important in a group setting especially when you have five different members writing their own code with their own unique styles. TypeScript is essential in this process because it forces developers to be consistent with their variable types and function parameters.

TypeScript is supported in most code editors such as Visual Studio Code or Atom. Our group will be using Visual Studio Code as recommended by the previous senior design team since it supports TypeScript and has many additional features that allow for increased standardization of code.

Plain JavaScript is prone to a vast array of bugs from passing the wrong values inside of a function to syntax errors. Many errors are not found until runtime execution of the code and could cause entire pages to fail. TypeScript ensures that developers set a type for variables when initializing and then ensures that those types are consistent with the properties that are accessed later in the code. This feature makes TypeScript like typed languages such as Java or C++.

To illustrate the usefulness of TypeScript in catching common bugs the following two figures are shown below. *Figure 27* shows the declaration of an object by using an interface to set the shape of the object. In this example an object with type User must have an id, a first name, and a last name. *Figure 28* shows an error, while declaring a constant user object with a type of User from *Figure 27*, denoted by the red line under the user object declaration. A developer can hover over the error

to see the explanation for the error. We can see that the error shows that the last name property is missing from the declaration and fix the bug immediately without having to test the function and run the application. JavaScript would throw a runtime error bringing the entire page down.

```
interface User {
    id: number;
    first_name: string;
    last_name: string;
}

const user: User = {
    id: 1,
    first_name: "Example",
    last_name: "User",
};
```

Figure 27. Successful Declaration of an Object Using TypeScript Interface.

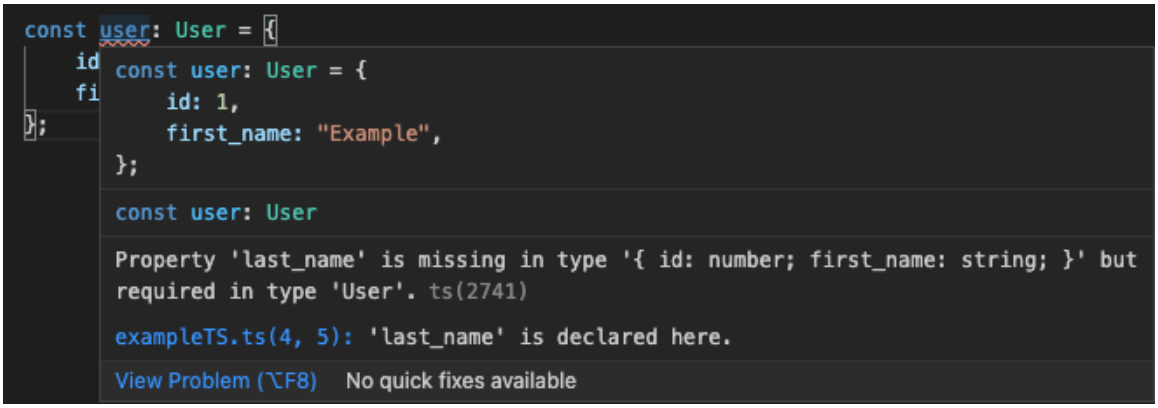


Figure 28. Declaration Error Caught by TypeScript.

7.3 Testing

Explained throughout the document, one of the most important concepts for the Orlando Math Circle is the emphasis placed on involving the high schoolers in the process. Because some are showcasing an interest in programming-related topics and Computer Science itself, the OMC sees involvement of the high schoolers as an integral part of the application. Additionally, when the group ceases production and improvements on the application, it would have been configured in a way

where the high schoolers can continue making revisions and modifications on the application.

Before each sector of testing is explained, the following example below houses a testing diagram for an event which will likely model the testing outline for all the requirements expanded on previously in Section 3.

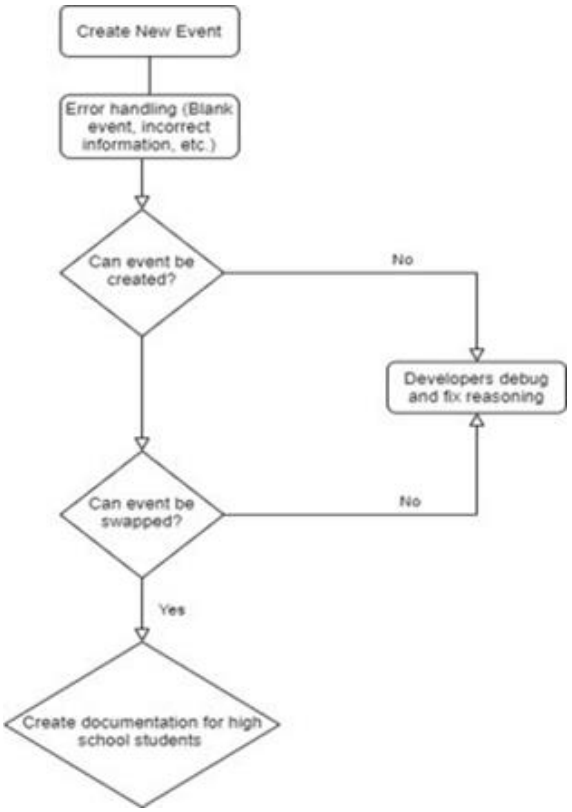


Figure 29. A diagram that illustrates the flow of testing procedures for an event functionality.

Step 1: Initiate Preparing Function (Create New Event)

While the testing diagram represents an example for events, the idea here is that the new feature that is programmed is initiated. Using the idea of swapping events as the feature to test, for an event to be swapped it must first be created, initiating the function that prepares the actual feature to test. It would be here where steps

would be figured out and test procedures would be drafted to verify out if cases could be replicated.

Step 2: Cover Error Handling Cases

After initiating the test, to ensure uniformity throughout the application, ensure that the preceding feature of the application handles errors correctly. Common errors that may be checked for are including, but not limited to:

- *Blank information.* While it is important for the user to be able to have an option to leave text blank (descriptions for events, contacts, etc.), there are instances where data must be filled. For these cases, the correct exceptions should be thrown, or data should be pre-filled with default information.
- *Incorrect information.* If a user is logged in to the OMC web application, any data that is accessed or manipulated by the user must correspond to the active authenticated session. Ensure that access is always granted to the correct user and throw errors if not.
- *Logical information.* Ensure that negative values cannot be entered into fields that require positive fields (time, age), text fields that logically cannot contain numbers do not, and more.
- *Runtime errors.* Dividing by zero, running out of memory allocated, etc. This ensures that all solutions drafted are programmed efficiently.
- *Syntax errors.* Speaks for itself.

Step 3: Test Success of Preceding Function (Can event be created?)

Once the preceding feature finishes testing – in this case, the event is created upon submission – we check if the feature was successful in its intended purpose. While success is variable, it will normally be tiered in different levels. Questions that may be asked and checked for could be verifying if there were errors in the feature, if the feature finished but desired behavior was not achieved, or if the feature generated an exception and did not finish altogether.

If successful, move on to the next step of the test procedure function. In the case of the testing diagram, the next feature would correspond directly to the feature that is looking to be tested, swapping events. On a feature that is deep within the developmental framework, the next step could involve repeating steps 1 to 3 for the next preceding features that set up for the main one to be tested.

If unsuccessful, here is where the testing will cease, for now. The developers who are responsible for programming the feature will come back, take note of the errors or behavior that the test procedure generated, and code further in attempts to fix the issue. In the case of our group, which consist of frontend and backend developers, the event swapping, a backend functionality, would be focused on directly by the backend portion, with help from the frontend developers as they see fit. One tool that may be of use to the developers include, but is not limited to, Visual Studio's debugging functionality that will allow developers to step individually through a line of code, catch the exact location of the place that causes the undesired behavior, and takes note of the location for the developers to fix.

Step 4: Test Intended Function (Can event be swapped?)

Once all the preceding features have been tested and verified for success, the main intended function will be tested. While it will be similar in nature to the first three steps, the difference here is that each step of the feature will need to be verified for success. For example, for a potential solution in creating a plan to swap the event, for instance, the user will need to first click on the event, then click edit, then click swap, the name, etc. The point here is that if the event is clicked, the edit pane needs to be brought up. If that fails, then the developers will step in to rectify the issue until the edit pane is retrieved.

Once successful, the next step to the feature is tested, and the system here is followed until all steps are successful. At any point in this step, an area where the test fails must be fixed by the developers before the rest of the flow can be followed.

Step 5: Create Documentation for High School Students

The final step in the testing procedures, documentation for the high school students gets created about the successful feature for them to read about and reproduce. As reiterated, one of the areas of focus that is most important to the Orlando Math Circle is high school involvement, as the program that they offer provides a scope outside high-level math concepts and applications. By crafting documentation that encompasses the steps taken to create the feature, explain the feature, and show the feature's use, the high schoolers will be able to see clearly how the implementation benefits the OMC web application.

Additionally, storyboarding documentation for high schoolers will hold both developmental sides accountable as understanding for a feature will be known. Understanding why the feature was created, what problems it will resolve, and the code and logic involved in fixing the problem will prove instrumental in the testing process.

7.3.1 Automated Testing

In terms of automating the testing process, it is understood that portions that comprise the steps to test, specifically the last step where documentation would need to be created for high school students, would not be able to be automated. However, due to the inherent nature of the process where it recursively descends into further cases to test, scripting could be used to handle instances where the application would need to manage multiple segments of data at a time.

For example, in the event where multiple events would need to be appended to a calendar, a script could be written that would execute a set of commands for the bot to launch the application, authenticate itself, and create events that could satisfy the concepts seeking testing. While the test cases themselves would need to be written, the scripts would automate the process and take efforts away from the future developers looking to test the feature further.

7.4 Emailing

7.4.1 SendGrid

SendGrid is a cloud-based SMTP provider that acts as an email delivery engine and a leader in email deliverability. Given that it is a cloud service it removes all costs associated with hosting and maintaining an email server, it also manages the technical details of email delivery, like infrastructure scaling, ISP outreach, reputation monitoring, and real-time analytics. SendGrid's SMTP API delivers custom handling instructions for email through a header, X-SMTPAPI, that is inserted into the message. The header is composed of a JSON encoded list of instructions and options for that specific email [22].

Additionally, SendGrid provides multiple guides on how to maximize their resources, email templates, and marketing campaigns that could be very helpful

for the organization. It is important to keep in mind when making a decision on which plan to move forward that every time an email is sent to a recipient, it's deducted as one email credit from the plan. This occurs whether the email is ultimately deferred or sent to spam by the inbox provider [23].

In order to further understand the emailing process, we have included a high-level diagram of the email flowchart in the figure below.

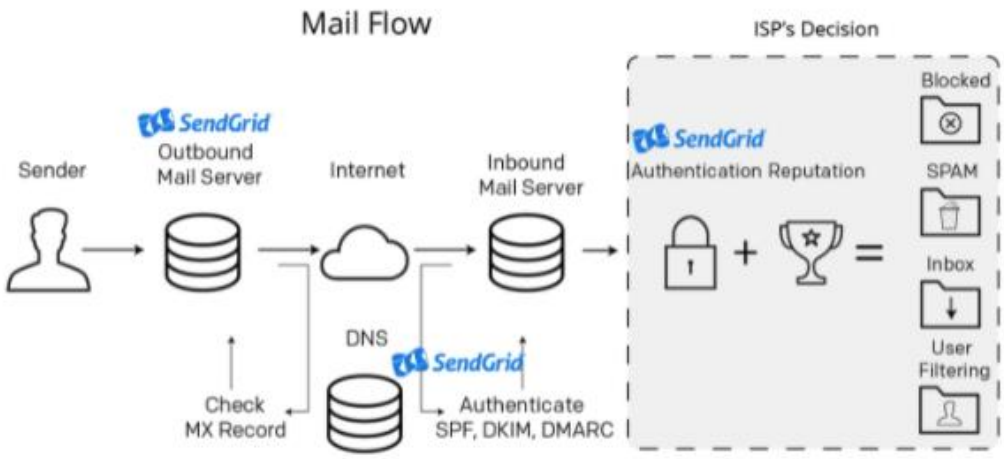


Figure 30. SendGrid's email flowchart.

7.4.1.1 Pricing

SendGrid offers a number of email API plans including custom plans to best match the business' needs. Their base tier includes the following at no cost:

- 100 emails per day
- APIs, SMTP Relay, and Webhooks
- Delivery optimization tools
- Insightful analytics

However, OMC's growing userbase would require an API plan to handle a higher volume of emails. The next tier available starts at $14.96 per month to send up to 40,000 emails per month before overages apply. The Essentials 40K Plan offers everything the free tier does in addition to more features such as guaranteed technical support response times, expert services and personal support [24].

Additionally, OMC can qualify to receive a 25% discount on email solutions after signing up for being a non-profit organization (501c3).

7.4.2 Alternative Providers

Even though we suggest implementing SendGrid as the emailing provider, we have included some alternatives below for reference:

- ***Mailgun:*** this service charges $0.80 cents per every 1,000 emails [18]. It also includes analytics on the percentages of emails missing users and provides insights on how to fix those problems.
- ***Mailjet:*** offers a free tier that includes 200 emails per day [19], with a cap of 6,000 emails per month. However, it does not deliver any emails once the limit is reached, and it adds a watermark to all emails.
- ***SendInBlue:*** offers a free tier with 300 emails per day [20]. However, this emailing service adds a large watermark to all emails.
- ***AWS SES:*** this service charges $0.10 cents per every 1,000 emails [21]. This is provider was suggested by the previous Senior Design team. However, it does not provide analytics, it has an extensive review process, and it has no inbound emails.

7.5 GitHub Organizations

Given that our project in the continuation of a previous one, the client and the Senior Design team have decided to migrate the current application into an Organizational GitHub so that the developer from the previous Senior Design team can continue to push patches he was be working on, as well as provide us access to the existing code. Creating an Organization account will also allow for an easier transition to Orlando Math Circle once the project has been completed.

Organizations are shared accounts where businesses and open-source projects can collaborate across multiple projects at once. Owners and administrators can manage member access to the organization's data and projects with security and administrative features. With an Organization account we will have access to a free option, GitHub Free, with unlimited collaborators on unlimited public repositories with full features and unlimited private repositories with limited features. Orlando Math Circle will also have the option to upgrade to GitHub Team or GitHub Enterprise Cloud for additional features at a later time if necessary [25].

7.5.1 Creating a New Organization

In order to create a new Organization, we will first need access to an existing GitHub account provided by Orlando Math Circle. Once we have access to this account, follow the steps outlined below to create an Organization account:

- Login to the provided GitHub account
- Navigate to the upper-right corner, click on the profile photo, and then click Settings
- In the user settings sidebar, click Organizations
- Click New Organization
- Select the Free option for now
- Follow the on-screen prompts to create the organization
- Add organization members
- Click complete Set up

7.5.2 Transferring a Repository into a New Organization

Once the new Organization has been created, we need to transfer the repository from the previous Senior Design team. The new owner should immediately be able to administer the repository's contents, issues, pull requests, releases, project boards, and settings. However, we must first verify the following prerequisites are met:

- The new owner must have access to the confirmation email. The confirmation email includes instructions for accepting the transfer. If the new owner doesn't accept the transfer within one day, the invitation will expire
- The owner of the repository to be transferred must have permission to create a repository in the target organization
- The Organization account must not have a repository with the same name, or a fork in the same network
- The original owner of the repository is added as a collaborator on the transferred repository. Other collaborators to the transferred repository remain intact
- Private forks can't be transferred

As soon as a repository is transferred to an organization, the organization's default repository permission settings and default membership privileges will apply to the transferred repository. The developer from the previous Senior Design team will follow the steps outlined below to transfer the repository from his user account:

- On GitHub, navigate to the main page of the repository
- Under repository name, click Settings
- Under “Danger Zone”, click Transfer
- Type the name of the Organization account
- Read the warnings about potential loss of features depending on the new owner’s subscription
    - If anything seems incorrect or will cause the loss of information, stop and contact the new Senior Design team
- Type the name of the repository to be transferred, click I understand, transfer this repository
- Verify all files has been transferred correctly

7.6 PayPal

The previous group implemented PayPal Smart Payment Buttons into the current application to handle transactions for event fees. An issue that was preventing payments going through caused an infinite loop of going through the payment process was caused by improper initialization (integers instead of money values) of the transaction value. This has since then been fixed. PayPal supports development accounts that allow for a sandbox environment to test the payment process.

Developers that wish to test PayPal functionality must sign up for a developer account before continuing with testing. This allows our group to test the functionality of the buttons using fake money in different situations such as a PayPal balance, credit card, or bank account to ensure each method is functional. Our group will need to port over this functionality to handle transactions for membership fees that are currently performed manually.

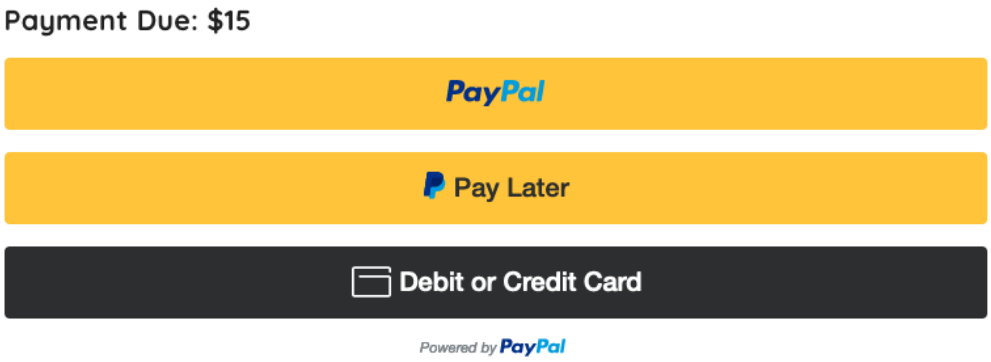


Figure 31. PayPal Smart Buttons inside Application

7.7 Square

Our clients have expressed their need to include more than one type of payment platform within the application to provide redundancy. After thorough research by our team, we have decided that Square seems like the perfect solution. Square was designed as an easy-to-use financial service that caters to any size business handle all of their transactions and manage them in a simplistic manner.

7.7.1 Square Development Environment

Support for Square is almost identical to PayPal. Once a developer creates a Square account there are two dashboards: The Developer Dashboard and Seller Dashboard. The Developer Dashboard allows developers to create applications, access the Sandbox Seller Dashboard, and perform other development tasks. The Seller Dashboard allows sellers and developers to see the transactions taking place and all sorts of information that is useful for accounting and managing a business. The Seller Dashboard also includes a sandbox similar to that of PayPal that allows for developers to test payments with fake money to ensure that the transactions and payment process are running as smooth as possible.

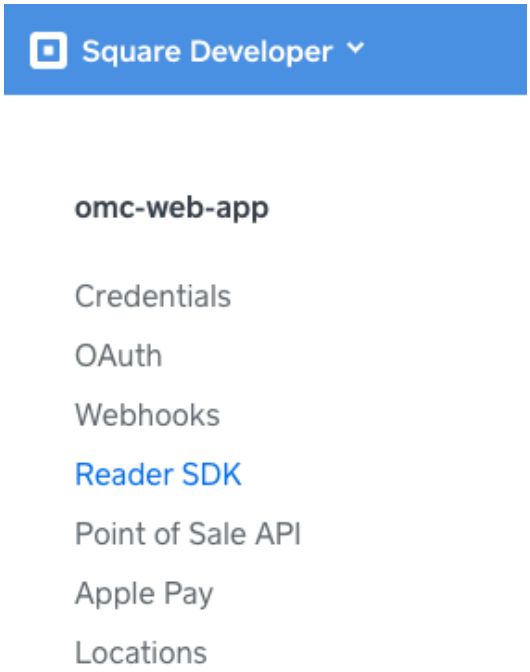


Figure 32. Developer Dashboard

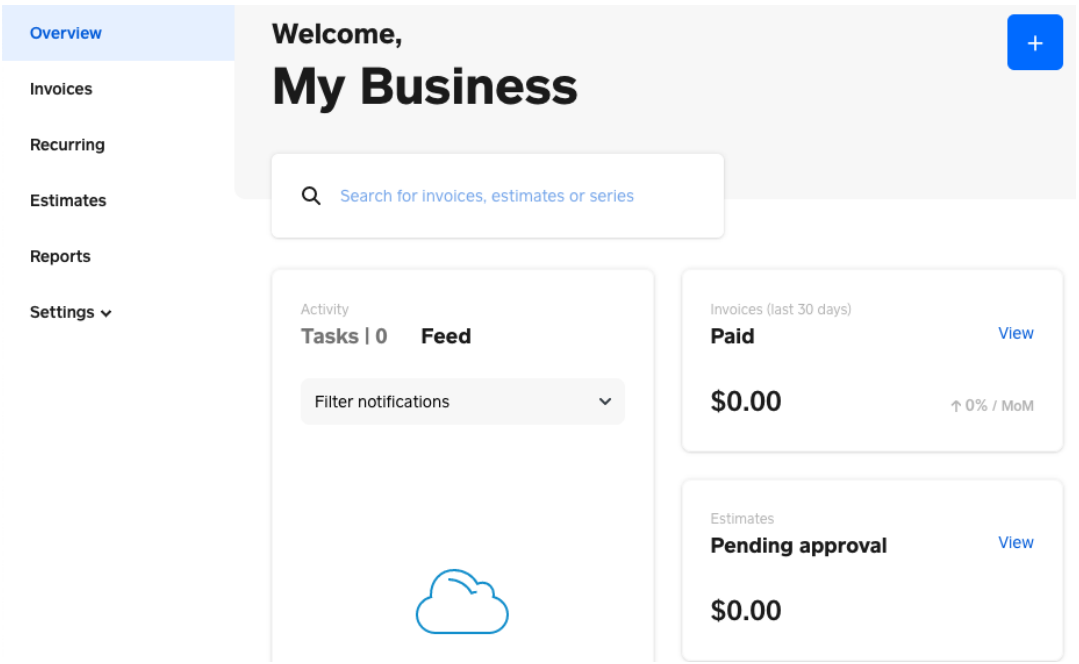


Figure 33. Sandbox Seller Dashboard.

Developers must create an application and connect their web application to be able to process payments and manage Square features. In doing so, the credentials for production and sandbox environments are provided to link to the application. This also allows for OAuth to be managed and initialized. If OMC would like Apple Pay to be supported, there is an option within the Developer Dashboard to configure Apple Pay.

If OMC decided to use Square as a point-of-sale service (POS) for other types of purchases other than memberships and events handled by the application, they could earn rewards such as the ability to process a certain amount of payments with no processing fees such as the 2.9% fee associated with each purchase. There are other rewards that appear periodically such as a discount on Square accessories and equipment like their card/chip readers and stands. Square could ease the financial burdens from OMC by maintaining it all in one place.

7.7.2 Handling Transactions with Square

Similar to PayPal, there are many libraries to choose from to implement Square and all of them have their advantages and disadvantages. The SqPaymentForm library creates an encrypted single-use token, nonce, from the user input data into the payment form on the web page. The charge to the method of payment happens later on the server side. Square has an SDK for Node.js that you must install before having access to the tools within the kit. This provides seamless support to call the API with predetermined functions to ensure that all transactions contain the necessary data before being processed. From there, the Square Payments API is called which will then charge the nonce and complete the payment. Rather than attempting to create a payment form by ourselves, we will be going with the most straightforward and easy to follow process as our clients suggested.

A Square-hosted checkout page is the best option that our group found. This allows the process of creating the order through the app by setting the payment value, name, buyer, etc. inside of the app and then sending that information to the Square Checkout API which uses a prebuilt UI to verify the purchase details and completing payments. It starts on the OMC server and the checkout request is sent to Square. The prebuilt checkout page is hosted by Square until the buyer provides a valid payment and the transaction is processed.

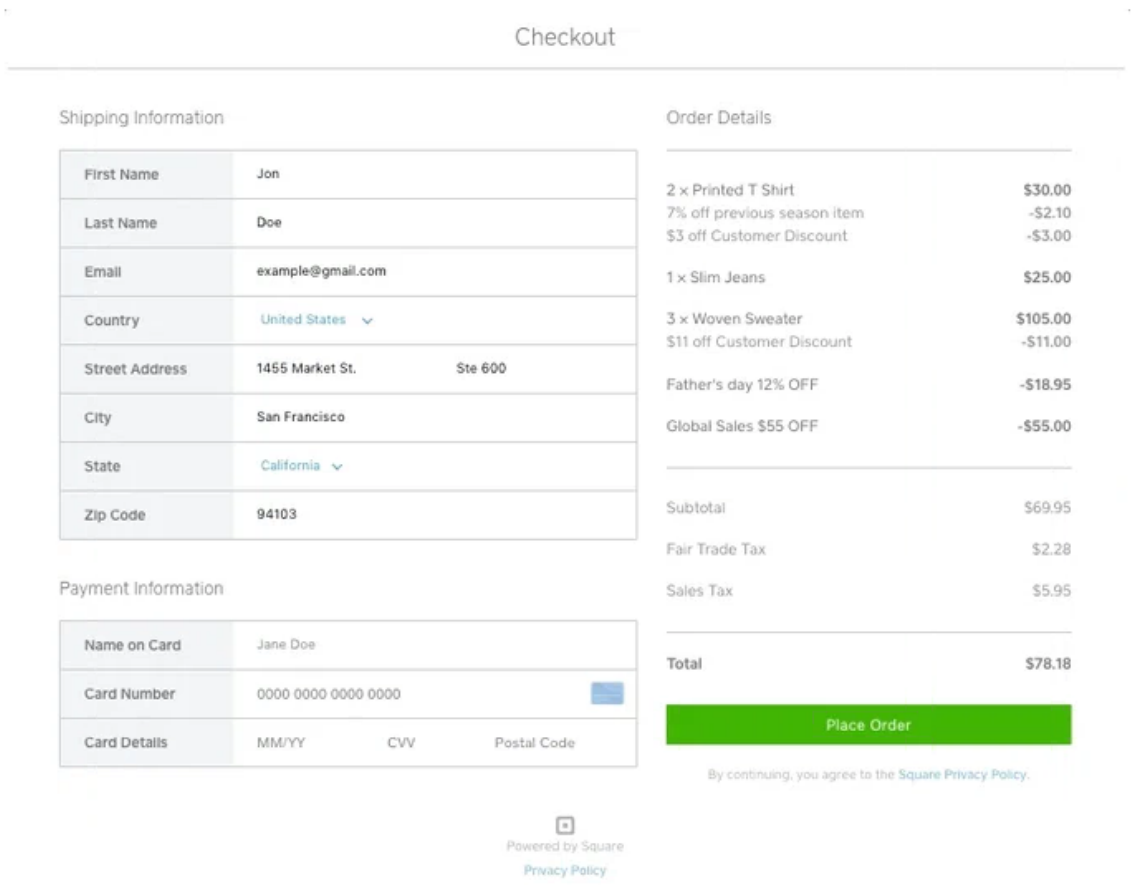


Figure 34. Sample Square Checkout Page [2].

If a payment fails due to an issue such as declined card, Square will emit an error stating the error and will reset the payment process. If any other issues arise during a transaction such as a payment dispute or chargeback, OMC will have to log in to their Square account to handle it manually.

8. System Design

The previous team decided to use NGINX which is a reverse proxy server to handle requests from the client to the API and main domain to reveal the correct information along with SSL/TLS.

- The frontend uses Vue.js and is a single-page application conforming to the Nuxt.js framework. The app will be able to support swapping volunteer shifts, allow for an alternative method of payment, Square, to be used concurrently to PayPal, and ease the process of onboarding volunteers within the app. A volunteer dashboard will be built to allow for volunteers to have a central hub of necessary information consisting of volunteer hours and upcoming events, a leaderboard of each volunteer's hours, and the option to swap shifts.
- The backend uses the NestJS framework for Express.js that communicates with a PostgreSQL database.
- The application will be hosted on a virtual machine within Microsoft Azure.

8.1 Environment Variables

Environment variables will help this application connect with the frontend and backend features. The previous team has provided extensive documentation on the dependencies of the application and the interaction that each environment variable has with the application. There are measures in place to validate if these variables are configured correctly and the application will throw errors if a dependency is not met. Two .env files are present, one in the frontend directory and one in the backend directory. The frontend .env file can be left blank within development mode while the backend .env file must be filled in correctly.

The backend environment variables include:

- SECRET – a randomly generated key for each developer
- SERVE_STATIC – allows the backend to serve images and file uploads
- PAYPAL_CLIENT_ID – client API key for each developer from PayPal developer account
- PAYPAL_SECRET_KEY – secret API key for each developer from PayPal developer account

- PAYPAL_SANDBOXED – allows the application to process payments without live currency (on by default)
- TWITTER_KEY – API key for Twitter that allows pulling tweets.
- TWITTER_SECRET – secret API key for Twitter to show tweets from OMC on home page
- SENDGID_API_KEY – allows for the application to connect with SendGrid (email service through Azure)
- SENDGRID_IN_DEV – allows for printing of emails in the console rather than sending to prevent charging OMC while testing

Obviously, these variables will have different values when running the live application and the sandboxing will be turned off. Errors for misconfigured environment variables will be shown in the console.

8.2 PostgreSQL Database

Since the previous team set up the application with a PostgreSQL database in mind, we will continue to use PostgreSQL so that we are not backtracking.
This application requires a lot of relationships between all types of users making a SQL database useful. Parents with children, volunteers with event shifts, students with courses, etc. At the time of writing this, the existing database will provide almost all the necessary information needed for the new features that we will be implementing. The ERD for the current application is shown below.

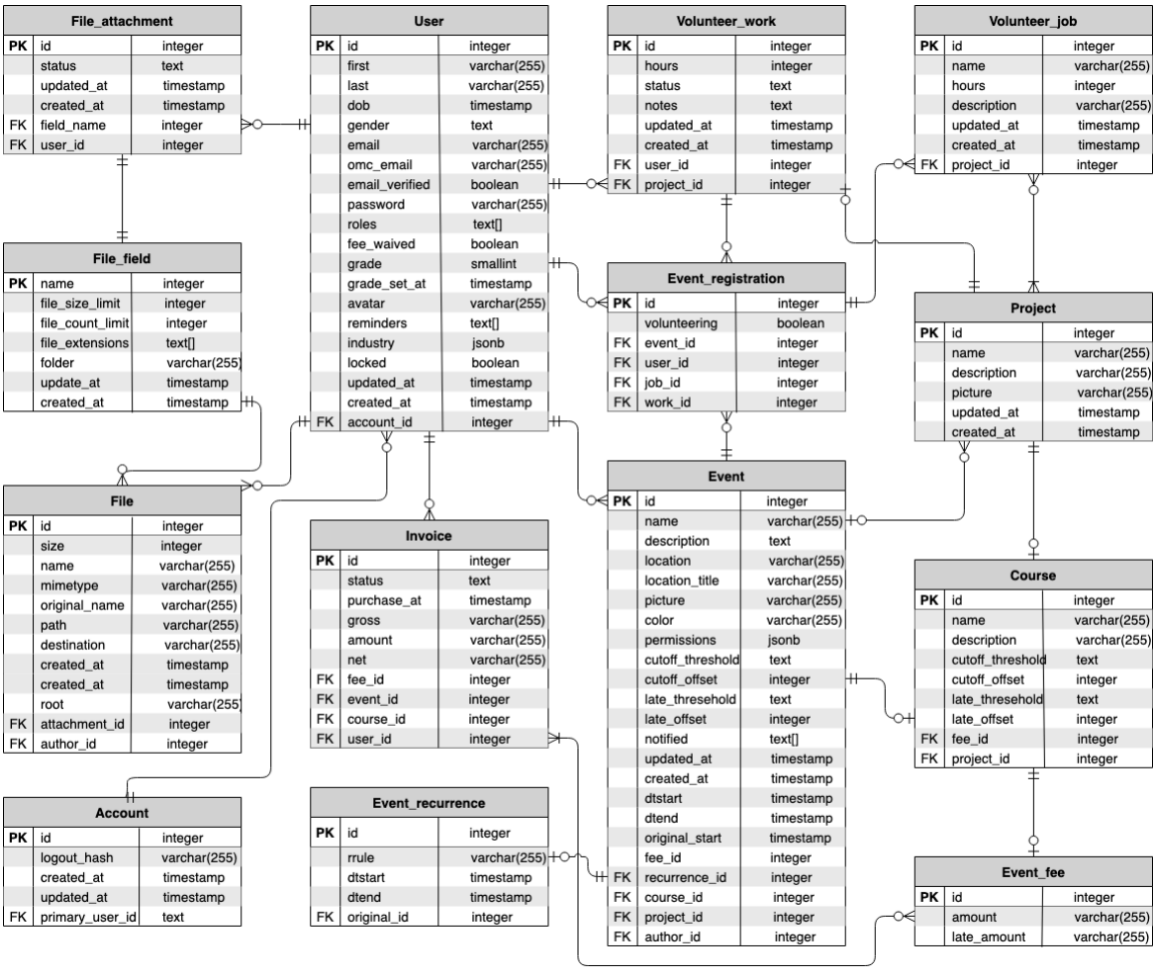


Figure 35. Database ERD.

Currently, there is no support for the swapping shifts which was a highly requested feature among volunteers that was left out of the current application. Below adds the extra fields to the User and Volunteer worktables that are needed to be able to support the balance feature for the accounts and the swapping feature as well. It is assumed that the rest of the ERD, tables and their interactions, is present and the same as above but not shown in the diagram below to highlight the changes and additions.

User		
PK	id	integer
	first	varchar(255)
	last	varchar(255)
	dob	timestamp
	gender	text
	email	varchar(255)
	omc_email	varchar(255)
	email_verified	boolean
	password	varchar(255)
	roles	text[]
	fee_waived	boolean
	grade	smallint
	grade_set_at	timestamp
	avatar	varchar(255)
	reminders	text[]
	industry	jsonb
	balance	money
	points	integer
	locked	boolean
	updated_at	timestamp
	created_at	timestamp
FK	account_id	integer

Volunteer_work		
PK	id	integer
	hours	integer
	status	text
	notes	text
	updated_at	timestamp
	created_at	timestamp
	need_swap	boolean
	confirm_swap	boolean
FK	swap_with	integer
FK	user_id	integer
FK	project_id	integer

Questions		
PK	id	integer
	description	varchar(255)
	grade	integer
	points	integer
	answers	text[]
	correct_answer	text
FK	correct_users	boolean

Figure 36. Necessary Additions to ERD.

Here is how we will be able to leverage the current design and these new fields to develop the new features that OMC would like us to implement.

- A volunteer dashboard will be available to anybody that was detected to be a volunteer upon sign in. Upon clicking on the page, the application will pull the necessary hours from each volunteer from the Volunteer_work table to display inside of the leaderboard. This page will also pull any events that the volunteer is registered for within a certain time period, possibly two weeks away, and display those events. The dashboard will also show any swap requests that are sent to the volunteer that is logged in.
- The balance will keep track of any money that was either overpaid or from another type of credit awarded to the user through means such as promotions or the point system.
- Any user that wishes to be a volunteer will have to complete the volunteer onboarding process by reading and signing the Volunteer Handbook and going through the child abuse training. This process will start once a user signs up to be a volunteer.

- Questions, at the start, will only be able to be created by admins for a grade level. In the future, it is completely possible to upgrade this feature to allow for students to upload questions for approval by admins to have their question featured as the question of the day.

8.3 Frontend

8.3.1 Volunteer Dashboard

In addition to the requirements originally requested by the client we have decided to add a high schooler approved dashboard for volunteers to view hours, shifts, and awards to the application. This volunteer dashboard will provide a unique experience to the user where they can see their current status, standing on the volunteer leaderboard, and it will provide a space where they can swap shifts with other volunteers if needed.

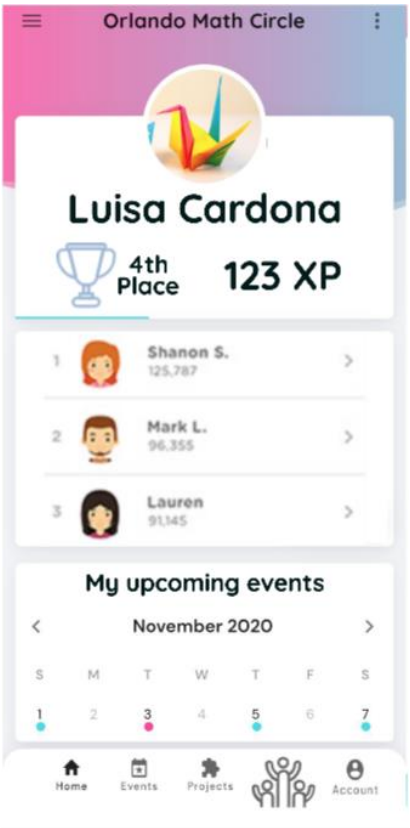


Figure 37. Proposed Volunteer Dashboard.

The volunteer leaderboard should be the focal focus of the volunteer dashboard. Figure 37 above shows a proposed prototype for this particular page. Here we can see at the top the avatar of the current user as well as their name, current standing, and number of points received from the events/courses they have participated in at the very top.

Right below this section the user should be able to see the rest of the leaderboard, this area should be scrollable, and it should originally display the top three volunteers. At the very end of the page the volunteer should be able to see their upcoming events and courses they are registered for, in this area they should be able to click on any given date, view the event, and be given the option to swap shifts with other volunteers if they no longer can attend the event.

8.3.2 Frontend Block Diagram

A high-level frontend block diagram for the construction of the volunteer dashboard is presented in the figure below.

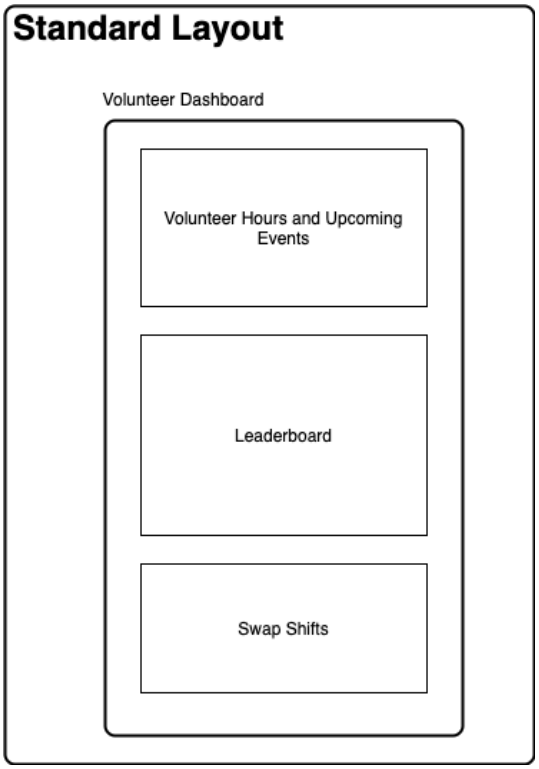


Figure 38. Block diagram.

8.3.3 Event System

As it stands, the current state of the application has a working event system that allows the creation of events for students and/or volunteers to attend. In the current event creation form, the fields will be presented in a bulleted form with explanations below. They are as follows:

- Title. The name of the event. Currently there is not a character limitation on the length of the event name so they can be as detailed as one would like, but there may be a possibility to implement this in the future to account for brevity.

- Duration. How long the event will last. There is an *All-Day* toggle box that allows the event to last for an entire day, helpful in the case where events translate to online. There are boxes that allow for inputting of date and time, and the length of these events are seemingly infinite.

- Grades. The grade of the user who can attend the event. Checkboxes range from Kindergarten to 12th grade, and many can be selected at a time. The implementation for this is smart since events can cater to more than one grade level. However, this field could be compartmentalized into three major school levels (Elementary, Middle, High) with individual grade selections upon expansion for a refined precision on material instructed.

- Gender. Events can cater to both males and females; later down the line, this selection box may be removed altogether in accordance with the laws and regulations that the Orlando Math Circle looks to stay compliant with.

- Thresholds. Two boxes follow that indicate the thresholds in which events can be paid or attended. The first one is a Late Threshold, which determines the maximum time that is allowed for registration of the event to occur before it is considered late. The Orlando Math Circle uses this for payment purposes to determine pricing when events are registered. Following that box, the next one is a Cutoff Threshold which delegates the timing where registrations for the event will no longer be allowed. Implemented to prevent attendee overflow, this is a good feature that is necessary for the Orlando Math Circle. However, the selections within the box, while concise in nature

("Minutes from Start", "Minutes from End"), do not allow for much specification. The possibility to change these to a time-box could be explored.

- Recurrence. The field indicates whether the event repeats or not. Used for events that may have a weekly frequency or once a month.

- Location. The area where the event will take place. Currently, the OMC application supports Online, Zoom, or In-Person events with a form box that allows for input of a custom location.

- Description. The information of the event, with a box that dynamically resizes as input is captured inside. Later down the line, if necessary, a character limit may be imposed as well to ensure that events maintain clarity and readability, but for now the system works as intended.

- Projects. The box appears that allows for the event to be linked to an existing project. Expanded on later, a Project is a series of events that come together to expand on a topic. What is interesting here is that a Project can be created right within the event to link it immediately. Innovative if one is looking to create a quick event and realize that it should be part of a larger scoped project.

- Course. Like the *Projects* box, the event can be connected to a course and if the course does not exist, can also be created right within the event pane.

- Payment Mode. This box specifies the type of payment that is taken for the event upon registration; whether the event is a Pay Per Event or a Pay Per Course, setting the flag on the event will indicate to the user what type of frequency the event is charged as.

- Picture. An optional picture that can decorate the event as intended.

- Color. An optional color field that can decorate the borders of the event.

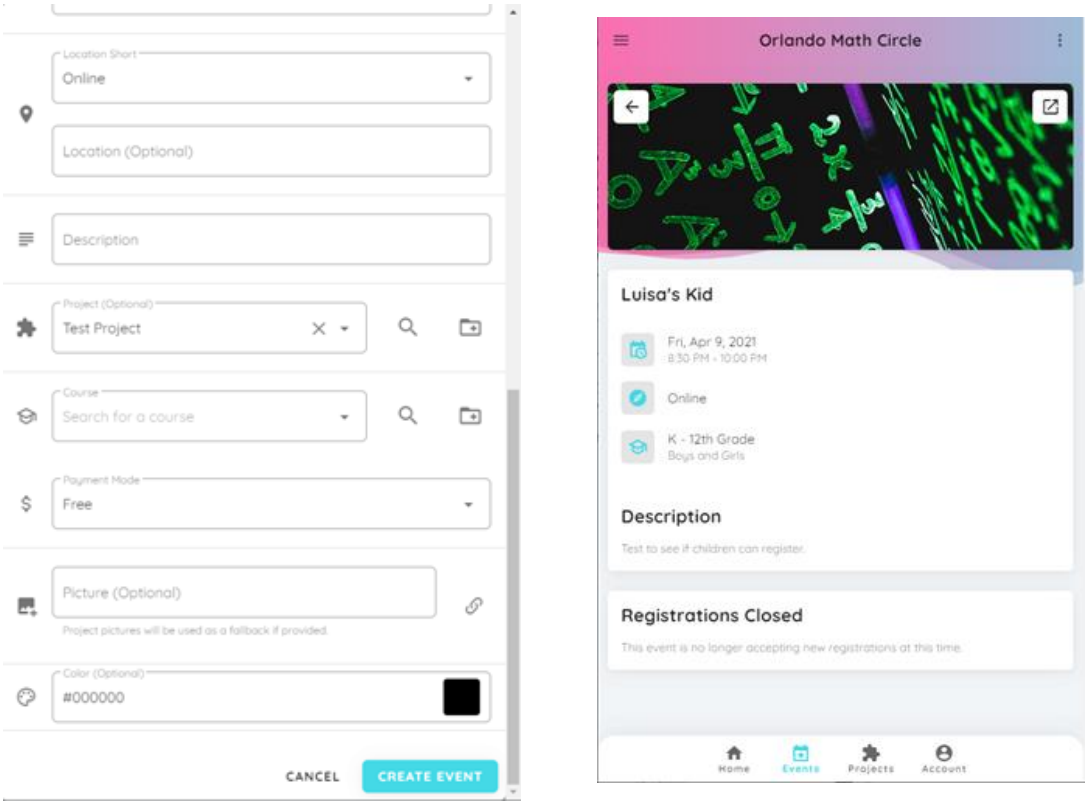


Figure 39. Event Creation System (left), User View after an Event is created (right).

When “Create Event” is clicked, the event appears on the calendar as a small dot on the day registered, indicating successful creation. The views on the “Events” page also contain the event. Expanding the event brings about the following display, as shown in the Figure above (right).

As one can see, the information of the event that is relevant to users is shown with the date and time of the event, location, grade, and description. Additionally, the status of the event – if registrations can be accepted or not – is shown below the description pane.

When creating an event using the current iteration of the Orlando Math Circle application, the process is quite procedural. There are many fields that must be populated and specified for an event and in terms of the UX interface, there are components of the registration that do come across as redundant. In accordance with the new features that the sponsors of OMC would like added to the event

system, ways to streamline the process and simplify the event creation form while keeping the required information intact is of most importance.

For instance, the Title and Description pane can be combined into one section where, after the name of the event is inputted, the description of the event can be expanded on as well. Right now, the description is required before the event posts, but a toggle can be turned on, like the Time section, that notifies the system if the event needs a description or not. The time section of the event registration can stay as the area contains important information that the OMC staff and volunteers will use to instruct the event.

In terms of the grades that the event will be presented to, it may be feasible to break the pane down into schooling levels – elementary, middle, high – as it represents a more natural configuration between the volunteer and the user. When the schooling level is selected, a dropdown of grades can be presented instead should the event creator wish to refine the topics to differing levels. Additionally, the connection to the Projects and Courses that are present in the event creation pane, while convenient, could logically be moved from that area and be connected after the event is created, allowing for an easier configuration throughout.

8.3.3.1 Desired Event Features

The event system, while fleshed out with features that the Orlando Math Circle deemed important at the time, quickly transported to a phase where improvements were needed. With more users looking to migrate to the OMC mobile platform and the timeframe that the application creation is taking place in (COVID-19 prevalent in the United States), event creation is more important than ever.

Currently, when an event is created, the assignee of the event along with the details are published and available for view. When an event is edited, only the event details can be changed. The assignee of the event cannot, and the Orlando Math Circle realized that this could pose a potential problem if a volunteer observed a life event that required them to postpone or cancel an event. In this situation, they utilize volunteers which come on and instruct the material whenever a class is available.

The Orlando Math Circle deals with this issue by deleting the event and having the volunteer that will take over the event recreate the event. However, the solution, while it satisfies the general case, is inefficient in nature and could cause additional problems down the line if the event is linked to a Project or Course. For this use case, the sponsors have concluded that shifts (events that are scheduled at a certain time) should have the ability to be swapped amongst volunteers. In doing so, the volunteer can accept the task and take it on for additional hours.

Events can also have cases where additional preparation time is needed to set up the environment before initiation. In this case, adding time is not feasible for the event because the users see when the event starts, and if they report to the time while preparation is occurring, it could create a frustrating experience for both the volunteer carrying out the event and the student attending the event. By allowing admins the ability to verify and/or add extra hours to the events, the volunteers can focus on orchestrating the event when the time is scheduled while still being credited for the entire time that was spent on the event.

In the event creation form, an event that is created currently needs both a start time and end time to signal the duration. What ends up happening in cases where events are coordinated to teach a certain topic is that, in varying situations, the actual time that the event takes to finish is unknown. Factors that are not accounted for range from student questions that could take up time to unexpected events and/or cancellations that may shorten event time. To account for this, the sponsors of the Orlando Math Circle determined that supporting open event times would solve the problem, allowing the volunteers to indicate ended event time if needed.

As mentioned above, students and volunteers can attend events through attending or volunteering respectively. The clients have expressed their concern for tracking attendance at these events for both types of users. Currently, OMC is manually writing down attendee's names on paper and then exporting that information to an Excel spreadsheet to track attendance. Automation of this feature will save the clients a substantial amount of time. Attendees should have the ability of marking their attendance to events and administrators should have the ability to view these attendance records to ensure that those who have marked attendance truly attended those events.

8.3.3.2 Possible Event Solutions

To storyboard possible solutions for the features that the OMC sponsors seek inside of the revamped application, the sections will be broken down into headings that contain ideas that can be used to explore the functionality.

Swapping Shifts

Idea #1: Create an additional field within the event creation form that specifies the assignee of the event.

Pros: Swapping shift functionality is achieved for free because when the event gets edited, the assignee of the event can be edited as well. Additionally, this can open the opportunity for other users to create events for one another, easing the process on the administrative side.

Cons: If assigning an event to someone is delegated to a form box field, this means that any user who wishes to create an event can assign an event to someone. While seemingly convenient, it can create issues down the line if an event was assigned to someone at the last minute.

Workarounds: If a volunteer were to register an event for another, the person who is receiving the event could have the option to approve or deny the event. That way, if the event were to be accepted, the event would appear on the calendar with notice from both parties that the event was created.

Idea #2: Implement an option, alongside editing event, that allows the event assignee to be edited. Designed as a read-only option accessible only to the user that created the event, they would select this option to change the person teaching the event to another individual.

Pros: Because the option is available only to the person who created the event, if the user needed to swap shifts with the individual, once the shift was swapped the permissions would pass onto the user. It would also observe a One-to-One relationship where maintenance of shift swaps could be delegated strictly between both parties.

Cons: Implementing event swapping in this manner would directly cut out the administrative privileges in managing events. Because the transaction happens

between two parties, visibility would not be granted to administrators, and should a case appear where the event is placed in a pending state and the receiving volunteer cannot accept the event, administrative users cannot override the request.

Workarounds: Add visibility of the event swapping to administrators so they can manage the transition as necessary.

Idea #3: Implement a Trello board-like style where an event that gets created has no assignee. As the system currently stands, the event can be created, and a name is prefixed by the authenticated user who created the event. In this new solution, the event can be placed in a state where it must be picked up by an individual. An idea of how this would look can be seen below in Figure 40.

Pros: Like Idea #1, the event swapping functionality is achieved for free because the user that can no longer pick up the event can return it back to the pool of events for another user to pick up. Additionally, developing the event system in this manner will ensure that each volunteer who picks up the event is in fact teaching the event.

Cons: Using this solution would mean that every event created would essentially be placed inside a pending state and would never cement onto the calendar unless a volunteer picks up the event. It could also add another layer of frustration should the volunteer who is creating the event also wants to teach the event (Create Event à Drag to Event Pool à Assign Event rather than Create Event).

Workarounds: Assign the event to whoever originally creates the event and should the event need to be swapped, allow the user to drag the event to the pool of open events.

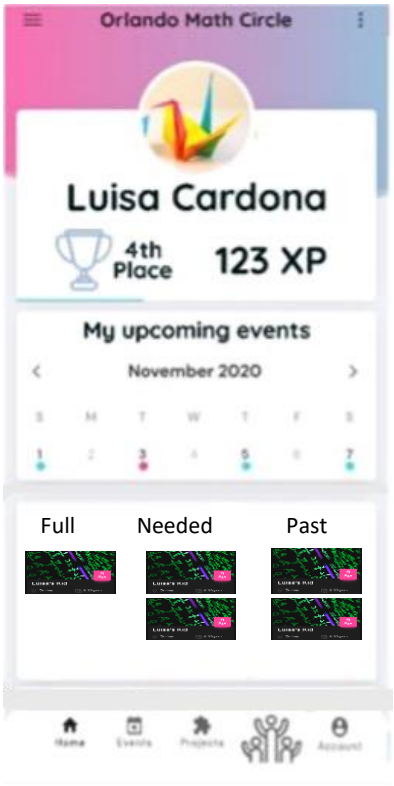


Figure 40. Trello Volunteer Idea Visualized.

Adding Extra Hours to Events

Idea #1: Let the user mark an event or toggle a flag that will alert staffing on the administrative side to add hours to events.

Pros: By doing this, admin can ensure that the correct events receive the extra hours that were owed from the event. For instance, if the event took place from 7:30 to 8:30 but ended up ending at 8:50, the admin would notice the flag that would prompt them to add the additional time.

Cons: The volunteers that need additional time would need to remember to flag the event at the end should time go past. Later down the line, it could lend itself to a cumbersome user interface and bring down morale of the application.

Workarounds: Craft a solution that allows the events to flag dynamically and alert admin that extra time needs to be added.

Idea #2: For the volunteers, implement a “Check-In” and “Check-Out” system that logs the time the volunteer started the event and the time the volunteer ended the event, including preparation and cleanup.

Pros: Because the volunteers control logged time, the algorithm that grabs the hours to record would only have to account for two factors: time in and time out. Using this would also eliminate the need for admins to add in the time and would only require verification.

Cons: Extra developmental effort would need to be taken to create the check-in and check-out buttons. The question would ultimately arise if coding the buttons and logic would compensate for easier implementation of extra hours added.

Workarounds: Not applicable.

Idea #3: When the time comes for admins to verify/approve hours, give them the option to update the time altogether based on volunteer discretion.

Pros: Resembling closest the needs of the Orlando Math Circle, the simplicity and ease of use in changing the hours logged would be best implemented within the admin panel. Additionally, no extra features would need to be developed.

Cons: As there are not cons that can be thought about at this time, this seems like the most reasonable choice.

Support of Open Events

Idea #1: Expanded upon earlier, implement a toggle function that allows the inputting of the end time to be bypassed. When the event finishes, the volunteer can close out the event by inputting the ending time.

Pros: In doing this, open events are supported, and events can be created without knowing the timeframe that they will end. This can allow for planning of events, setting up a calendar that can change dynamically as the needs of OMC change.

Cons: This would require that the volunteer remembers to close out the event every time. Additionally, the administrative side would have to approve all events that come in.

Idea #2: Implement functionality that can append events to the calendar like a placeholder. Coupled with the logic that allows the volunteers to check in and out, all events would specify a range of time that the user could set, with true timing spent from the user check-in and check-out.

Pros: Events, in this state, would truly be “open”: they would show to the students and minors looking to attend the event only the time that the event starts, while the volunteers would control the duration spent on the event.

Cons: Event cancelling would become more difficult if the volunteers oversaw controlling the event time. Because the volunteers manage the crediting of time, if an event were to be cancelled a different set of rules would have to be created to manage that process.

Workarounds: Perhaps, if the events were to be implemented in a Trello-like fashion, the events could just be dragged to the pool of events that would be opened to be picked.

Tracking Attendance of Events

Idea #1: Allow users to select a dropdown with the options attended or did not attend after an event has finished.

Pros: Attendance will be tracked eliminating the need for manual tracking of attendance by the clients.

Cons: Students/volunteers will have to remember to log in and navigate to the event to check select whether they attended the event or not. If some time has passed, students may forget that they had not marked their attendance losing out on the hours/rewards from said events.

Idea #2: Allow administrators to mark attendance inside of the admin panel inside the registration tab.

Pros: Users that have registered for an event are visible from within the registration tab in the admin panel. This method would ensure that users that attended the events are marked without worrying that a user may falsely mark themselves as attended when they absent.

Cons: The clients will still have to manually track attendance during the event with a solution such as pencil and paper and then manually mark the registered users as attended or absent.

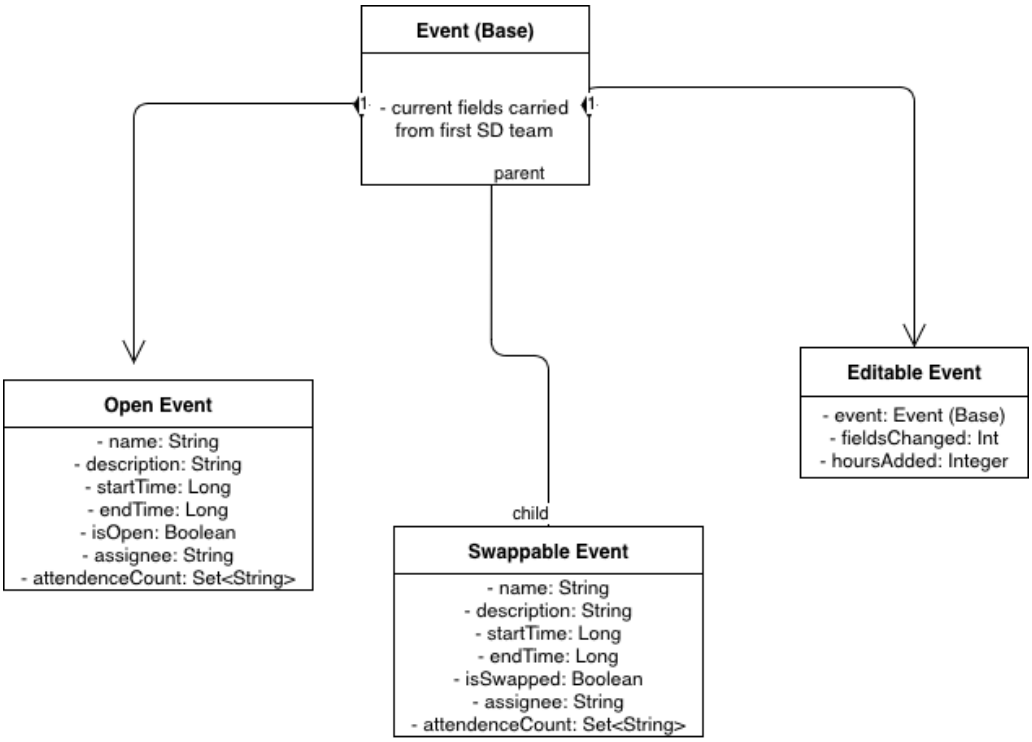


Figure 41. UML class diagram outlining the potential design of the classes that will support the features explained above.

The UML diagram above showcases the ideas generated by the Senior Design team into classes that could be replicated or designed throughout the lifecycle of the Orlando Math Circle application. In keeping consistent with the previous team's design choices for information deemed important inside events, the information will be represented as a base event. In other words, for an event to be created it will have the fields that were previously coded up available for population. The only change here is that logic will be constructed that will allow for some of the fields to be optional and/or required. Additionally, this segues into the reason why most of the fields represented inside of the Event classes are objects instead of primitive types; by making this design choice, the empty value can be supported by setting fields to null should this UML diagram outline the system for the event features.

In the case of an open Event, the fields available that will be able to be represented is a name, description, start time, end time, a flag indicating if the Event object is in fact open, the assigned person to the Event, and a Set of Strings denoting the number of attendees for the Event. Design choices made that ensured that the Event was represented as open ranged from objectifying the startTime and endTime fields as Long values to a field that denotes the Event as open. By making the startTime and endTime fields Long values, existence can easily be checked by verifying that the time values are not null. If they are, the administrative side of the application can then manipulate the values as needed to satisfy the requirements of an Event on the backend side. Additionally, while null startTime and endTime values are enough to show that the Event is open, for additional safeguarding and practices, a field will be populated describing if the Event is truly open. In adopting this practice, the OMC application can take additional steps to process an Event like it was intended.

Swappable Events carry over the same fields as Open Events, but the key difference here is that the Boolean flag checking if the Event is open is refactored to check if the Event is swappable. Rather than incorporate said Boolean flags into one Event object, separating the logic into Event objects help to support clarity and brevity throughout the application backend and is less likely to cause issues for future developers armed with the task of improving upon the logic. If an Event is swappable, it will be defined as a Swapped Event that will provide getter and setter methods to redefine the assignee of the Event.

All Events in theory are editable, but to limit the chance of cyberattacks happening to the application and to promote security, an editable event will be defined as such, instantiated as an Editable Event object. Here, the object, inheriting from the base Event, will contain a count of the fields that were changed as well as extra hours that can be added to the event should the volunteer stay past scheduled time. Defined as an Integer field for now, this has potential down the line to change to a Float or Double field to allow extra precision in adding hours. However, since this is a new concept introduced to the OMC application, mathematical formulas may be more efficient in generating the exact time to keep interactions simple inside the Event pane.

To summarize, the UML diagram emphasizes the following relationships:

- Base Events serve as the parent, containing information that each Event child will base additional information on.
- Open Event, Editable Event, and Swappable Event will inherit from the Base Event utilizing an “is-a” relationship.
- (Possible solution) a later implementation may represent all the fields specific to Open Event, Editable Event, and Swappable Event inside Event (Base) with the only statements checking if the Event is an instanceof the supported Event types.

8.4 Backend

The backend of an application is one of the most, if not the most important aspect of an application as it simulates the heart of the application. With data flowing through the application from handlers that send and receive the necessary information, the application needs to have a logical hierarchy in which each and every aspect of a user’s needs, functions, and wants are considered. Through the use of NestJS, which the OMC currently uses, the previous Senior Design group was able to achieve the modular framework that the dynamic web application through breaking functions down into classes nested inside folders. Before the hierarchy of the web application is explained, it would be best to delve into the specifics of what exactly NestJS is and why the previous Senior Design group decided to use it.

8.4.1 NestJS

NestJS, building on NodeJS, is a framework that allows for efficient Node.js server-side applications and uses progressive JavaScript, is built with and fully supports TypeScript (yet still enables developers to code in pure JavaScript) and combines elements of OOP (Object Oriented Programming), FP (Functional Programming), and FRP (Functional Reactive Programming). [3]. Because of its basis in Angular, it shares features such as a file structure that is modular in nature, an injection system, and usage of decorators, which will be expanded on below. Through the robust documentation website, it provides information on fundamental techniques that help to make use of NestJS’s most powerful concepts such as providers, which can be both asynchronous and synchronous in nature, dependencies, and injection. Additionally, it yields support for authentication, database management,

validation, file management, and more. In the first senior design group's thought process, they've focused extensively on the NestJS basics that allow for Create, Read, Update, and Delete (CRUD) operations, the database management, and the idea of relationships, which is most important for the use case of the OMC application.

8.4.1.1 NestJS Basics

In working with NestJS, the modular structure helped to ease developmental efforts in creating the OMC web application that would highlight all of the CRUD operations that were most important for OMC. In hindsight, the Module class serves as the building block for NestJS as it specifies the metadata that is used to organize the application structure, the Controller class help to handle incoming and outgoing requests, bridging the connection between the client and server, the Provider class is a class that can be used to inject dependencies and create relationships with one another, and Validation, while not a class, is a pipeline that automatically handles checking for correctness of data that is provided through Controllers.

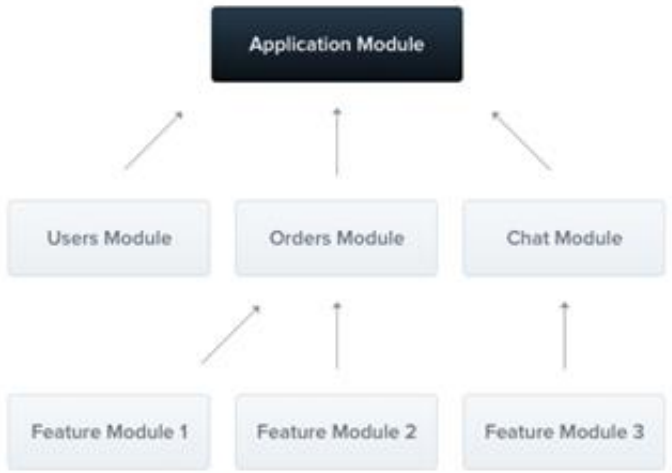


Figure 42. Visual diagram illustrating Modules.

Modules

Modules serve as the building block for the structure of applications written through NestJS. When many are created together, they help to form the basis for a structured application. Each application that is created always starts with a root module, and the root module is used to build the application graph that helps to establish relationships between providers and dependencies. Inside the Modules class, it's presented as a decorator that accepts the components listed above (providers, controllers, imports, and exports) and they are encapsulated within to ease the developmental process. There are varying types of Modules that NestJS supports, and they will be expanded on further below.

Feature Modules

Feature Modules, as explained from the documentation, helps to organize code for a specific feature of the application. Connecting the premise to the OMC application, feature modules are an integral building block to the application, structuring the files amongst folders that correspond to the specific feature. By doing this, NestJS manages complexity and descending relationships with the principles of the development intact.

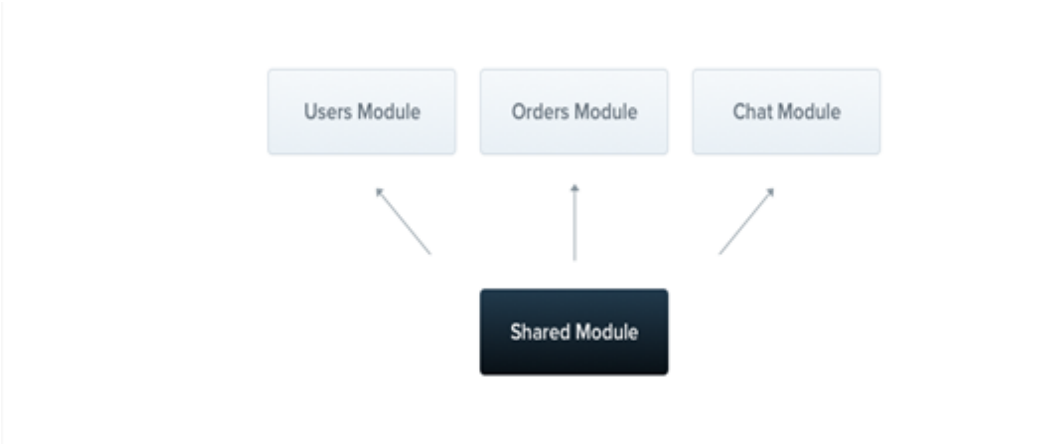


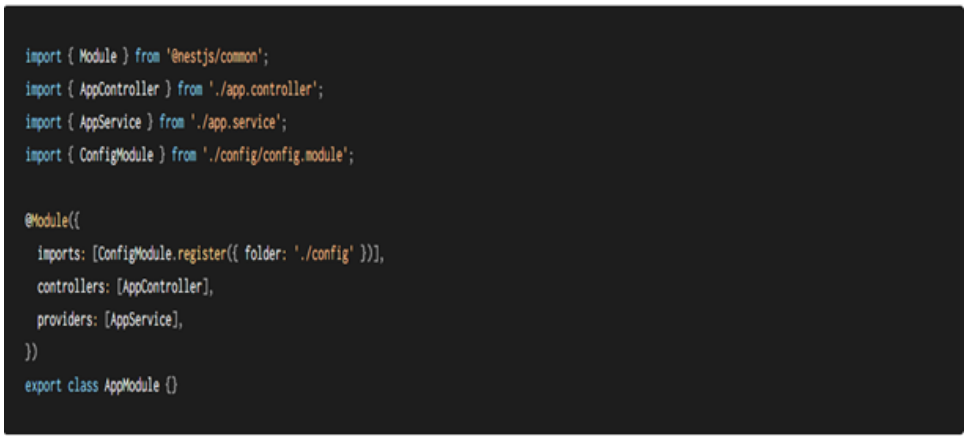
Figure 43. Illustrated relationship of a Shared Module, having the ability to spawn different modules.

Shared Modules

Shared Modules allow for instances of the same provider to be used amongst multiple modules. Following the same principle of Modules where a type of module is always based on a root module, each module that is created is also shared due to them being singletons by default. To reuse a module throughout a project, export the module inside of the project that is seeking to use the functionality.

Global Modules

As defined in the name, a global module is a module that has the ability to be available everywhere within the scope of the application. By using the @Global decorator, modules will only need to be instantiated once and won't need to be imported or exported throughout. While convenient, Nest does not recommend that all modules are defined as Global - the proper way to request module functionality would be to import the module as necessary.



```
import { Module } from '@nestjs/common';
import { AppController } from './app.controller';
import { AppService } from './app.service';
import { ConfigModule } from './config/config.module';

@Module({
  imports: [ConfigModule.register({ folder: './config' })],
  controllers: [AppController],
  providers: [AppService],
})
export class AppModule {}
```

Figure 44. Code example of Dynamic Modules. What denotes the module as dynamic is the .register() method found inside the imports[] block. By allowing the passing in of a .config file, one is granted a module that can change based on the configurations upon launch.

Dynamic Modules

Perhaps the most important and integral part of Nest, dynamic modules is a feature of Nest that, for customizable modules, can configure and register providers dynamically. While simple in nature, the power behind modules having the ability

to change configurations is that the same module containing information specific to an application can be instantiated using different fields and settings. Beforehand, for all of the module examples and definitions provided above, they each operated under the pretense that the modules were static. In static modules, the definitions are outfitted with providers and controllers that are kept stagnant throughout; the scope of the components depends entirely on where the module is imported and exported.

The process explained above where the module is imported and exported into others built upon *Shared Modules.* For a module to be imported, the module that is receiving the import is responsible for configuring the module to work as intended with the application feature. In doing this, inconveniences arise when other modules need to use the imported module. For each module that wants the features of the providers and controllers, they must specify the workings of the module before it is used within Nest, and for an application that requires the use of the module multiple times – for instance, a home pane – the implementation could get tedious on the developer end. This is where dynamic modules come into play: when the application is started, Nest will create a module at runtime with the specified parameters and options as configured in the .env files. In fact, the current method to launch the OMC application in the developmental side requires the use of env variables both in the frontend directory and the backend directory.

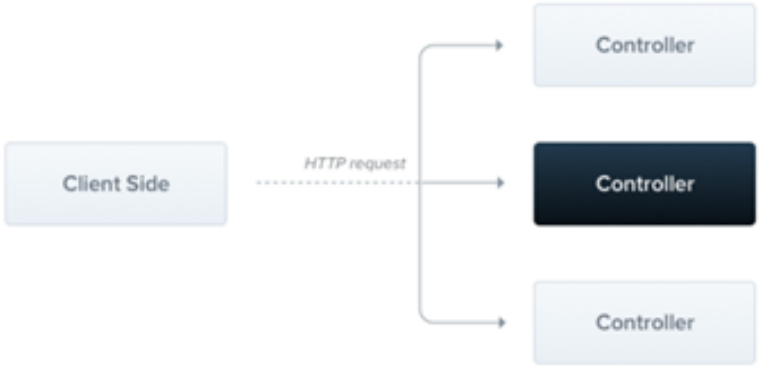


Figure 45. Diagram illustrating the operations of a Controller.

Controllers

While modules serve as the building block for the application structure, controllers handle the information routed between the client and server. Because they handle incoming requests and responses, they help to ensure that data and information is passed around real time. Controllers allow the configuration and usage of different routes and each route can perform different actions depending on the request.

Creating a controller can be completed either by using Nest decorators and classes or invoking the following command from the command line, assuming that Nest dependencies are injected into the terminal:

nest g controller cats.

Routing

One of the key functionalities of a Nest controller is routing, and it describes the process where a path is selected to direct network traffic between controllers. Defined using the @Controller() decorator, for a controller to be able to handle messages, a path prefix must be specified. The path prefix contains a group of routes that the controller is configured to route the response to, and the information is handled through methods that are bound to the controller itself. Once the data is manipulated as necessary, a response is generated through Nest that is manipulated using either of the two methods:

- Standard (Nest-recommended). For any JavaScript object or array that is passed into the controller as data to be manipulated, the resulting payload is automatically encoded and decoded as JavaScript Object Notation (JSON). Primitive JavaScript types, however, are sent through Nest as they are, allowing for an eased user-experience and efficiency of the routing.
- Library-specific. Response-objects that are library-specific allow for customizable responses to be returned to the application.

HTTP Methods Supported

Currently, the HTTP methods that are handled on the network side also yield support from Nest. Using decorators to invoke them, they are as follows:

- GET - @Get()
- POST - @Post()
- PUT - @Put()
- DELETE - @Delete()
- PATCH - @Patch()
- OPTIONS - @Options()
- HEAD - @Head()
- (Nest Specific) - @All(), handling all of the requests above.

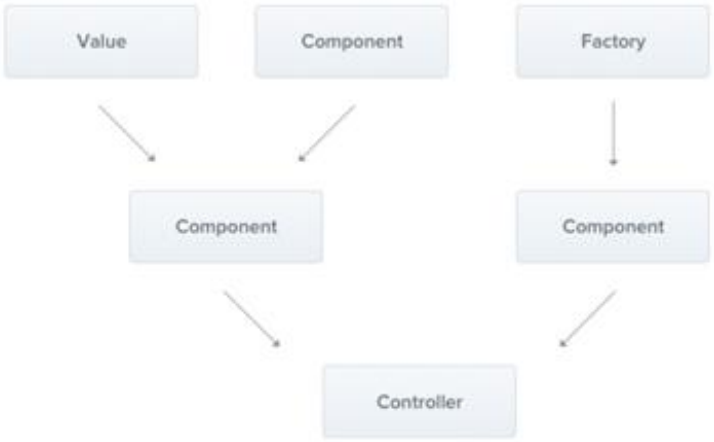


Figure 46. Diagram illustrating a Provider.

Providers

One of the key components of Nest, providers are important because they inject dependencies into the project, creating relationships between components of the application without defining them explicitly.

Dependency Injection

For a provider to be able to utilize the components of a class or module incorporated inside, the dependencies are injected. In using this design pattern, the module, instead of creating an entire new instance to make use of the functionalities present inside the imported module, requests only the dependencies from the external sources, returning them once completed.

Security

With a framework as developed as Nest, it is no surprise that they provide extensive documentation on various technologies used for authentication. Rather than detail a single prominent method in dealing with authentication, they provide the information based on approaches that the developer may take. The host of methods and functions that Nest supports are compartmentalized into Passport, an Express-compatible authentication middleware for Node.js.
As Nest supports this usage from Node.js, they promote the usage of Passport throughout the documentation examples to yield a streamlined authentication process.

Passport

Created specifically for Node.js and compatible with Express, the purpose of Passport is to provide authentication for requests, using plugins native to the software. These plugins are known as strategies, and Passport employs these strategies to help ease the process and promote efficiency in authentication. Through the API, Passport is provided a request to authenticate the information sent through, and – with the use of the strategies – provides logic to delegate the success or failure state of the requests.

Strategy	Protocol	Developer
Local	HTML form	Jared Hanson
OpenID	OpenID	Jared Hanson
BrowserID	BrowserID	Jared Hanson
Facebook	OAuth 2.0	Jared Hanson
Google	OpenID	Jared Hanson
Google	OAuth / OAuth 2.0	Jared Hanson
Twitter	OAuth	Jared Hanson
Azure Active Directory	OAuth 2.0 / OpenID / SAML	Azure

Figure 47. A table illustrating the various authentication strategies that Passport supports.

Strategies

Strategies are defined as the logic that drives the handling of requests. Ranging from authentication logic that can be used by popular social media handlers (Facebook, Twitter, etc) to custom strategies that can be instantiated to define relevant information that is needed to allow a user into an application. Passport requires that strategies that are used to authenticate requests are handled by the user, delegating which information from the user is needed to allow access. For example, if a user is logging into an application, the developer may require that the user provides an email and password. The strategy utilized by the developer will define the pathways that will be taken once the information is received.

```
passport.use(new LocalStrategy(
  function(username, password, done) {
    User.findOne({ username: username }, function (err, user) {
      if (err) { return done(err); }
      if (!user) { return done(null, false); }
      if (!user.verifyPassword(password)) { return done(null, false); }
      return done(null, user);
    });
  }
));
```

Figure 48. Code sample showcasing a custom strategy that checks if the user's login credentials were successfully authenticated.

Once a user is authenticated through the system, Passport will store the state of the request and utilize sessions to remove the need for a user to be authenticated again into the system. For the developer, once the user is authenticated, they must manage the state of the session by serializing and deserializing the user to the session upon login and logout. If one wishes to use the standard, built-in authentication logic provided by Passport, they can use the *authenticate()* function, as demonstrated below in Figure 49.

```
app.post('/login',
  passport.authenticate('local', { failureRedirect: '/login' }),
  function(req, res) {
    res.redirect('/');
  });
```

Figure 49. Usage of the passport.authenticate() function.

Passport in Nest

Nest implements the use of Passport by allowing a user to configure the strategy by extending the PassportStrategy class. Through the strategies defined above in working with modules, the module that defines the methods and features for authentication would only need to be declared as an extension of a PassportStrategy. Afterwards, a module that would be used to inject the dependencies is then created and following instantiation, authentication services are handled elegantly.

Current Usage

The previous Senior Design group orchestrated the framework for the backend using the four building blocks described above. They have decomposed the functionalities that the Orlando Math Circle wanted into categories such as account, auth, course, email, events (which have other sub-folders nested into them to elaborate on deeper relationships), users, volunteers, and more. We applaud the decision to structure the data in this manner because when needing to add features and refine requirements presented to the group now, we'll know exactly where to go to build upon the functions. In focusing the requirements into a package that the team could build, the previous group pooled most of the time into cementing the requirements and building a scalable, workable solution.

8.4.1.2 Database Management

One of the benefits that the previous Senior Design group received from using NestJS is how the software manages database support. Since Nest is database agnostic, it supports integration of any SQL or NoSQL database, which was necessary since the database of choice that they used was PostgreSQL.

8.4.1.3 Relationships

The previous Senior Design group, aptly, designed the structure of the backend of the application through the use of relationships; by ensuring that each component of the data is shared either by ***Many to One, One to Many, One to One,*** or ***Many to Many*** relationships, the connections become efficient and natural throughout, making it easier to draft features since other operations are secured for free. Explained further below, quoted directly from the documentation [31]:

- **Many to One** – Many instances of the current Entity refer to One instance of the referred Entity.

- **One to Many** – One instance of the current Entity has Many instances (references) to the referred Entity.

- **One to One** – One instance of the current Entity refers to One instance of the referred Entity.

- **Many to Many** – Many instances of the current Entity refers to Many instances of the referred Entity.

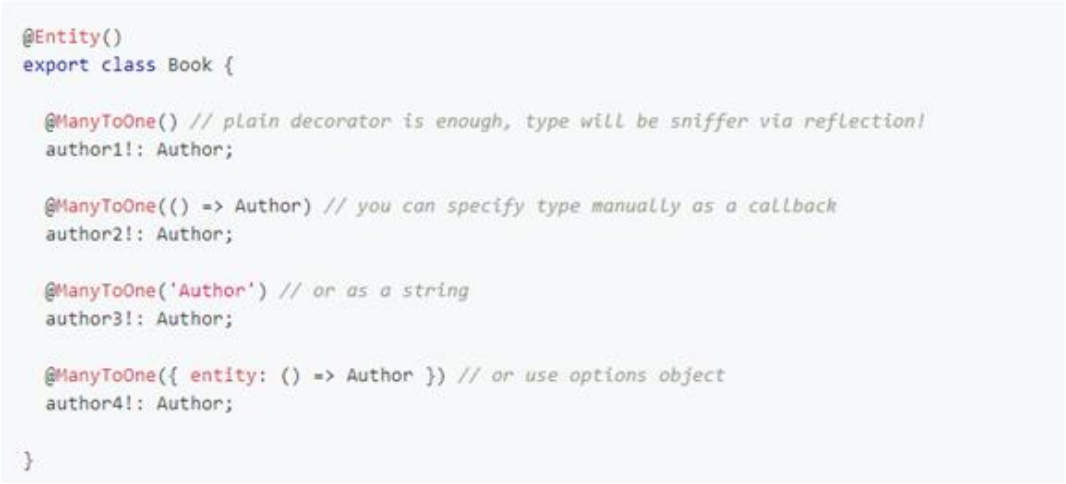


Figure 50. Code example of a Many to One relationship, demonstrated in Nest.

The application, in the current state, utilizes many of the concepts explained above, employing relationships where Many to One and One to Many takes common precedence. For instance, the process of registration, where parents and adults can register for multiple children into the Orlando Math Circle application operates under a One-to-Many relationship since one entity (the parent) yields control and access to many other entities (the children that they have registered to OMC). Because of this, the backend is coded in such a way that relationships and data that are passed around can be accessed by each.

Administrative abilities, delegated by the Orlando Math Circle staffing and volunteers that control the sector, would translate to a Many-to-One relationship simply because all accountholders and users of the application would be managed by the admin. This would also translate to the backend adapting a hierarchical design where common data shared amongst all entities are leveled at the top and permissions that are specific to other types of users are naturally placed lower within the design.

The following points below will highlight the current user relationships shared by the application:

- Admin à Parent – One to One
- Parent à Minor(s) – One to Many (the above relationship could also be stated as One to Many for this reason)
- Parent à Volunteer – One to One
- Minor à Volunteer – One to One

8.5 Networking

Networking, while fleshed out from the previous Senior Design team, is still an integral concept of the Orlando Math Circle application that provides the mechanism to pipe data throughout the databases in the respective parts. In handling this, the team plans to continue to utilize the solution ironed out by the Senior Design team of the past, which handled most of the work in determining the domains that will configure the frontend and backend to communicate.

Currently, the method that is used to handle networking is as follows: NGINX, open-source software that assists with web serving, cache services, proxy servicing, and more, is used to transport requests between the frontend and backend. The backend also contains its own separate URL that serves as an oversight for all the data needed by OMC and receives traffic from a server that is different from the frontend. The current Senior Design group believes that this was put in place to mitigate the number of collisions between packets that get transferred throughout the application. In conjunction with NGINX, the previous group sought out an automated tool, Certbot, that makes use of a secured socket layer (SSL) to generate keys in both the public and private side for authentication of ownership by the user.

The previous Senior Design group also makes usage of non-profit software named LetsEncrypt, and while they do not describe exactly how LetsEncrypt operates, it is known that the software assists in providing certificates for web applications to use when seeking SSL support. The documentation on LetsEncrypt is descriptive

with sections assisting the developer in setting up the web application for certificate support and use cases that the software can help to satisfy.

```
location / {
  expires $expires;

  proxy_redirect                      off;
  proxy_set_header Host               $host;
  proxy_set_header X-Real-IP          $remote_addr;
  proxy_set_header X-Forwarded-For    $proxy_add_x_forwarded_for;
  proxy_set_header X-Forwarded-Proto  $scheme;
  proxy_read_timeout                  1m;
  proxy_connect_timeout               1m;
  proxy_pass                          http://127.0.0.1:3030;
}
```

Figure 51. A code sample that describes the configuration of a root domain for proxying requests.

8.6 Server Management and Tooling

As the previous senior design group handled most of the management of servers through the PM2 library – a library that supports the usage of npm commands, commands heavily used throughout the OMC application nomenclature – the library assists in automating most of the handling of server errors that may come across. When a service interruption is encountered (internet disconnection, database outage), the library automatically restarts and brings the application back to the latest remembered usability state before failure.

While automation of these tasks helps to promote efficiency throughout the application, it's important to break down the information that the Orlando Math Circle stresses to maintain and how the server management plays a role in safeguarding this information.

Software Management

Software is a sector that changes dynamically as hardware is updated and optimizations are made in current technologies. Knowing this, it is of most importance that the Orlando Math Circle keeps current with the needs of the

consumer and software demands to promote the greatest possible user experience. Software management helps to achieve this by keeping up to date records on dependencies and programming language changes.

For instance, if a particular section of the application is storyboarded using methods that have been deprecated, optimizations and performance enhancements may be missed out on if the logic is not updated to support the latest methods. Through robust documentation of the testing process, classes held that will utilize high-school students to familiarize oneself with the application and programs used, and the convenience of the PM2 library in automating these tasks, the OMC application safeguards itself in longevity.

Security

Needing no explanation, security of an application is just as, if not even more important than the application itself. Operating together with software management, keeping the libraries and software up to date along with the industry standard allows the OMC application various benefits; for one, the application is supported should a programming concern appear that is necessitating the use of external aid, software exploits that may have been discovered in previous iterations are defunct, and most important of all, users registered with the Orlando Math Circle can use the application knowing that information about themselves and minors is most secure.

8.7 Gantt Chart

We decided to utilize a Gantt chart to outline the project sequence and illustrate the scheduling of the project.

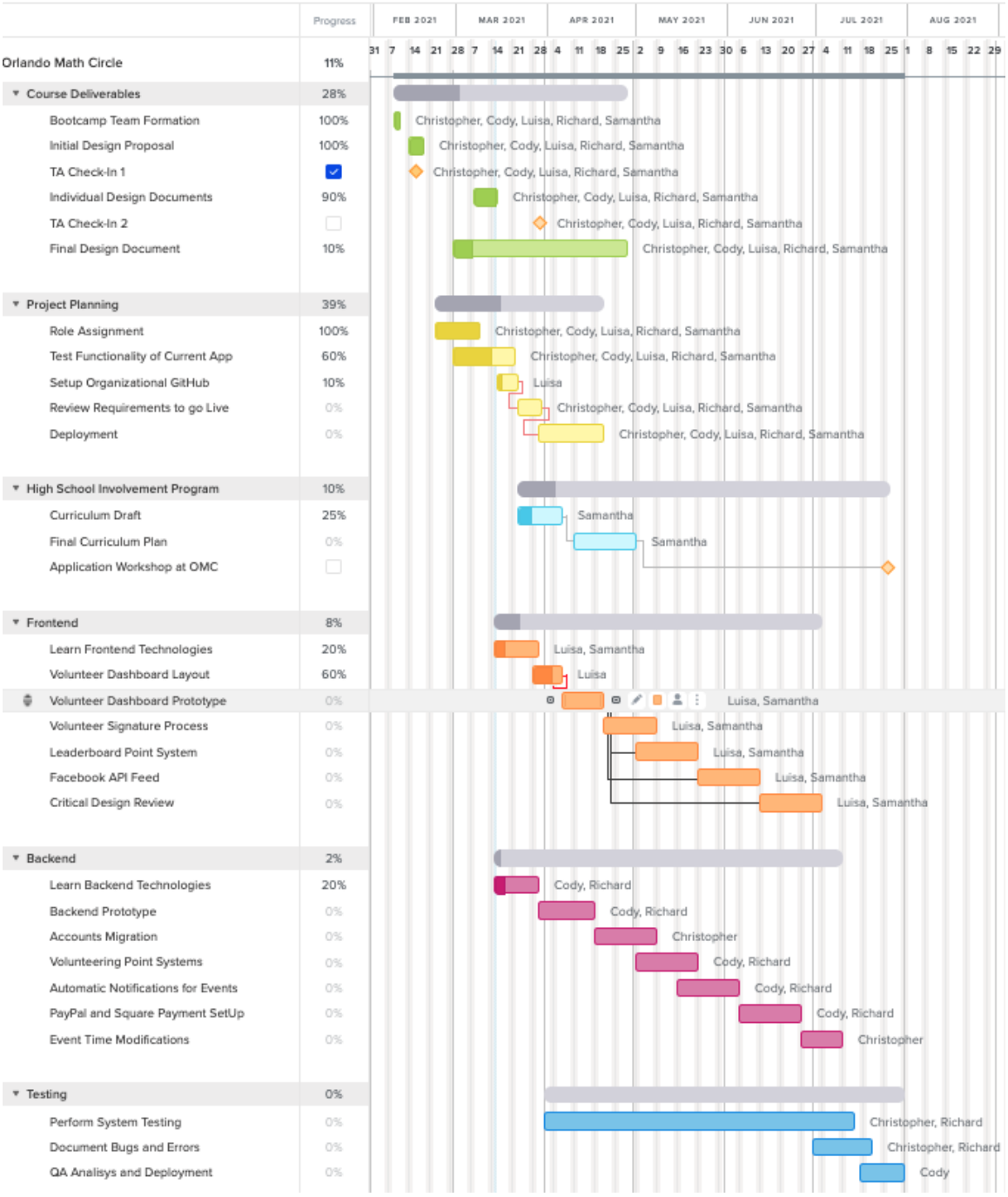


Figure 52. Gantt Chart

8.8 Bug/Issue Tracker

Given that a main component of our project relies on testing we have implemented a bug/issue tracker to record any bugs encountered during the review process. This dashboard allows any team member to report an issue with the app in detail (Figure 53), all information needs to be filled out so that the error can be duplicated and assessed.

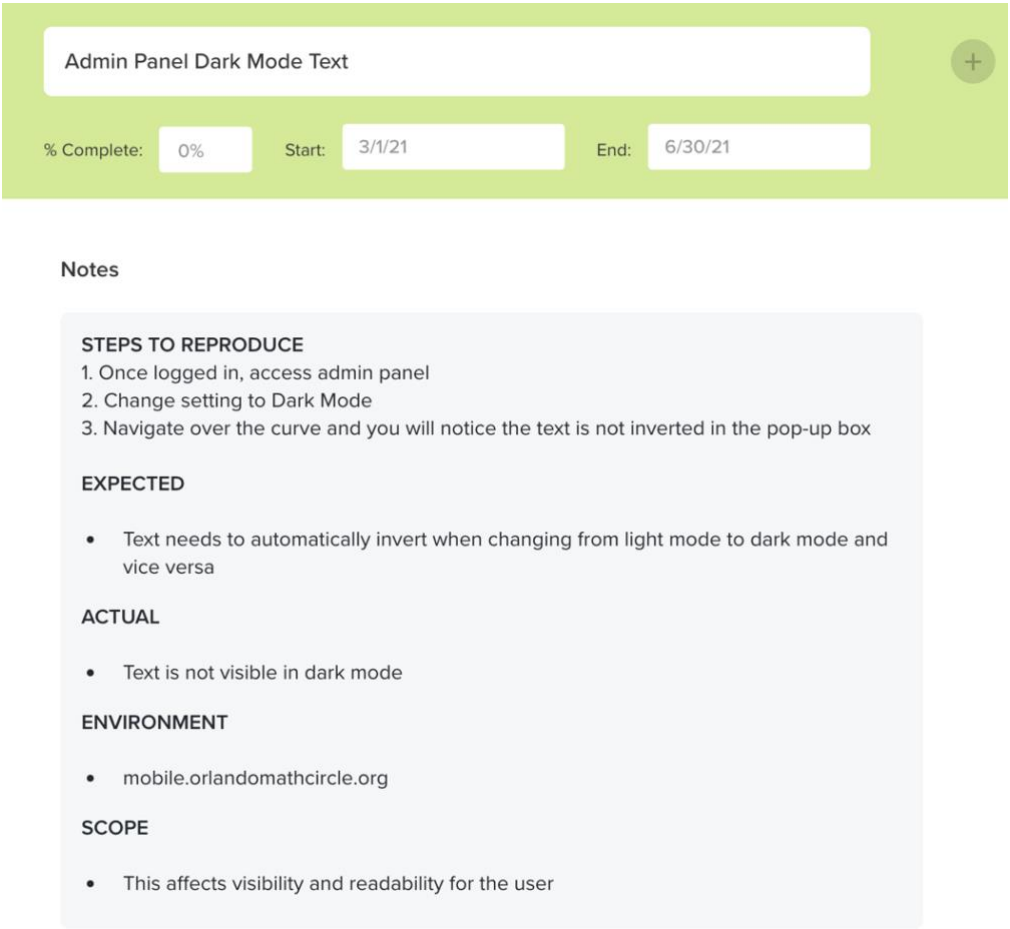


Figure 53. Reporting a bug sample.

As we familiarize ourselves with the application, every team member is in charge of reporting any issues or UI inconsistencies that are encountered during the process. Eventually, we will collectively assess and assign all bugs reported in the dashboard.

Once a bug has been assigned to a team member, that person is responsible of moving the task over to the “In Progress” section. If any issues occur or if the report wasn’t detailed enough, the task must be moved over to the “On hold/needs more information section” so that the person who reported this specific task can go over it and add any additional information needed. Once a bug has been resolved, the task must be moved over to the “Closed” section.

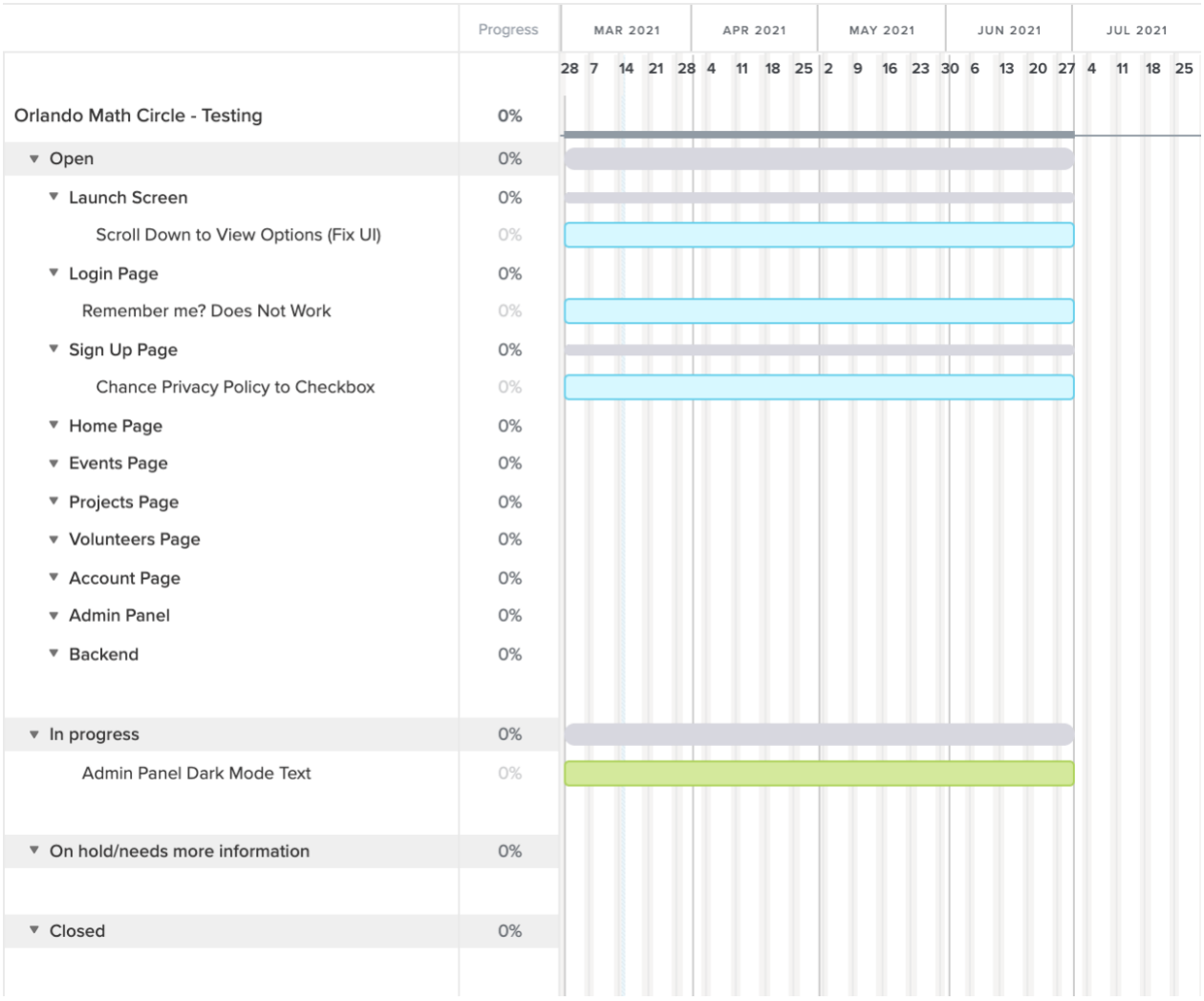


Figure 54. Bug/Issue Tracker.

Any individual assigned to a bug is responsible for updating the progress of said task and make sure it is resolved in a timely manner. For simplicity we have broken down the application based on pages i.e., Home Page, Login Page, etc. and every bug should be classified accordingly (Figure 54). We hope this dashboard provides an efficient way for the Senior Design team to document any existing bugs in the

current application as well as any issues that might arise during the development process as more features are added.

8.9 Use Case

Figure 55 below exhibits the use case diagram for how users will interact with the application. This diagram is an updated version from the previous senior design team and introduces the features our team will be implementing. Upon the application's login page, users will have the ability to either sign up for a new account or login with existing credentials. Though users will cover a variety of ages, only users above the age of 13 will be able to register an account to follow COPPA regulations. Any user can become a volunteer, but they will not have access to volunteer specific actions unless they sign the volunteer handbook. The following list shows the actions a user can take after signing in:

- Users can upload files for administrators to process, such as free/reduced lunch forms for students
- Users can view the OMC event calendar
    - Parents can add children to events
- Parents can add children accounts under their own, creating a profile that can be switched to within the app
    - Parents can edit their student's information, where they will be prompted to on an annual basis
- Volunteers can view and sign the volunteer handbook
    - Volunteers can view their collective hours and overall leaderboard
    - Volunteers can sign up to volunteer for events
        - Volunteers can swap events with other volunteers
        - Volunteers can delete events, which will notify other volunteers that a spot under the event is available
- Admin accounts can manually register volunteers under events
- Admin accounts can create individual events to place on the event calendar
- Admin accounts manually edit user account information if desired

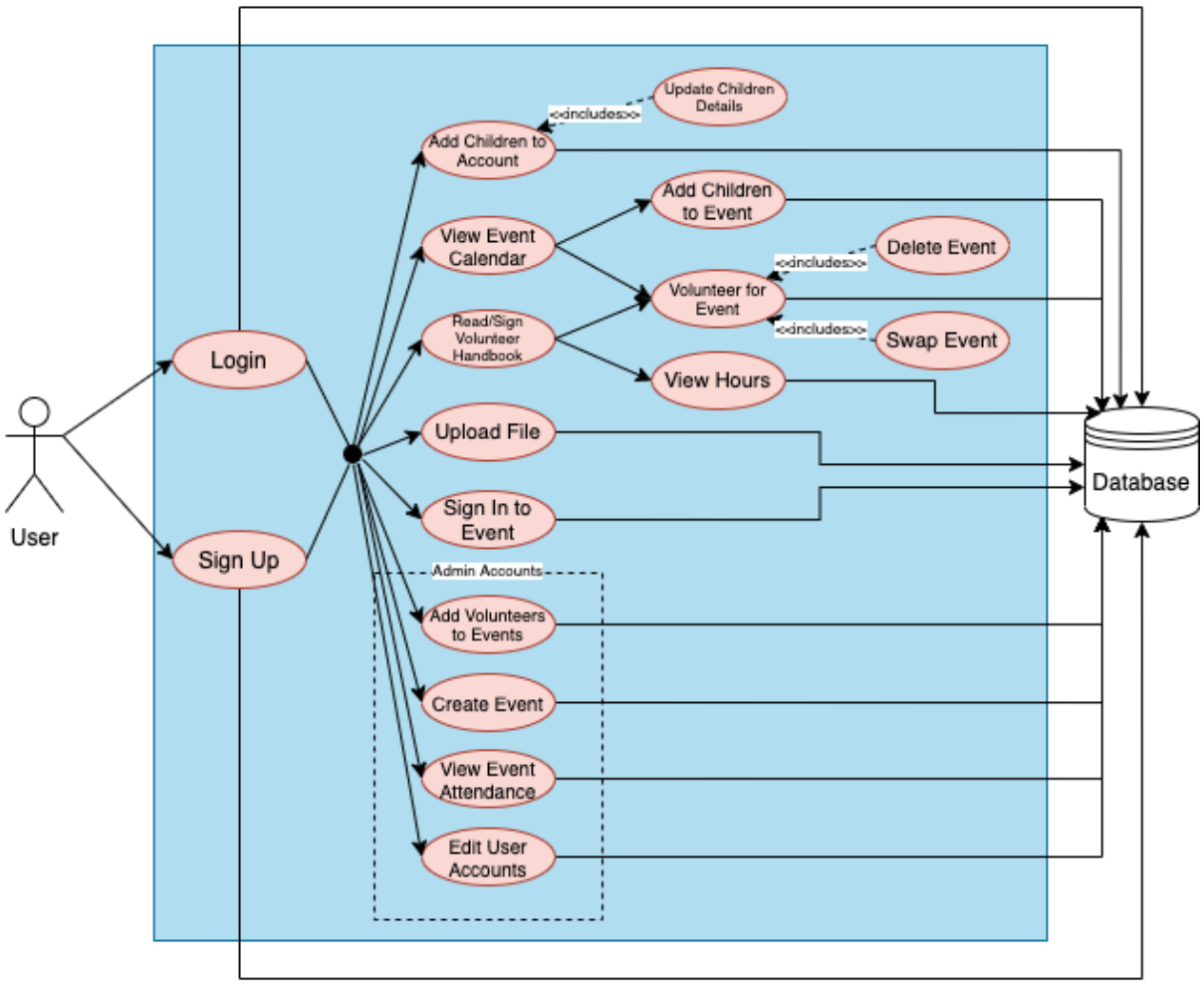


Figure 55. Use Case Diagram.

9. Development

9.1 Production

In readying the application for production, it is first important to maintain the requirements and specifications that were set by the previous Senior Design group along with the new observations that will be reflected into this iteration. Currently, hosting the web application utilizes Azure, which is a cloud computing service that allows configuration and management of applications for a user. Coupled with tools such as virtual machines, databases, and external account creation for the management of payment systems and one can see how the Orlando Math Circle web application came to fruition.

For this team, since the task decomposes into improving upon an already existing framework, the following section formats will list a description on the current services used for the web application, an explanation on how the previous senior design team used the systems, and the direction that the current team will follow to implement the solutions listed in the requirements.

9.1.1 Azure

Microsoft Azure, or *Azure*, is a cloud computing service that allows for building, testing, deployment, and managing of applications through the data service that has been storyboarded by Microsoft. With the combination of these features, it supports software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS) and is one of the most widely used cloud computing software for applications to date.

For computer services, Azure supports the usage of virtual machines; with Microsoft-backed software, users can start up virtual sessions of Linux and Windows, also supporting the use of imaging for sensitive software packages. On top of this, Azure has services for users looking to build websites, applications that can manage the cloud services, and can utilize special applications to configure script-based web jobs and processing that may need to take place behind the scenes for an application.

While Azure centers implementation of cloud computing through computers and machines, Azure also hosts solutions for authentication, identity, mobile, storage,

media, data, and messaging. Well-known services that are used from Azure include *Azure Active Directory,* which encapsulates common services such as single sign-on, consumer identity, and verification throughout the Azure infrastructure; the power of software such as this is that when services or virtual machines need to be utilized throughout, it eliminates the need for reauthentication for each service. *Azure Data Explorer* provides data-exploration and analysis for software that pipes data throughout, *Azure SQL Database* integrates with other Azure services to provide a SQL-backed database that works with the backend to streamline data.

Lastly, the *Microsoft Azure Service Bus* extends support to applications that have been configured either on Azure or off Azure architecture. Connecting the applications through a service-oriented architecture (SOA) allows the service bus to yield support for Event Hubs, which can be used to manage updating events in real-time (i.e., GPS location), Queues, which manage communications travelling in one direction (to the queue, from the queue), Topics, which work in the same fashion as queues except that each receiver of the message is granted a copy of the message as opposed to the original, and Relays, which allow for the message to be passed around, or in other words, supports bi-directional message movement.

Figure 56. A diagram that illustrates the one-directional movement of messages through a queue.

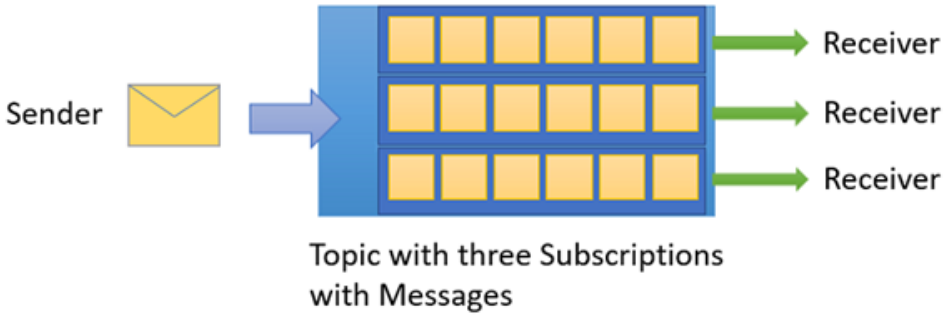


Figure 57. A diagram that shows the behavior of a Topic, distributing the same message as copies to receivers.

While the previous senior design team readied the web application to be hosted on DigitalOcean, the final project and documentation lists instructions for configurating the application to host on Azure. The design choice could have been made in part to the lightweight cost per month on the service and the ease of implementation throughout. Regardless, the previous team takes use of Azure's virtual machine service, using what is known as a *Standard_B2s,* consisting of 2 vCPU cores and 4GB of memory.

Upon further procurement of the documentation, it is noted that the service is hosted and configured to run directly on Ubuntu firmware, which suggests Linux support. The previous senior design group could have made this decision knowing that various mobile operating systems – except iOS – are backed by a lighter version of Unix, so by developing and hosting in Linux, it ensures that there is maximum support piped through the application. After configuration of the virtual machine, the design group then segues into Azure's data management sector through PostgreSQL and has the service run alongside Docker to handle data piped through the application from users. The documentation states to add a server using configuration that can parallel with Docker and to make sure that the virtual machine that was started utilizes the correct specifications to take advantage of the database.

With Azure, the final service that the previous senior design group utilizes is SendGrid: with SendGrid, the web application extends support for emailing and notifications for the users that hold relationships with OMC. Completed on the administrative side, the group states that for developers, an account will need to be created then managed through SendGrid. Once inside SendGrid, the final steps that end usage of Azure software is to authenticate the domain, connect it to the

OMC website to ensure that mail is sent from that domain, and to obtain DNS keys that SendGrid will use. After configuring these steps, the application would, at that point, be ready to be started.

In implementing the second iteration of the Orlando Math Circle application, the current group should first consider the steps that were taken to bring about the version of the application that is present now. For instance, the procedure that was utilized to set up the backend of the application, while compact and efficient due to its containment inside Azure, involves a lengthy process and attention to intricate detail before the application is even started. While we plan to keep the web application hosted in Azure for the same benefits the other senior design group discovered, a system will be put in place to streamline the setup process and make it more user-friendly. Ideas of approaching this include, but are not limited to:

- **Creating a script.** If we write logic and instructions that configure the files and databases intended for use, setting up the application could be as simple as executing a single program.

- **Blending the frontend and backend.** Currently, the application is configured in a way such that when testing in development mode, both the frontend and backend have to be initialized in a terminal. In doing this, steps are added, and configuration steps are duplicated for environment variables that the web application needs access to. If the group can discover a method in refactoring the logic such that when the application is used the frontend and backend start at the same time, that would reduce many of the steps that would be taken for configuration.

- **Exploring other Azure options.** Azure, as explained beforehand, is home to various development tools and services that help the user to get maximum efficiency out of their application. While the virtual machine service, PostgreSQL, and SendGrid are the only services that the OMC application is using, through further research there may be a better option that is offered through Azure that could accomplish goals set.

Regardless of what is chosen, one of the most important factors for the web application that already has a robust backend is to make it easier to use and be more accessible throughout.

One key factor that the current senior design group must achieve that the previous senior design group did not is deploying the application to be accessible to all users. In knowing this, easing developmental preparation becomes even more important because if the OMC web application is deployed, if developers are looking to improve the application, they need to be able to have access to the mode simplistically.

9.2 Environment Setup

This section will help you set up an environment on your system to begin development of the application, run the current application, as well as test the application. Most of the tools used in development is industry standard and It is what we recommend for development, but alternatives can be used. It is recommended to use a MacOS or Linux environment for development of this application. It is entirely possible but a little more involved if using windows.

9.2.1 NodeJS

This application uses a NodeJS server for both the front end and backend. NodeJS is used based on its widespread use and popularity thus It being well maintained. The large community of developers that uses It makes It easy to find resources and help when using the application. It will run on any major operating system. Though Windows uses might have a little more difficulty using some of the tools used in this application such as Git, Docker, and terminal compared to UNIX based operating systems the application is supported by all operating systems.

Installing on Linux or MacOS

1. Install or update NVM (Node Version Manager) typing either of the scripts below in the terminal.

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.37.2/install.sh | bash
```

```
wget -qO- https://raw.githubusercontent.com/nvm-sh/nvm/v0.37.2/install.sh | bash
```

2. Check if NVM is properly installed by entering help message in the terminal.

```
nvm --help
```

3. Install the most recent version of Node 14.X.X

```
nvm install 14
```

4. Check if installed.

```
node -v
```

Windows

1. Go to https://nodejs.org/en/download/ and download version '14.X' or higher for your respective operating system
2. To confirm installation, open a terminal, either Command prompt or PowerShell, and enter:

```
node -v
```

9.2.2 GIT

For this application we are using Git for version control. GitHub is the service we are using to host our Git repositories. There are some Graphical user interfaces to use Git such as GitHub desktop and Git Kraken. But for this project we are using the terminal for our git commands. When developing though Visual Studio you can use its' integrated tools to make your GIT operations seamlessly while developing. You can do a lot of the basic GIT commands such as pull, push, commit, sync changes, change branches, and compare changes made on branches right in visual studio code.

Windows

1. Go to https://git-scm.com/download/win to download GIT
2. Open Command Prompt or PowerShell and enter:

```
git --version
```

Linux or Mac

Linux based operating systems come with GIT pre-installed. Check the version of GIT by opening terminal and entering:

```
git --version
```

9.2.3 Visual Studio Code

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js. Other text editors may be used but this is industry stand and typically is a good fit for projects such as this one. It is also highly recommended that you also install all of the following extensions as well. These extensions enforce a consistent coding style that the IDE will enforce through linting. A team of developers can benefit from this to ensure that there is consistent coding style throughout the application making It easier to read for future developers to be onboarded. VsCode's linting, intellisense, and Veteur provide the best coding experience when coding in VueJs.

Download

1. Download Visual Studio Code on all operating systems by visiting https://code.visualstudio.com/Download
2. Click the boxes on the left side of the page to open the extensions menu install the following extensions:
3. ESLint is the linter used by the project managed by .eslintrc.js files found within the repository. This extension will ensure stylistic consistency and prevent minor programming errors and common mistakes.
4. Vetur is required for VSCode to understand Vue files and properly apply intellisense and is managed by the vetur.config.js file.
5. This extension is less mature than the others and I occasionally have issues with it. You can attempt to restart the extension in the command palette (Ctrl + Shift + P) and selecting >Vetur: Restart VLS (Vue Language Server). If that doesn't help you can reload all extensions in the command palette by finding >Reload Window.
6. Prettier is an opinionated code formatter that will help keep your code clean and is managed by .prettierrc files found in the repository. It can be activated

by entering Shift + Alt + F on the keyboard, by right-clicking a file and selecting Format Document.

7. For streamlined usage you can configure VSCode to format on save by enabling this setting found by entering Ctrl + Shift + P to open the command palette, search for >Preferences: Open Settings (UI) and enabling Format on Save.
8. Remote - WSL is only required if using WSL as your development environment. It will run the internals of VSCode itself in a Linux environment. This allows you to use run and debug applications inside Linux using VSCode and use a Linux Git installation.
9. (Optional) DotENV provides syntax highlighting for the .env files.
10. (Optional) Project Manager allows you to save different open folders as projects and lets you easily switch between them.
11. (Optional) Remote - SSH is a similar extension to Remote - WSL but instead of WSL it will run VSCode through SSH. This allows you to open editors on a remote machine and edit files completely remotely. An example would be editing the .env files on the production machine without having to use a terminal-based IDE.
12. (Optional) npm integrates npm commands and tasks into the command palette so you don't have to type them out all of the time.
13. (Optional) Docker is a utility for quickly enabling and disabling different docker environments.

9.2.4 Insomnia

When working with an API you need a means of testing the endpoints. This is very similar to the popular tool “Postman”. Used for interacting with APIs. If you are comfortable with another tool for testing APIs that will suffice. To install Insomnia, visit the Insomnia website and download Insomnia Core at https://insomnia.rest/.

9.2.5 Docker

Docker was not a need when creating the application, but it saves us from running into some issues in the development process. Having the project in a docker container solves multiple problems including but not limited to:

- Missing or incorrect application dependencies
- Conflicts of multiple programs fighting over the same resources.
- Limited memory the application can use.

- Missing or incomplete scripts to start, stop, and uninstall the application.

Windows

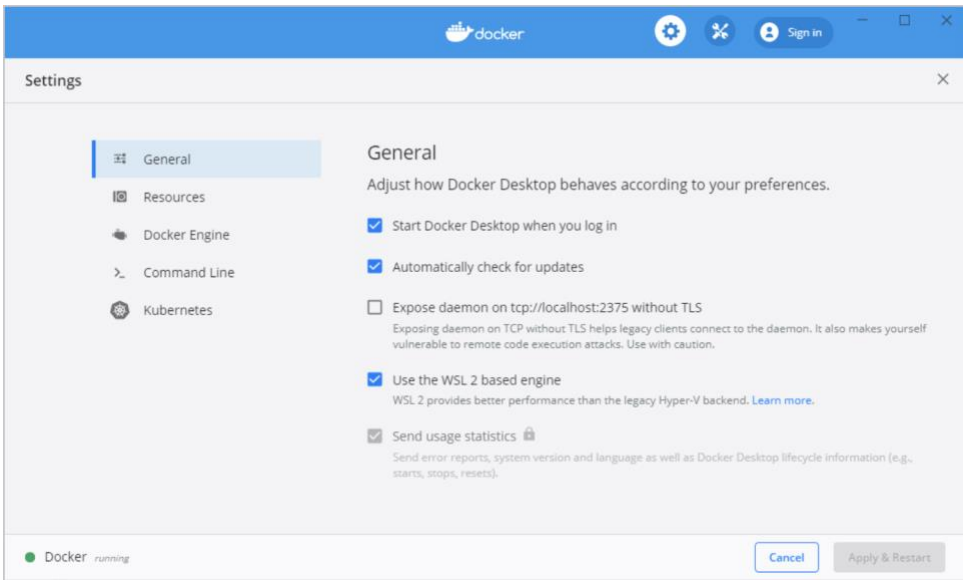
In order to install docker on windows you first need to find out what version of windows you have. To find out your windows version [29]:

1. Select Start > Settings > System > about
2. Under Device Specifications > System type see if your running 32 or 64 bit
3. Under Windows Specifications see what windows edition you are running

Docker has different installation methods depending on the version of windows you have. For all versions, first download docker for desktop from the docker website and follow the instructions [26].

If using WSL or Windows Home:

1. When following the instructions when installing docker for desktop, enable WSL 2 when prompted.
2. Start docker
3. In Docker select Settings > General



4. Select the WSL 2 engine check box
5. Click Apply & restart
6. To develop with docker remotely in WSL open VSCode using the WSL extension first type: *wsl* and then type code.

If using Windows 10 enterprise, education or pro:

1. Run the docker installer.
2. Enable Hyper-V Windows Features when prompted.
3. Proceed with installation.
4. (Optional) if not on the admin account you must add the current user to the docker-user group. Run Computer management as administrator then select Local users and groups > groups > docker-users and right click the user to add to the group.

Linux

If you are a Mac user you can install Docker Desktop on Mac. Linux users can find their own version of Linux and install the Docker Engine. Docker Compose, the tool that interacts with the docker-compose.yml file in the repository root, does not come with Docker Engine, so also Install Docker Compose. There are unnecessary performance implications if you try to install Docker Engine through WSL, use Docker Desktop instead as it will interface with WSL on its own.

Verify Installation

You can verify if Docker is properly installed on your system after following an installation guide with the following commands in your terminal and receiving back versioning information.

```
docker --version
docker-compose --version
```

9.2.6 Data Grip

While MikroORM will be used as an ORM (Object Relational Mapping) to translate the database tables into JavaScript objects for easier management, some tasks are still best performed more directly. Even if this is not the case, running queries and seeing what the result is in the database can be a productive diagnostic tool. The recommended graphical client is JetBrains DataGrip. DataGrip is not normally free software but students can apply to use it for free at the JetBrains Products for Learning page.

9.2.7 Services

The OMC application rely on three external APIs and services PayPal, Twitter, and SendGrid. Having proper access to these APIs and having the necessary credentials is essential for the application to function properly.

9.2.7.1 PayPal

The application will not start unless the paypal service is properly configured. If running the application in development mode using the paypal sand box you can use a personal pay pal account. In production a business account is required to process real transactions, the account owner receives the payment.

1. Go to https://developer.paypal.com/home/ to get credentials
2. Click “Log in to dashboard” at the top right of the page.
3. Sign in if you have an account or sign up for an account.
4. Select sandbox mode if the purpose is testing and development. Select live mode in production to process live transactions.
5. Click “Create App” and a business account will be created for you. In production use the Orlando Math Circle business account.
6. You will receive a “Client ID” and “secret” to configure the application. (Explained in section 9.3)

9.2.7.2 Twitter

Registering for the twitter API service is a little more tedious then then the other services. Twitter requires developers to apply for access to the API and detail what the application they are using will be used for. You should be given the twitter credentials from the previous team working on this project. In the initial request It was detailed that the twitter API is being used to exclusively look up tweets made from the Orlando math circle twitter account. No tweets are meant to be made from the API, the tweets will show up on the application home page and will link to twitter when clicked. There may be extra steps to be made is the twitter API Is used for any other reasons.

We are approved to pull 50,000 tweets monthly from the API which should suffice for our use case. In order to make sure this is adequate we cache the tweets every minute so if multiple users log in within the same minute, they will not be seeing new tweets until the cache refreshes. After your request is approved an application

is created for your use case. You will get an API key and secret, be sure to save these because twitter will not display It again. You will then have to generate new credentials making the old ones invalid.

9.2.7.3 SendGrid

Follow the directions in section 9.1 to create a sendgrid account in azure. Follow that guide on how to authenticate an email address to send emails from.
Enable the Sandbox mode for emailing provided in the application. When SENDGRID_SANDBOXED is set to true in the ./packages/server/.env file emails are printed to the console instead of actually being sent.

9.3 Application Installation

In order to start development of the Orlando Math Circle application, you are required to first setup up the necessary environment in section 9.1.

Downloading and Installing
Download the repository using Git. This will make a folder called omc, though you may change it.

```
git clone git@github.com:duckies/omc-app.git omc
```

Install the frontend dependencies.

```
cd ./omc/frontend && npm install
```

Install the backend dependencies. (Assuming you're in the frontend directory)

```
cd ../backend && npm install
```

Copy the default avatars and event picture to the static directory. (Assuming you're in the backend directory)

```
cp -R ../resources/images ../../uploads
```

9.4 Configuration

Details on the three external services PayPal, SendGrid, and Twitter need to be configured to initialize the application in a development mode. Acquire access to the credentials for each and save them as described in the following configuration section. More details on these services can be found in section 9.1.8. Keep in mind that without proper configuration and setup of the services the application will not work as intended.

There should be two .env files in the project, ./omc/backend/.env and .omc/frontend/.env assuming omc is the name of the root project folder. Environment variables are variables in the form of key value pairs that allows software programs to know what folder to store temporary files and where to find user settings. The image below is an example of the environment variables formatting.

```
PORT=3000
FRONTEND_URL=http://localhost:800
```

The following table is for the environmental variables in the frontend.

Key	Required	Default	Description
PAYPAL_CLIENT_ID	FALSE	sb	The Client ID provided by the PayPal developer application page to configure the PayPal Smart Buttons.
STATIC_BASE	FALSE	http://localhost:3000	The root of the fully-qualified URL for where images and uploads are found. No ending slash.

AVATAR_BASE	FALSE	/defaults/avatars	Relative to the STATIC_BASE, contains the location of the default avatars. No ending slash.

Table 2. Frontend environmental variables.

The following table is for the environmental variables in the backend.

Key	Required	Default	Description
PORT	FALSE	3000	The port the backend operates on. This does not need to be changed unless you have a specific reason.
SECRET	TRUE		A random string of characters and numbers that is used to encrypt the auth tokens. Changing this will invalidate all tokens.
DATABASE_NAME	FALSE	omc	The name of the database to connect to.
DATABASE_HOST	FALSE	127.0.0.1	The host or IP of the PostgreSQL server.
DATABASE_USER	FALSE	postgres	The username with access to the DATABASE_NAME database.
DATABASE_PASS	FALSE	postgres	The password to the

			DATABASE_USER username.
DATABASE_SSL	FALSE	false	Instructs the PostgreSQL connection driver to connect using SSL encryption on remote database connections.
DATABASE_CACHE	FALSE	false	If enabled, subsequent restarts will be slightly faster. Not really worth enabling.
DATABASE_DEBUG	FALSE	false	Prints all MikroORM SQL queries to the console for debugging purposes.
FRONTEND_URL	FALSE	http://localhost:8080	The URL where the frontend is located for links in emails. No ending slash.
ADMIN_EMAIL	FALSE		Specifies an email that, if seen during registration, will promote the user to admin. Can be removed once used.
PAYPAL_SANDBOXES	FALSE	true	Toggles between the PayPal sandbox and production environments.
PAYPAL_CLIENT_ID	TRUE		The Client ID provided by the

			PayPal developer application as part of an OAuth2 token grant.
PAYPAL_SECRET_KEY	TRUE		The Secret provided by the PayPal developer application as part of an OAuth2 token grant.
TWITTER_KEY	TRUE		The Key provided by the Twitter developer application as part of an OAuth2 token grant.
TOKEN_SECRET	TRUE		The Secret provided by the Twitter developer application as part of an OAuth2 token grant.
FILE_DIRECTORY	FALSE	../../uploads	The directory for storing uploaded files relative to the backend folder.
FORM_SUBDIRECTORY	FALSE	form	The name of the folder that is created inside the FILE_DIRECTORY for storing uploaded reduced lunch forms.
SENDGRID_IN_DEV	FALSE	false	If true emails are printed to the console instead of being sent through SendGrid.

SENDGRIND_API_KEY	TRUE		The API key with full permissions provided by SendGrid.
SERVE_STATIC	FALSE	true	If enabled, then static files found in the FILE_DIRECTORY will be hosted by the backend.
DEFAULT_EVENT_PICTURE	FALSE	/defaults/neon-math.jpg	The path to the file used relative to FILE_DIRECTORY when an event lacks a picture or project picture.
DEFAULT_AVATAR_FOLDER	FALSE	/defaults/avatars	The path relative to FILE_DIRECTORY where default avatars are located.

Table 3. Backend environmental variables.

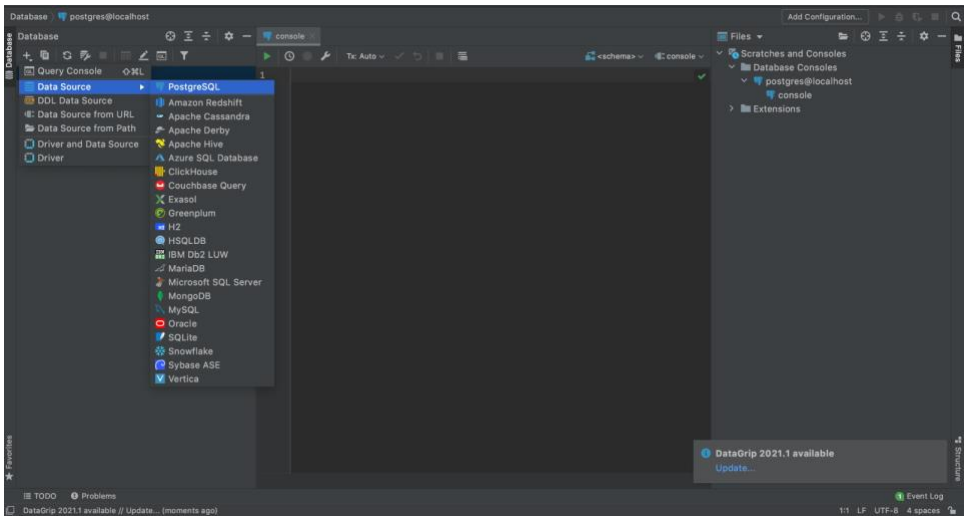
9.5 Database Setup

The application requires a single PostgreSQL database to operate. If you are using Docker continue along with this section, otherwise fill in *DATABASE_** options found on the configuration page in the *./omc/backend/.env* file as appropriate. Go to the root directory of your project file, It should contain a docker-compose.yml file.

1. Run the following command then wait for docker to initialize the PostgreSQL database.

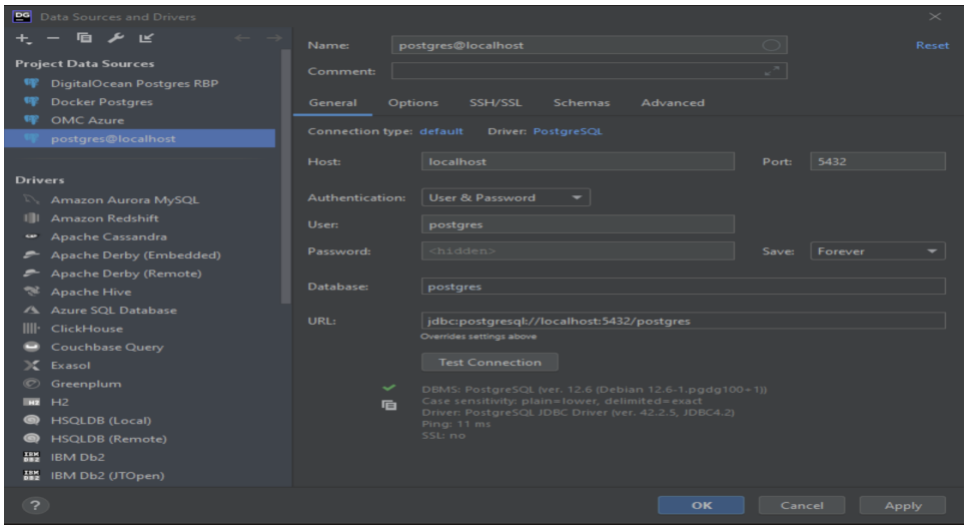
```
docker-compose up -d
```

2. Open DataGrip and click on the + sign located on the top right then go to Data Source and select PostgreSQL and enter the following information to establish a connection.

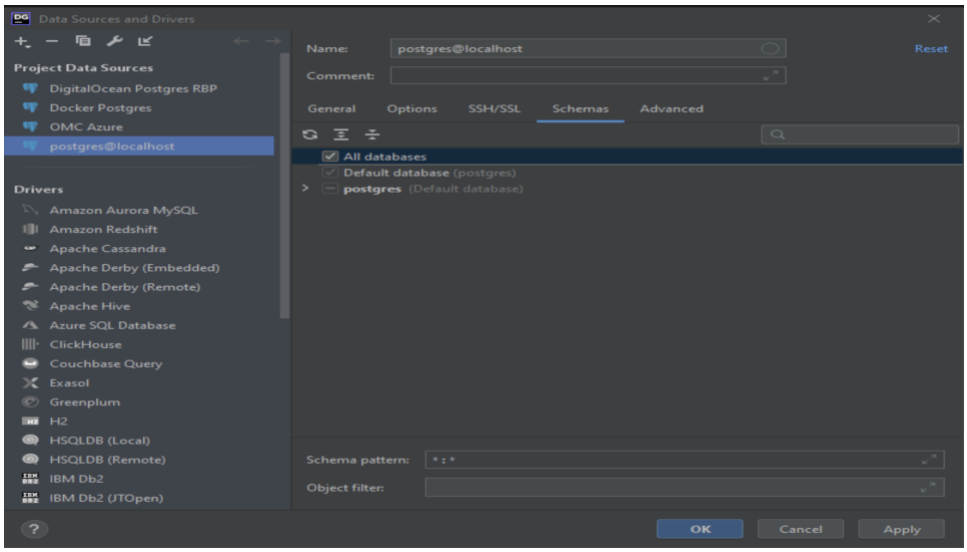


Option	Value
Host	localhost
Port	5432
Authentication	User & Password
User	postgres
Password	postgres
Database	postgres

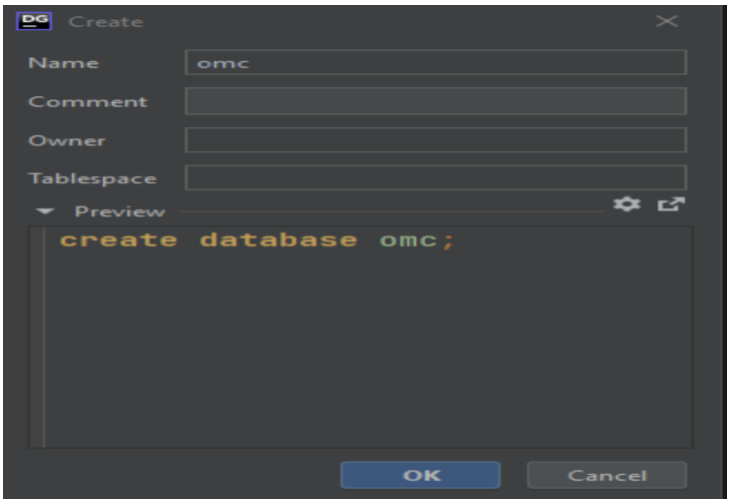
3. Click on Test Connection and a green checkmark should indicate the connection was successful.



4. Navigate to the Schemas tab on the toolbar above Host and click on All databases.



5. Name the connection anything you would like, then hit OK to save and close it.



6. On the left sidebar expand the connection you created and right click on databases and in the New submenu select New Database.
7. Name the database omc then lick OK and exit DataGrip once the new database is created. You can rename the database, say for other testing environments, but the DATABASE_NAME environment variable will need to be configured.
8. Navigate to the backend folder and run the following command to test the connection and create the database schema.

   If using docker:

```
npx mikro-orm schema:create -r
```

   If using a remote external database (azure):

```
npx mikro-orm schema:create -r --fk-checks
```

9.6 Initialization

After the application and all the dependencies have been installed, the database can be started.

While in the backend directory run the following command.

```
npm run start:dev
```

Navigate to the frontend directory and run the following command.

```
npm run dev
```

The backend server is available on http://localhost:3000 and frontend is available at http://localhost:8080.

9.7 Evaluation and High Schooler Involvement

Our team plans on working directly with our clients to receive feedback on the application's functional features and UI design to provide a final product that goes beyond just satisfying the initial requirements. Usability and accessibility are two of the most important factors to consider during development and therefore usability testing will be an essential process in the summer months. However, consistent feedback from the high school volunteers will be the most substantial form of evaluation throughout the course of both semesters. As the core functionalities of the application are already in place, our focus will be redirected to fixing the most trivial of bugs and UI inconsistencies to further optimize user experience.

9.7.1 High Schooler Involvement

We will be working closely with the high school volunteers to not only receive input on the application's design and functionality but also teach them more about the application's inner workings and frontend technologies as we implement new features. A member of the previous senior design team (John Gilbert) has set up extensive documentation regarding the application's installation, production,

relevant tools, and important research notes. He has also included recorded tutorials on some of the technologies used within in the application. We plan on utilizing this documentation to familiarize high schoolers with the application's framework and eventually build on it during development.

As one of our core requirements is to implement a method for volunteers to swap events with one another, it is essential to receive feedback from the high schoolers throughout development and design mockups as they will be the direct users of this feature. Similarly, one of our stretch goals is to log volunteer hours in the database and calculate awards for students based on the number of hours they collect. We plan on brainstorming with the high schoolers to come up with unique badges and awards that would further motivate the volunteers to sign up for more events and in turn increase involvement in the organization.

9.7.2 Methods of Collaboration

Due to the COVID-19 pandemic, the team will hold online Zoom sessions with the high school volunteers, which will be recorded and monitored to comply with Orlando Math Circle's Code of Conduct, who wish to participate in the development process. Every team member was provided with a Gmail account through the organization and all forms of direct communication outside of the Zoom sessions will occur through these emails.

Additionally, the high schoolers will be invited to a shared Discord server between the development team and OMC sponsors to allow for quick and easy communication between all parties. The use of organized channels will allow the students to ask questions, give instant feedback on UI issues, and share useful resources they encounter surrounding the application's technologies if desired.

Participating students will be added to the OMC Trello Board to gain familiarity with professional collaboration tools and be provided with a convenient method to view tasks surrounding the project. On top of being assigned smaller-scale tasks through the board, the students will be able to report bugs and UI inconsistencies through a designated Trello card as seen in Figure 58.

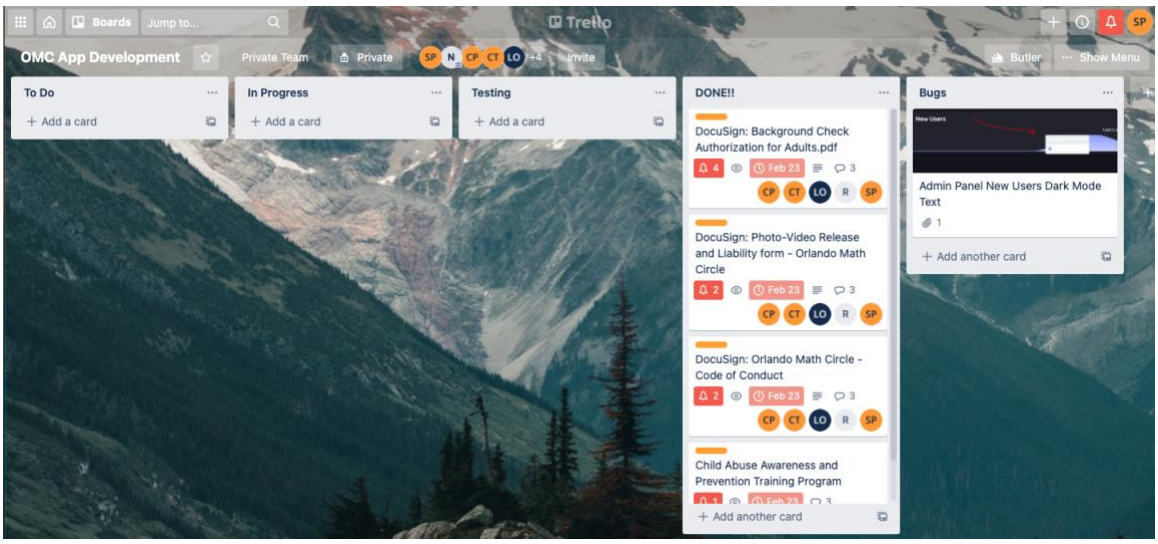


Figure 58. Trello Board.

9.7.3 GitHub Pages

GitHub Pages is a static site hosting service that takes HTML, CSS, and JavaScript files straight from a GitHub repository and publishes a website. By default, GitHub Pages sites will be hosted on GitHub's github.io but custom domains can be used if desired. Orlando Math Circle has created a GitHub account as an organization which will be used for the project's repository and the respective GitHub Pages site used for extensive documentation will be available at

`http(s)://<organization>.github.io.`

The webpage generated through GitHub Pages will be the primary form of documentation used to provide instructions on the application's development and production process for developers who will continue to enhance the app after our project has concluded in the summer. It will provide information about both senior design teams that worked on the app, tutorials and exercises on some of the technologies used within the app, and thorough instructions on the installation process needed for future development. The documentation will be technical but comprehensive enough to where the high schoolers can successfully download and work with the necessary technologies regardless of their current level of familiarity with them. Figure 59 displays the current documentation that previous team member John Gilbert has set up that the current team plans to continue contributing to.

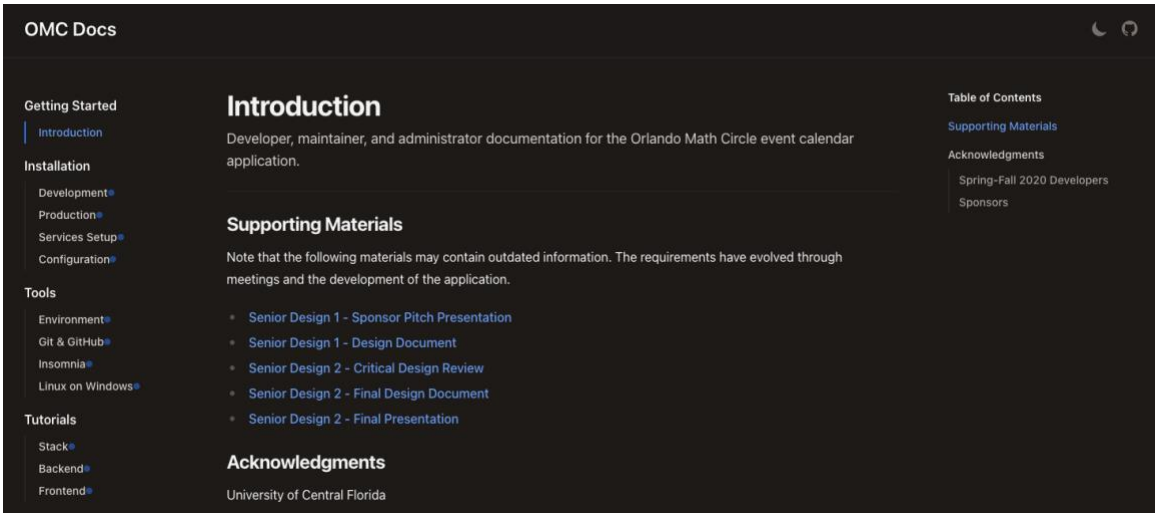


Figure 59. Current documentation hosted on GitHub pages as developed by previous team member John Gilbert.

GitHub Pages recommends configuring Jekyll, a static site generator, into GitHub Pages sites. Jekyll combines webpage templates with site-specific markdown files to create dynamic components for websites. Additionally. GitHub Pages comes with a theme chooser feature through Jekyll that can be configured and modified in the repository's settings.

In the event that developers continue building onto the documentation in the future, the usage limits for GitHub Pages are noted as follows:

- GitHub Pages source repositories have a recommended 1GB limit.
- Published GitHub Pages sites may not exceed 1GB in size.
- GitHub Pages sites hold a soft bandwidth limit of 100GB per month.
- GitHub Pages sites hold a soft limit of 10 builds per hour.

9.7.4 Markdown

Markdown is a lightweight markup language for creating websites, notes, presentations, technical documentation, and more. It is a formatting syntax designed to convert plain-text to structurally valid HTML and is widely used due to its easy-to-read and easy-to-write structure. In the event the high school students continue the OMC documentation, Markdown is a simplistic language that will take little to no time for the students to become familiar with.

The following table showcases Markdown's basic and extended syntax along with its respective HTML equivalent [17]. Note that all Markdown applications will support basic syntax elements but not all applications will support extended syntax elements. The image element in the following figure marks the end of Markdown's basic syntax.

Markdown Syntax	**HTML**

`# H1`
`## H2`
`### H3`
…
`###### H6`
Creates a heading where the number of number signs corresponds to the appropriate heading level.
	`<h1>H1</h1>`
	`<h2>H2</h2>`
	`<h3>H3</h3>`
	…
	`<h6>H6</h6>`

`**bold**` Bolds a word. Double underscores can be used as an alternative, but not to bold the middle of a word.	`<strong>bold</strong>`
`*italicized*` Italicizes a word. Single underscores can be used as an alternative, but not to italicize the middle of a word.	`<em>italicized</em>`
`***bold and italicized***` Bolds and italicizes a word. Triple underscores can be used as an alternative, but not to bold and italicize the middle of a word.	`<strong><em>bold and italicized</em></strong>`
`> blockquote` Creates an indented section. For multiple paragraphs, add a > on lines between.	`<blockquote>blockquote </blockquote>`

`1. First`
`2. Second`
`3. Third`
Creates an ordered list. Lists only need to start with the number one and will render in
	`<ol><li>First</li>`
	`<li>Second</li>`
	`<li>Third</li></ol>`

numerical order regardless of what numbers follow.	
- First - Second - Third Creates an unordered list. Asterisks or plus signs can be used as an alternative.	- First - Second - Third
`code` Denotes a word or phrase as code.	<code>code</code>
[title](url) Creates a link. Tooltip text may be added when enclosed in quotes and placed before the closing parenthesis.	title
![alt text](image.jpg) Renders an image. An image title may be added when enclosed in quotes and placed before the closing parenthesis.	<img src=”image.jpg” alt=”alt text”>
| Title | Description | | -------- | ----------- | | Element | Text | Creates a table using hyphens and pipes. Markdown offers a table generator for simplicity due to its tedious nature.	<table> <tr><th>Title</th> <th>Description</th></tr> <tr><td>Element</td> <td>Text</td></tr> </table>
EX [^1] [^1]: Here is the footnote. Creates a superscript footnote link and reference using carets and identifiers.	EX [1] [1]: Here is the footnote.
# Heading {#custom-id} Creates a heading with a custom ID to allow for CSS styling modifications.	<h1 id=”custom-id”> Heading</h1>
~Strikethrough text~ Renders text with a horizontal line through the center of it.	<s>Strikethrough text </s>
- [x] Task 1 - [ ] Task 2 - [ ] Task 3 Creates a task list that renders a list of items with respective checkboxes.	There is no direct implementation for task lists in HTML.

Markdown also supports the implementation of emojis through text shortcuts, which may vary from one Markdown application to another and therefore developers should check the most recent documentation to prevent issues during the compilation process. Suitably utilizing emojis in the OMC documentation and curriculum plan can make the material more engaging for the high school students and more visually appealing to follow along.

9.7.5 Curriculum Plan

Orlando Math Circle holds hands-on internship style workshops in the summer for the high school students, an example being a week dedicated for the students to read tutorials and complete exercises in Java programming. As OMC has stated, one of their biggest requirements is making the application's framework extensible for the students to continue working with the app after our development team has graduated. While our team could focus on containerizing the application and improving modularity, we discovered it is equally as important to ensure the students are comfortable with the relevant technologies before tackling the code.

We have chosen to break up the curriculum plan into two phases, both of which will be accessible under the GitHub Pages documentation. Phase 1, the 'Learning' phase, will be dedicated to supplying resources, tutorials, and any necessary background information on the application's technologies for the students. While our sponsors have informed us that the high school students are exceptionally intelligent for their ages, each student comes with a different programming background. Our team will create a survey through Google Forms designed to determine the level of familiarity the students have with standard web development technologies and the frameworks the application uses. The survey results will then shape the curriculum plan our team implements.

If a student has no web development background, we will include resources on the basics of web and application development, along with an introduction to HTML, CSS, and JavaScript. If most students have minimal experience that only goes as far as HTML/CSS, we will provide additional online exercises to help them master the languages before overwhelming them with information on web stacks and the advanced languages involved with application development. The largest area of content within this phase will be on resources surrounding TypeScript, PostgreSQL, Node.JS/NestJS, and Vue.Js/NuxtJS. The documentation will include tutorials and helpful tips on GitHub and Visual Studio Code as well, regardless of the students' level of familiarity with the two. The students will spend

the late spring and early summer familiarizing themselves with the relevant languages to prepare for Phase 2.

Phase 2, the ‘Applying’ phase, will be a set of hands-on exercises meant to be completed from directly within the application’s code. Prior to Phase 2, the students will have spent an ample amount of time becoming familiar with the technologies the OMC application uses. This workshop is dedicated to allowing the students to become familiar with the code itself and learn how to easily navigate the structure and directories within. Figure 60 displays a visual of the summarized initial curriculum presented to our sponsors.

Figure 60. Initial high school curriculum plan proposal.

9.7.6 Experiential Resources

Before the high school students are provided with the code to the application, they will spend time becoming accustomed to the relevant languages the app utilizes. However, not all learning can be achieved through watching video tutorials and reading documentation alone, especially when it comes to coding. Some students may be kinesthetic learners and naturally gain more knowledge through hands-on activities over lecture styled demonstrations.

Regardless of the type of learner each student is, unlike other disciplines, programming has little to do with memorizing and everything to do with hands-on experience. Keeping this in mind, the team decided to research and compile a list of online code editors and integrated development environments (IDEs) that are free to the students. Online editors come with great convenience as no servers are needed to run for developers to view their rendered code.

The following resources will allow the high schoolers to not only gain direct experience with the languages but open the door for the students' creativity to flow. One goal we have is for the students to come up with their own innovative ideas or suggestions regarding the OMC application itself after becoming more aware of the capabilities of each language by using these resources.

- ***JSFiddle.net*** – JSFiddle is one of the most popular online IDEs that allows you to edit and run HTML, CSS, and JavaScript code all at once. Its UI features four split planes by default, one for each of the languages and one for the rendered result. As JavaScript is the language that enables the functionality behind web components, developers can directly interact with functional features in the 'Results' panel upon render. The IDE allows developers to adjust the individual languages as well. HTML can be changed to HAML, an abstraction markup language for HTML designed to make HTML code cleaner through the elimination of inline code. CSS can be changed to SCSS, a more expressive subset of CSS, or SASS, the preprocessor scripting language that allows you to use advanced features not offered in traditional CSS. For more advanced usage, the IDE allows users to change the JavaScript panel to a variety of other languages, including TypeScript, Babel + JSX, React, and even Vue, on top of a wide collection of frameworks and extensions.

A key feature of JSFiddle is its collaborative environment. The playground allows developers to set up their avatar and name before providing them with a link to their session to share with teammates or friends. The collaboration dock shows users in the current session, a chat message board for users to communicate as they code, and offers the opportunity for audio chatting by enabling browser supported microphones. JSFiddle offers extensive settings to customize the IDE's layout as well as the opportunity to save session fiddles.

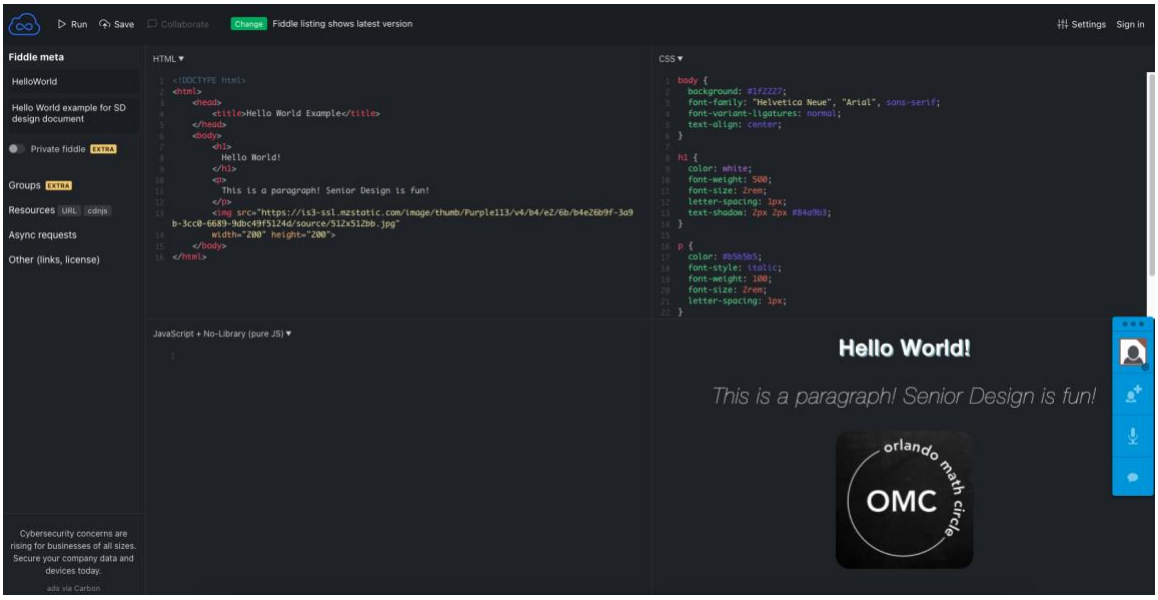


Figure 61. JSFiddle user interface with the collaboration dock in view. The IDE does not provide any sample code upon first visit. The code shown in the figure is a simple Hello World example created by team member Samantha Perez.

- ***Dillinger.io*** – Dillinger is an online Markdown editor that renders Markdown directly into the web browser. Its split pane design allows developers to view a side-by-side of their Markdown text file and its rendered formatted output, making it more convenient for developers to adjust text accordingly and preview updates within the same screen. Since Dillinger is cloud-enabled, users can save documents created during sessions and have the option to both import and export documents to other cloud services like GitHub, Google Drive, and Microsoft OneDrive.

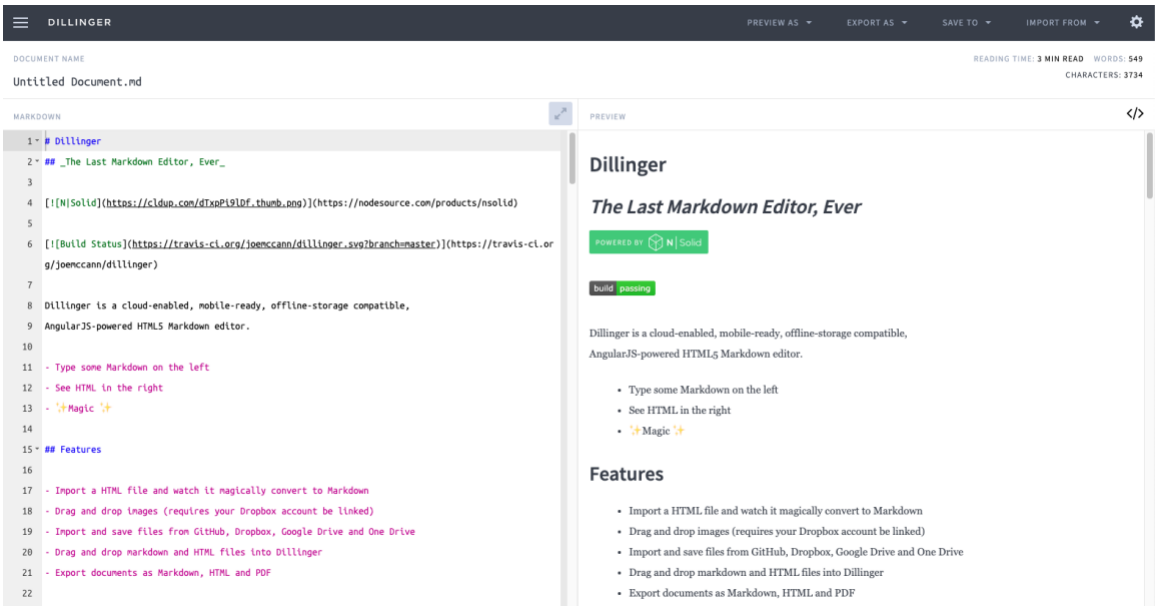


Figure 62. Dillinger user interface. A default Markdown file is loaded when Dillinger is first opened that provides basic instructions about Markdown syntax and basic editor usage.

- ***Katacoda.com*** – Katacoda is an onlne Node.js playground that consists of a remote version of Visual Studio Code and an Interactive Bash Terminal for Node.js servers to be run through command execution. Because the editor provides the same functionality that can be achieved in VS Code, Katacoda will likely only be useful for beginners and for testing quick server instances.

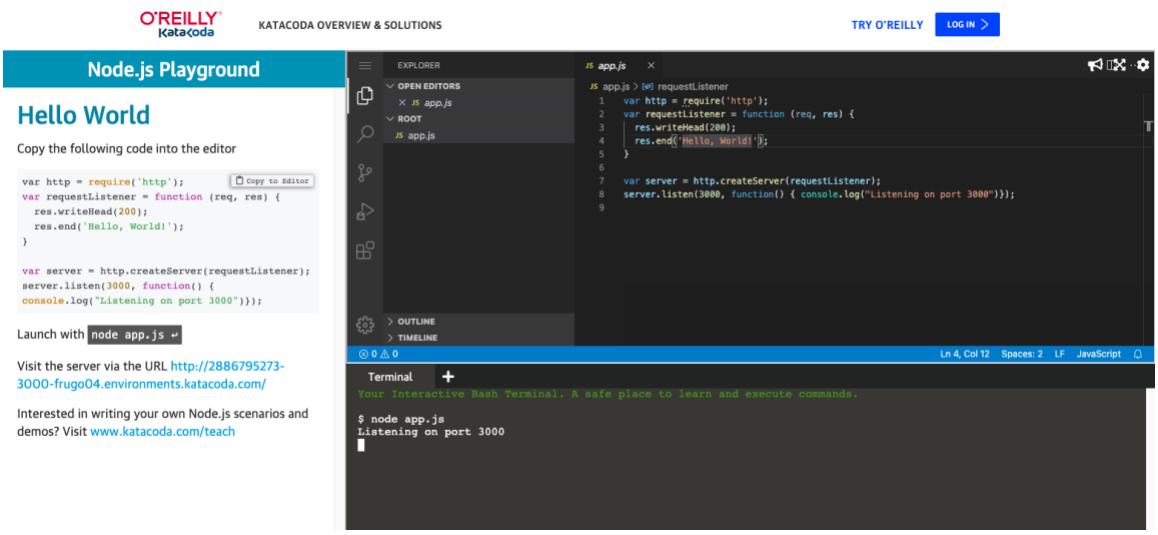


Figure 63. Katacoda Node.Js Playground user interface. A default 'Hello World' code snippet and launch instructions are provided when the editor is first loaded, allowing users to test and run the example server supplied by the editor.

- ***CodeSandbox.io*** – CodeSandox is an online prototyping tool and open-source IDE designed for rapid web development. It is compatible with a variety of languages, allowing developers to create static sites, full-stack web applications, or components on any device that uses a web browser [16]. CodeSandbox is another collaborative IDE that allows multiple developers to edit code at the same time through session link sharing. It automates dependency management and bundling, which assists with a quicker development process.

  A key feature to CodeSandbox is its real-time updating system. Developers can view live updates as they type, eliminating the need to monotonously save and run code even after the most minor of changes. Its user interface is extensive but easy to navigate. Because CodeSandbox offers a fundamental template for a Vue starter project, this IDE is a great choice for the high school students to experiment with.

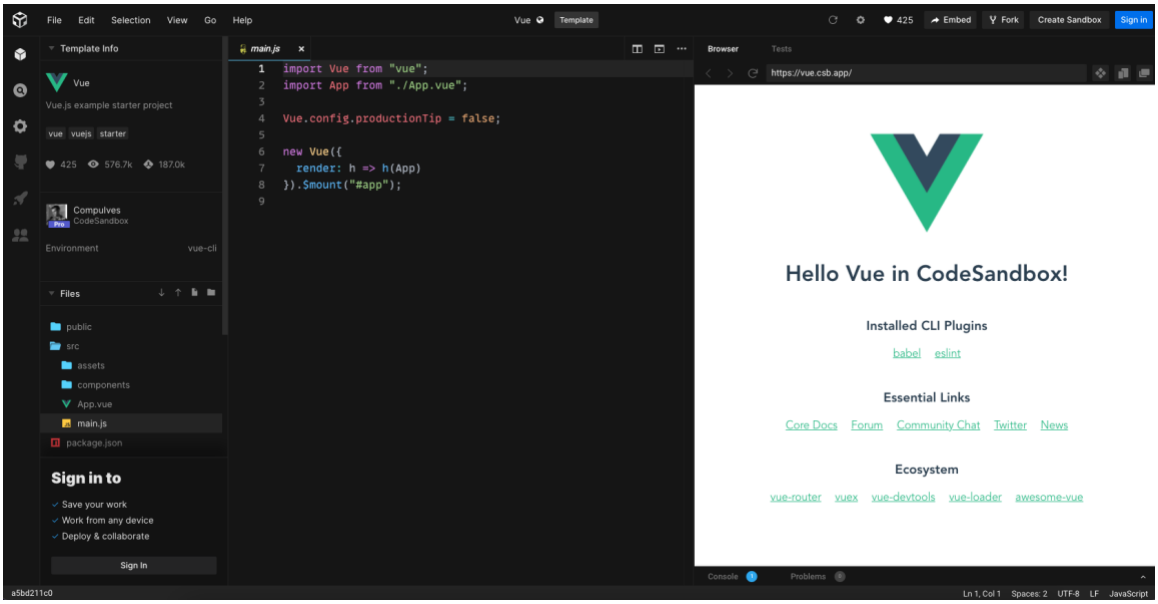


Figure 64. CodeSandbox user interface under its Vue environment and starter project.

- ***Codepen.io*** – CodePen is another open-source IDE primarily used as a front-end web development environment. It is similar to JSFiddle as it consists of separate side-by-side panels for HTML, CSS, and JavaScript code to be written and run. CodePen is widely popular due to the large collaborative community that stands behind it. There is a substantial amount of open-source code snippets generated by CodePen, called pens, that developers can fork over and continue building on. Pens can be embedded onto other websites, which will be seen quite often throughout the online tutorials provided to the students. These pens are often archetypes of the creativity that can be achieved in front end development. We hope the usage of CodePen guides the students into exploring the abundance of pens in existence to spark their own creativity and transfer their ideas into the OMC application.

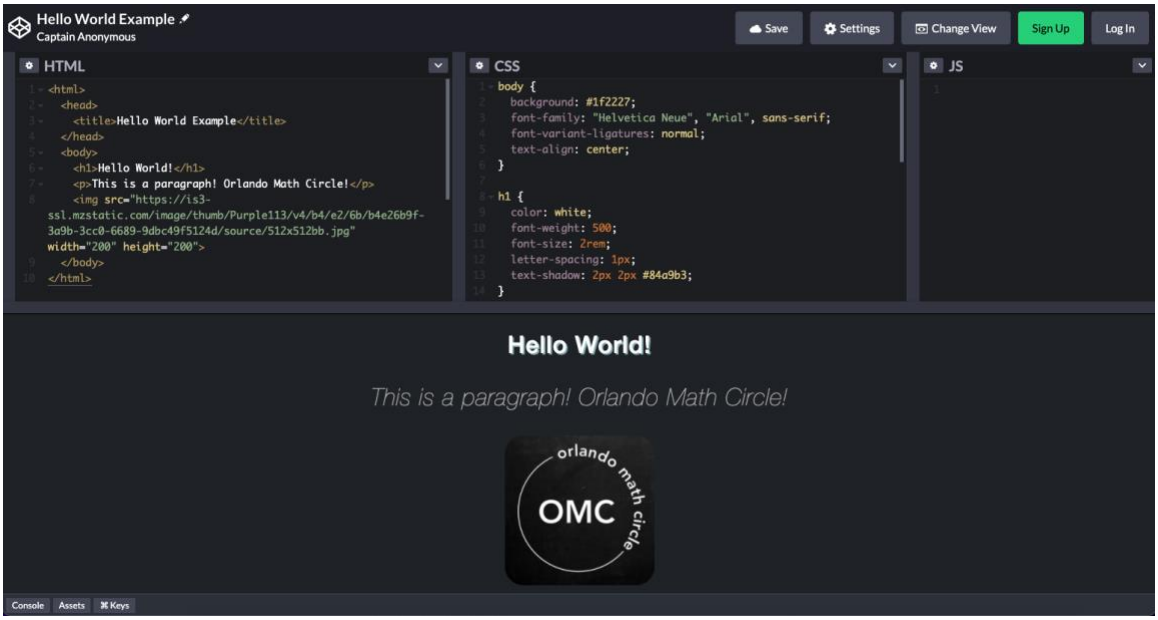


Figure 65. CodePen user interface. Since the IDE opens with an empty template, the 'Hello World' example displayed is almost identical to Figure 61's.

Even after the students are introduced to the application's source code, these online editors can remain beneficial for testing out and running smaller scale code snippets before adding them directly to the project. The main directory of the app can be overwhelming, and it is advised that smaller features are run through the IDEs listed above to prevent unexpected errors in the project.

10. Budget and Financing

As the purpose of our project is to improve the features of the current OMC application that the previous senior design team has built, there is no designated budget for the development of the application. OMC has purchased their domain and to deploy the application to Azure, the organization holds $3,500 in yearly Azure credits.

SendGrid, the integrated email delivery tool, has a free plan that is restricted to 100 emails per day. Therefore, we recommend the Bronze tier subscription, which has a charge of $9.95/month and includes up to 40,000 emails per month. If OMC exceeds the email quota on any given billing period, the organization will be charged a small overage fee on a per-email basis. However, OMC could qualify as a documented non-profit organization (501c3) to receive a 25% discount on email solutions after signing up.

There are no set up or monthly fees associated with PayPal and Square API integration. After providing support for both subscription services, PayPal and Square will each charge a standard transaction fee of 2.9% + $0.30 for every processed payment. This is not an external cost to OMC but is an important note to include in the documentation and may influence OMC’s pricing decisions for their events and courses.

11. Project Milestones

Est. Date	Milestones	Status
2/16	TA Check-In 1	Completed
2/18	Project Requirements / Initial Design Document	Completed
3/1	Setup Application v1 Locally	Completed
3/1	Test Functionality of Current Application	Completed
3/12	Status Presentation	Completed
3/15	Project Status and Documentation Review	Completed
3/23	Review Pending Requirements with Client to go Live	Completed
3/23	Setup Organizational GitHub	Completed
4/2	TA Check-In 2	Completed
4/6	Frontend Mockup with Client	Completed
4/6	Initial Curriculum Plan for High Schoolers	Completed
4/20	Frontend Prototype	Completed
4/20	Push Application v1 Live	Postponed
4/28	Final Design Document	Completed
5/1	Final Curriculum Plan for High Schoolers	
5/1	Senior Design 2 Kick-Off Meeting	
6/21	Critical Design Review	
7/1	Project Demo for Professors	
7/26	Application Workshop with High Schoolers	
8/6	Final Project Presentation	
8/6	Final Project Document	

12. Project Summary and Conclusions

All in all, with the strides that the Orlando Math Circle corporation is taking to deliver a virtualized, modern solution to instructional learning, the current Senior Design group is committed to bring to life the ideas that OMC has in the newly developed web application. Continuing on with the efforts that the previous group delivered, the current group is tasked with implementing improvements to the application such as providing a Volunteer Dashboard that places the most usable functions and information a volunteer would need to access into a tab, refactoring the event system to allow for support of swapped shifts, open events, and attendance tracking, developing fixes for bugs as well as an automated process that streamlines testing for future developers, and crafting the blueprint for high schoolers to expand Orlando Math Circle's teaching footprint into the realm of Computer Science. While the needs of the Orlando Math Circle seem large in nature, the effects and ideals that the staff, sponsors, and Orlando Math Circle hold as a whole are more profound in nature. With the goal to expand the application into a national scale that could extend past counties and state boundaries, the Orlando Math Circle is in need of a team that can realize the goal.

The document outlined the research efforts, ideas, current observations, and future implementations in a manner that is consistent with the meetings of the sponsors, the aptitude of the Senior Design team, and the developmental skill that will be utilized. As the document reaches an end, the research, meetings, and ideas will continue to develop. The final product will be one reflective of the shared goals of both the sponsors and the Senior Design team, and we look forward to helping transform the Orlando Math Circle into a hub that can impact student lives both in-person and virtually.

12.1 Gained Insights

After a semester has come and gone, we have learned as a team how to work together and perform when necessary. In our previous classes we were able to procrastinate and put assignments off until the last minute because we were only worried about our grades and personal performance, but in a team, we had to maintain a steady stream of communication and hold ourselves accountable for the activities that we said we would do. We had never been tasked with such a large project where we would have clients, other than professors, request a certain

list of features and expect us to create a solution and deliver a product that not only were they happy with using, but the users we happy with as well.

Towards the beginning, we were in a limbo period where we wanted to work on the project but had no idea where to start because all we could see was the end goal with no starting point. Once we started having regular meetings with our clients, the pieces of the puzzle started falling into place and we could start to have productive meetings on the intricacies of each requirement and how they worked together. We quickly began to realize that the warnings from our professors to spend at least 10 hours a week on the project had a true meaning behind them.

Our group would have benefited from extra planning before meetings with the sponsors to help reduce the duration of the meeting and receive concise feedback on solutions that we proposed. There were many times where we would schedule a meeting for thirty minutes that ended two hours later because of some minor details that may be altered in the future. Additionally, the group could have storyboarded the recording of notes more efficiently; while Zoom recordings of the meetings were put in place to potentially limit the amount of documenting that would be needed for each feature, the time consumed to re-watch the meetings and skip to the times necessary did limit the total efficiency of the group in deciding upon features and functionalities needed.

While there definitely were lessons learned in the process of drafting the document and performing reconnaissance, there were many things that we as a Senior Design group got right even during a virtualized period. For starters, punctuality from all of the group members were observed when it came to sponsor meetings, TA Check-Ins, and professor meetings. All group members were present and had substantial input to provide for meaningful discussion. Responsibility and weight of tasks that needed to get done were shared equally amongst the group with support from other group members that were not prioritized in the tasks. For example, early on within the group we separated tasking into two categories: frontend and backend. Naturally, the Orlando Math Circle application requirements were chunked in this manner with shift swapping falling more specific to a backend implementation and a Volunteer Dashboard lending itself more appropriately to the frontend. However, there were time periods where members from each side needed extra assistance and collaboration, an example being coordinating the High School Curriculum plan with the code learned on both ends. Because the OMC sponsors stated that the high schoolers will be learning about code and, if knowledgeable enough, key players in the maintenance of the application, all

group members needed to come together to ensure the best possible curriculum for the high schoolers.

Below is a table summarizing the individual critiques of the Senior Design 1 journey with a rating on the job done and potential improvements.

Critique	Rating (1 – 5)	Potential Suggestions
Punctuality – reporting on time for scheduled meetings with sponsors, outlining plans to talk, questions asked.	4.5	Like explained above, conduct personal meeting amongst the group to hammer out topics to talk about before sponsors. Could also report earlier to the Zoom meetings to conduct earlier if members are present.
Deadlines – meeting all deadlines of the Project Milestones, updating the required parties of any changes	4	All but one of the Project Milestone deadlines were met and it was pushing the Application live. This was due in part to a change that was realized amongst the sponsors and team that required additional time to refactor.
Communication – establishing healthy means of communication throughout both the group members and the sponsors	4	The group, through the use of communication mediums such as Discord and Zoom, created different channels that separated the team from the sponsors as well as additional channels dedicated to documenting bugs and software hiccups observed upon testing.
Comprehension – understanding the task and what is asked of the sponsors as well as the coding abilities of the team	3	Because of the dynamic needs of the Orlando Math Circle, requirements that were finalized ended up having to be revisited to account for the new functionalities needed. We are sure that even past this point we will need to conduct additional conversations with the sponsors to truly solidify the responsibilities needed.
Morale – the overall spirit of the group, pulling through in difficult times and showcasing	4.5	Even throughout the virtualized environment, the team was awesome in communicating with one another through Discord, keeping each updated with

support no matter the cause		happenings of the day-to-day that could have affected productivity. While a result of COVID-19, more in-person meetings could have been conducted as needed.
Initiative – taking charge of a situation should group member(s) need assistance with a particular task	4.5	The group stepped up and assisted others in need when the situation called for such. Each group member helped to support one another in the topics requested and ensured that morale was kept up because of that.
Accountability – holding oneself accountable tor group accelerations and/or setbacks that could have been observed throughout the SD1 experience.	5	The group understood that, especially in a virtualized environment, holding oneself accountable for the changes and work done was crucial in meeting the deadlines and furnishing the SD1 document as needed. We should expect to keep this same high level of accountability throughout the SD2 journey as well.

13. References

[1] Node.js v14.0.0 Documentation. Dockerizing a Node.js web app. Retrieved from https://nodejs.org/en/docs/guides/nodejs-docker-webapp/.

[2] Square. Square's Hosted Checkout — Set Up an Online Payment Form Today. Retrieved from https://squareup.com/us/en/townsquare/hosted-checkout-set-up-an-online-payment-form.

[3] Documentation: NestJS - a PROGRESSIVE Node.js framework. (n.d.). Retrieved April 28, 2021, from https://docs.nestjs.com/.

[4] Introduction: Datagrip. (n.d.). Retrieved April 28, 2021, from https://www.jetbrains.com/help/datagrip/meet-the-product.html.

[5] TypeScript Documentation. TypeScript for JavaScript Programmers Retrieved from https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html.

[6] Interaction Design Foundation. Usability. Retrieved from https://www.interaction-design.org/literature/topics/usability#:~:text=Usability%20refers%20to%20how%20easily,on%20three%20aspects%20in%20particular%3A&text=They%20should%20be%20able%20to,easily%20through%20using%20that%20design.

[7] Web Accessibility Initiative. Accessibility Principles. Retrieved from https://www.w3.org/WAI/fundamentals/accessibility-principles/.

[8] Introduction to Understanding WCAG 2.1. Understanding the Four Principles of Accessibility. Retrieved from https://www.w3.org/WAI/WCAG21/Understanding/intro#understanding-the-four-principles-of-accessibility.

[9] Web Accessibility Initiative. Essential Components of Web Accessibility. Retrieved from https://www.w3.org/WAI/fundamentals/components/.

[10] Vue.js. Directives. Retrieved from https://v3.vuejs.org/api/directives.html.

[11] Vue.js. The Vue Instance. Retrieved from https://vuejs.org/v2/guide/instance.html.

[12] Vuetify. Why you should be using Vuetify. Retrieved from https://vuetifyjs.com/en/introduction/why-vuetify/#what-is-vuetify3f.

[13] BootstrapVue. Retrieved from https://bootstrap-vue.org/.

[14] Buefy. Documentation. Retrieved from https://buefy.org/documentation/.

[15] Quasar Framework. Why Quasar? Retrieved from https://quasar.dev/introduction-to-quasar.

[16] CodeSandbox Documentation. Retrieved from https://codesandbox.io/docs#:~:text=CodeSandbox%20is%20an%20online%20editor,device%20with%20a%20web%20browser.

[17] Markdown Guide. Markdown Cheat Sheet. Retrieved from https://www.markdownguide.org/cheat-sheet/.

[18] Mailgun. (n.d.). Mailgun Pricing. Retrieved April 28, 2021, from https://www.mailgun.com/pricing/

[19] Mailjey. (n.d.). Mailjey Pricing. Retrieved April 28, 2021, from https://www.mailjet.com/pricing/

[20] Sendinblue. (n.d.). Sendinblue Pricing Retrieved April 28, 2021, from https://www.sendinblue.com/pricing/

[21] Amazon Web Services. (n.d.). Amazon SES pricing. Retrieved April 28, 2021, from https://aws.amazon.com/ses/pricing/

[22] Documentation: SendGrid. (n.d.). Retrieved April 26, 2021, from https://sendgrid.com/docs/

[23] Getting started with the SendGrid API: SendGrid. (n.d.). Retrieved April 26, 2021, from https://sendgrid.com/pricing/

[24] Pricing: SendGrid. (n.d.). Retrieved April 26, 2021, from https://sendgrid.com/pricing/

[25] GitHub Docs: GitHub (n.d.). Retrieved April 5, 2021, from https://docs.github.com/en/organizations/collaborating-with-groups-in-organizations/about-organizations

[26] Docker desktop WSL 2 backend. (2021, April 27). Retrieved April 28, 2021, from https://docs.docker.com/docker-for-windows/wsl/

[27] Gaon, R. (2020, April 30). Node.js explained. Retrieved April 28, 2021, from https://medium.com/swlh/node-js-explained-dad7b23d027d

[28] Geer, Z. (2021, February 11). How to learn node.js? - a complete roadmap for beginners. Retrieved April 28, 2021, from https://medium.com/edureka/learn-node-js-b3a9c6fb632c

[29] Microsoft. (n.d.). Retrieved April 28, 2021, from https://support.microsoft.com/en-us/windows/which-version-of-windows-operating-system-am-i-running-628bec99-476a-2c13-5296-9dd081cdd808

[30] Node.js. (n.d.). NodeJS Documentation. Retrieved April 28, 2021, from https://nodejs.org/en/docs/

[31] MikroORM Documentation v.4.5. Modeling Entity Relationships. Retrieved from, https://mikro-orm.io/docs/relationships