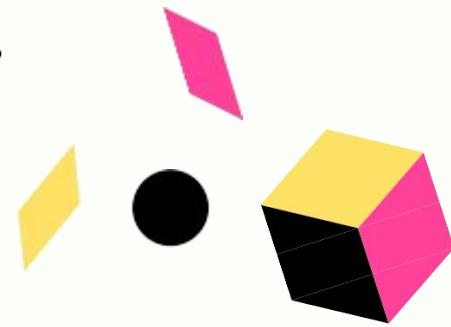


Group 10
John Gilbert
Mark Howell
Jason Mathew
Courtney Stewart
Joseph Tolbert



Sponsors
Margarita Azbel
Sheina Rodriguez
Matthew Villegas



Introductions

- John Gilbert — Project Manager, Lead Backend Developer
- Mark Howell — Lead Frontend Designer
- Jason Mathew — Frontend Designer
- Courtney Stewart — State Management & Backend Developer
- Joseph Tolbert — Backend Developer

Who is Orlando Math Circle (OMC)?

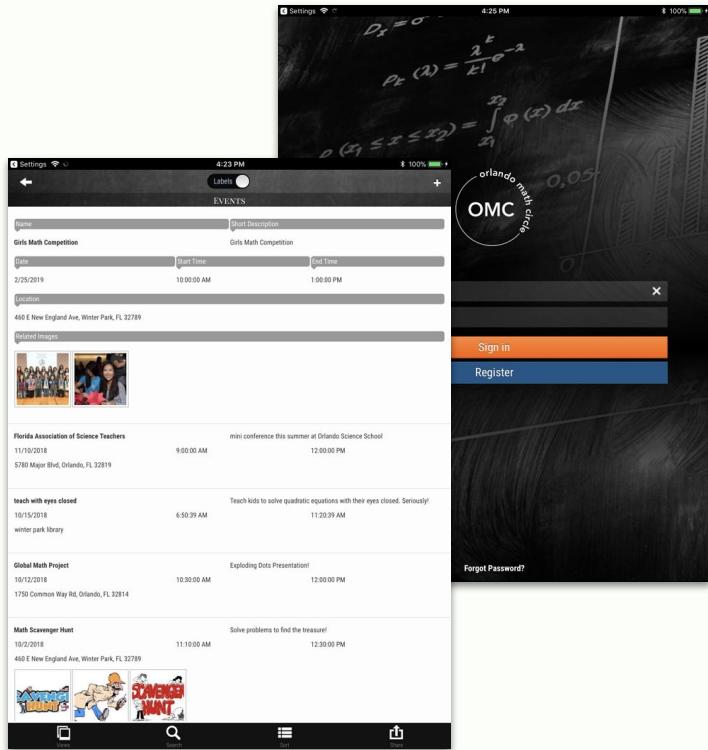
Orlando Math Circle aims to demystify mathematics for students of all ages through playful, joyous problem solving

- **Mission** — Create a diverse and inclusive community of student mathematicians
- **Strategy** — Provide engaging and fun opportunities to do mathematics outside of school
- **Vision** — Equitable access and greater student participation in mathematics
- 501(C)(3) non-profit organization



Existing Solution

- Broken, closed-source mobile app
- 7-step registration process that requires desktop
- Unintuitive interface
- Other calendar systems lack integration
- Existing accounts will be converted if possible



Project Goals & Requirements

We are developing an event calendar system with an app-like experience on a mobile browser.

- Event management and check-in system
- PayPal payments for events with fees
- Volunteer management and statistics
- Email notifications for events and newsletters
- Extended usability through admin panel
- Open source, documented, and extensible

Frontend

Frontend Overview

- 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
- Assets and design direction by Sheina

NUXTJS 

```
<template>
  <div>
    <v-toolbar flat color="transparent">
      <v-btn icon to="/">
        <v-icon large>mdi-chevron-left</v-icon>
      </v-btn>
    </v-toolbar>

    <v-container>
      ...
    </v-container>
  </div>
</template>

<script>
export default {
  layout: 'landing',
  auth: 'guest',
}
</script>

<style lang="scss" scoped>
.example {
  a {
    text-decoration: none;
  }
}
</style>
```



Landing Pages

A registration form titled "Registration" for the Orlando Math Circle. It includes fields for First Name and Last Name, Birthday Month, Day, and Year. There is a note asking for a parent or guardian to register. It also has fields for Email and Password, and a "Confirm Password" field. At the bottom, there are two checkboxes for receiving event emails: one for upcoming events and another for registered events.A sign-in form titled "Sign in". It has fields for "Email" and "Password". There is a "Remember me?" checkbox followed by a "Forgot password?" link. Below these are "LOG IN" and "SIGN UP" buttons. A "OR" button is positioned between the log in and sign up buttons.

Events Page

- Configurable calendar of events
- Sub-event page for registering to events
- Adults can register multiple users at once
- Fee payment through PayPal Orders API
- Users can register as volunteers and pick a job



Events on November 24th

The screenshot shows a sub-page for the "JavaScript Course" on November 24th. At the top is the course title "JavaScript Course". Below it are event details: "Tue, Nov 24th, 2020", "Jane Doe", and "Online". To the right is a small icon of a person sitting at a laptop displaying code. At the bottom is a navigation bar with four items: "Home", "Events", "Projects", and "Account".

The screenshot shows the "Event Details" section for the "JavaScript Course". It includes the date "Tue, Nov 11th, 2020", the registrant "Jane Doe", and the location "Online". To the right is a small icon of a person sitting at a laptop. Below this is a descriptive text: "Introductory JavaScript course for 6th grade students." Further down is a "Registration" section with a note: "Select which users you wish to register. You may add more users on the account page." A user profile for "Jane Doe" is listed with the status "Grade 6 — Selected" and a checked checkbox. At the bottom is a "Payment Due: \$15" section with a "PayPal" button.

The screenshot shows the "Payment Due: \$15" section. It features a large yellow "PayPal" button. At the bottom is a navigation bar with four items: "Home", "Events", "Projects", and "Account".

Projects Page

- Versatile page with function based on the user's role
- Admin statistics for tracking projects, volunteers, and volunteer hours
- Discovery tool for users to learn about the types of events hosted by OMC
- Volunteer section for manually reporting job hours not associated with an event
- Layout discussions in progress with sponsors

The screenshot shows the 'Projects' section of the Orlando Math Circle website. At the top, there is a header with the text 'Orlando Math Circle' and a three-dot menu icon. Below the header, a sub-header reads 'Projects' with a subtitle 'Events hosted by OMC'. A dropdown menu labeled 'Courses' is open. The main content area features two cards. The first card is for a 'JavaScript Course', which is described as an 'Occasional Course' available via 'Google Classroom' and 'Online'. It includes a small icon of a person standing next to a large laptop displaying code. The second card is for 'Girls Special Projects', described as a group to encourage girls in mathematics outside of school. It mentions activities like creating math videos, writing stories, developing lesson plans, and creating math art. It also notes that girls will have opportunities for math puzzles and games. This card includes an icon of a person standing next to a colorful, tangled ball of string. At the bottom of the page, there is a navigation bar with icons for Home, Events, Projects, and Account.

Orlando Math Circle

Projects

Events hosted by OMC

Courses

JavaScript Course

Learn the fundamentals of JavaScript, the language that powers the web!

Occasional Course

Google Classroom

Online

Girls Special Projects

A group to encourage girls in mathematics outside of school. Our mentors will help girls develop special projects like creating a math video, writing a math story, developing a lesson plan, or creating math art. Girls will have an opportunity to do math puzzles together and play math games.

Occasional Meetings

Google Classroom

P Sherman, 42 Wallaby Way, NY

Home Events Projects Account

Account Page & Switcher

- Customizable email notifications
- Volunteer and reduced lunch forms
- Adults can add other users to an account
- Switch easily between any user from the navigation bar
- (Stretch) Leveling or achievements system for child users

The screenshot displays the Orlando Math Circle application interface, illustrating the Account Page and Switcher features.

Account Page: The main screen shows "Orlando Math Circle" at the top. Below it is a circular profile picture with pink flowers. The "User Settings" section includes fields for "Email" (john@doe.com), "Birthday" (Monday, September 19th, 2016), "Roles" (Volunteer), and "Grade Level" (6). Buttons for "CHANGE EMAIL" and "CHANGE PASSWORD" are visible. A "Notification Settings" section allows changing the frequency and type of notification emails, with options for "Upcoming Events" (Newsletter-style emails) and "Reminders" (1 Hour Before, checked). A "Payment History" section lists recent transactions: "Event Fees - \$15.00" on Thursday, September 10th, 2020, for a JavaScript Course, and "Event Fees - \$50.00" on Saturday, August 1st, 2020, for Geometry Tutoring. A "Reduced Lunch Form" section provides instructions for students on the free lunch program.

Switcher: On the left, a sidebar titled "Select a User" shows three user profiles: "Jane Doe" (pink flower icon), "Jacky Doe" (geometric shapes icon), and "John Doe" (pink piggy bank icon). The "Account Users" section lists "Manage users" and "Jacky Doe Grade 6". The bottom navigation bar includes icons for Home, Events, Projects, and Account, with the Account icon being the active tab.

Home Page

- Reminder of time until registered events start
- News through Twitter feed integration
- Page awaiting further sponsor feedback for continued development

The screenshot shows the Orlando Math Circle website homepage. At the top right, there is a navigation bar with three horizontal lines, the text "Orlando Math Circle", and three vertical dots. Below this is a pink header section with the text "Latest News". Three news cards are displayed, each featuring a Twitter icon, the "Orlando Math Circle" logo (a circle with "OMC" and "orlando math circle" text), the date (September 14th, 2020), and a brief summary of the tweet content.

Latest News

Orlando Math Circle
September 14th, 2020

RT @BardMathCircle: A Thread: Today, we're excited to kick off the week sharing these free resources to help during the new normal; Remote...

Orlando Math Circle
September 14th, 2020

Register NOW!! Registration is now open for the Girls' Math Competition that will be held on October 17th. This...
<https://t.co/CQZIPFWTUP>

Orlando Math Circle
August 20th, 2020

Orlando Math Circle is changing the STEM landscape through calculus

Home **Events** **Projects** **Account**

Admin Panel

- Available only to users with the administrator role
- Create, update, or destroy any entity
- Search users by grade level, genders, ages, or roles
- Email users en-masse through the panel or by retrieving the emails comma-separated
- Dashboard with basic statistics
- Home of the captive attendance page
- Tabular data best suited for larger devices, but will operate just the same on mobile

The screenshot displays the OMC Admin interface. At the top, a header bar reads "OMC Admin". Below it is a table titled "Users" with columns: ID, First Name, Last Name, Email, Roles, and Fee Waived. The table contains three rows of data:

ID	First Name	Last Name	Email	Roles	Fee Waived
2	Jane	Doe	john@doe.com	admin	<input type="checkbox"/>
5	Jacky	Doe			<input type="checkbox"/>
4	John	Doe			<input type="checkbox"/>

To the left of the table is a vertical sidebar menu with icons for Home, Profile, Events, Courses, Projects, Volunteers, Uploads, Return to Site, and Logout. The "Users" option is highlighted. On the right side of the screen, there is a sidebar titled "OMC Admin" containing links for Dashboard, Users, Events, Courses, Projects, Volunteers, Uploads, Return to Site, and Logout. The "Users" link is also highlighted.

Add New Event

Add title
JavaScript Course

All-day

Start Date
Tuesday, September 22

Start Time
12:24

End Date
Tuesday, September 22

End Time
14:30

Does not repeat

Location
Online

Description
Learn about JavaScript in this introductory course!

Add New Event

Custom recurrence

Repeat every 1 week

Repeats On S M T W T F S

Ends

Never

On Thursday, October 1

After 10 occurrences

Add New Event

2:30 PM

CLOCK

AM PM CLOSE

OMC

5 AM XTreme Math 5 AM - 6:45 AM Tutoring, 5:30 AM - 6 AM

6 AM

7 AM XTreme Math, 6:45 AM - 7:30 AM

8 AM

9 AM Tutoring, 8:30 AM - 9:30 AM XTreme Math 9 AM - 10:15 AM

10 AM

11 AM Tutoring 10:15 AM - 11:45 AM

12 PM

1 PM

2 PM

3 PM Studying 2:45 PM - 4:30 PM

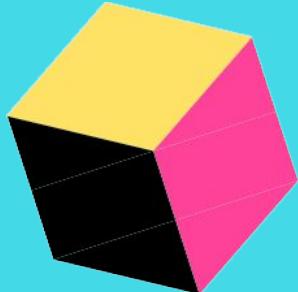
Home Events Projects Account

Vuex State Management

- Vue counterpart to React's Redux
- Event data is shared across homepage, projects, and events pages in a global state
- Components utilizing Vuex state react to changes automatically
- Retrieves auth and settings cookies on first load
- Attaches JWT as bearer token in each request
- Can reuse TypeScript definitions from backend for easier implementation

```
export const state = () => ({  
  token: {  
    jwt: null as string | null,  
    complete: false,  
    remember: true,  
  },  
  user: null as User | null,  
})  
  
export type AuthState = ReturnType<typeof state>  
  
export const getters: GetterTree<AuthState, AuthState> = {  
  loggedIn: (state) => state.token.jwt && state.token.complete,  
  isAdmin: (state) => state.token.jwt && state.token.complete &&  
    state.user?.roles?.includes(Roles.ADMIN),  
}  
  
export const mutations: MutationTree<AuthState> = {  
  SET_STATUS: (state, status: StateStatus) => (state.status = status),  
  SET_TOKEN: (state, data: AuthState['token']) =>  
    (state.token = Object.assign({}, state.token, data)),  
  SET_USER: (state, user: User) => (state.user = user),  
}  
  
export const actions: ActionTree<AuthState, AuthState> = {  
  async login({ commit, dispatch }, { email, password, remember }) {  
    commit('SET_STATUS', StateStatus.BUSY)  
  
    const { token, complete } = await this.$axios.$post('/login', {  
      email,  
      password,  
    })  
  
    dispatch('setTokenCookie', { jwt: token, complete, remember })  
  
    commit('SET_STATUS', StateStatus.WAITING)  
  },  
}
```

Backend

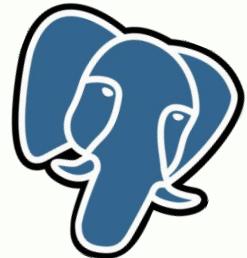


Backend Overview

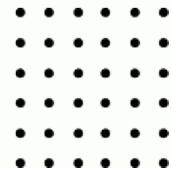
- PostgreSQL database with MikroORM
- Node.js and the Nest.js (Express Framework)
- Modular, class-based, and opinionated
- TypeScript and custom third-party interfaces
- Redis for the Bull queuing library
- Automatic crash recovery with PM2



Why PostgreSQL?

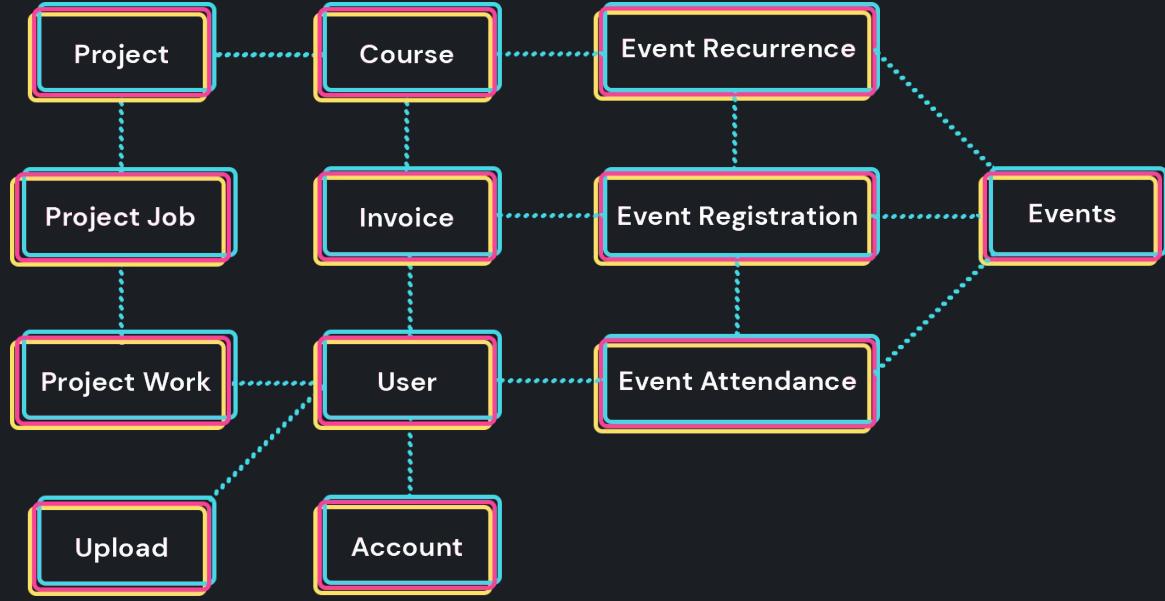


- Desire for joining data and reverse lookups eliminated NoSQL
- Constraints help enforce COPPA compliance and referential integrity
- PostgreSQL has a faster development schedule compared to MySQL
- JSON storage with query and index support through the *jsonb* type
- Automatic schema generation and TypeScript support with MikroORM



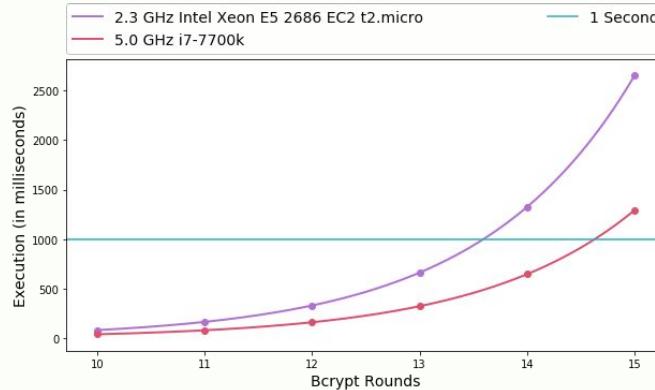
Entity Diagram

- Entities are generalized wherever possible for future development
- Constraints mapped to HTTP status errors for better developer experience (DX)



Accounts & Users

- Accounts have a primary user who owns the login credentials
- Semi-authenticated state after initial login which requires user selection for a full token
- Stateful JWTs stored as cookie
- Password hashed and salted with bcrypt
- Custom RBAC permissions decorators



```
@AccountAuth()  
@Get('me')  
getMe(@Acc() account: Account) {  
    return account;  
}  
  
@UserAuth('account', 'read:any')  
@Get(':id')  
findOne(@Param() { id }: FindOneAccountDto) {  
    return this.accountService.findOneOrFail(id);  
}
```

Events & Recurrence

- Events are metadata associated with a UTC **start** and **end** date
- Recurring events are described by the iCalendar RFC5545 **RRule** specification
- RRule created, serialized, and deserialized through the **rrule.js** library
- On-demand hydration of events
- Infinitely recurring events possible, but not desired

```
// Every week until January 30, 2021
const rrule = new RRule({
  freq: Frequency.WEEKLY,
  dtstart: new Date(Date.UTC(2020, 11, 1, 0, 0, 0)),
  until: new Date(Date.UTC(2021, 0, 30, 0, 0, 0)),
});

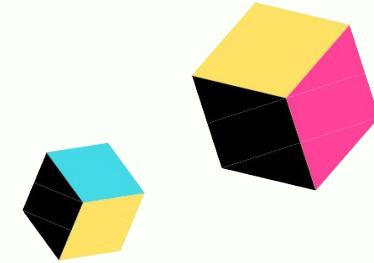
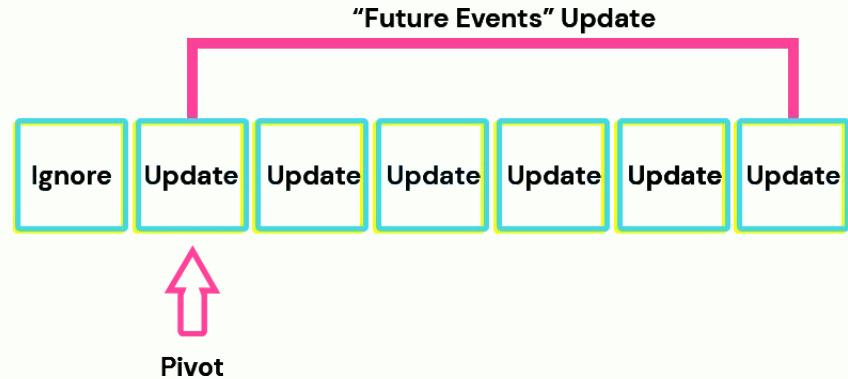
// Serialize (store in database)
rrule.toString();

// DTSTART:20201201T000000Z
// RRULE:FREQ=WEEKLY;UNTIL=20210130T000000Z;INTERVAL=1;WKST=MO

// Deserialize (get from database)
const recreatedRRule = rrulestr(...string);
```

Modifying Recurring Events

- Modifications can occur to individual events, future events, or to all events
- Deleting individual events stores an exception in the rrule
- Metadata changes are non-destructive
- Store original time if a single event is updated to avoid re-creating it later
- *All* or *Future* updates select a pivot, update it, then check if the other events can be updated or deleted



Event Registration

- Registration process
 - a. Receives an array of user ids and the event id
 - b. Compare user attributes to permissions structure on the event and validate
 - c. If the event has a course, retrieve it to determine payment modality
 - d. Ensure for each user there is a paid invoice
 - e. Generate registration records for each user
- Volunteering job requests are cascade inserted while creating the registration
- (Stretch) User tags to override specific permissions on a case-by-case basis

```
const capturedOrder = await this.paypalService.captureOrder(orderId);
const createInvoiceDtos: CreateInvoiceDto[] = [];

for (const purchase_unit of capturedOrder.purchase_units) {
  const capture = purchase_unit.paymentscaptures[0];

  createInvoiceDtos.push({
    status: InvoiceStatus.COMPLETED,
    id: capture.id,
    fee: capture.seller_receivable_breakdown.paypal_fee.value,
    gross: capture.seller_receivable_breakdown.gross_amount.value,
    net: capture.seller_receivable_breakdown.net_amount.value,
    purchasedAt: new Date(capture.create_time),
    event,
    user: +purchase_unit.reference_id,
  });
}

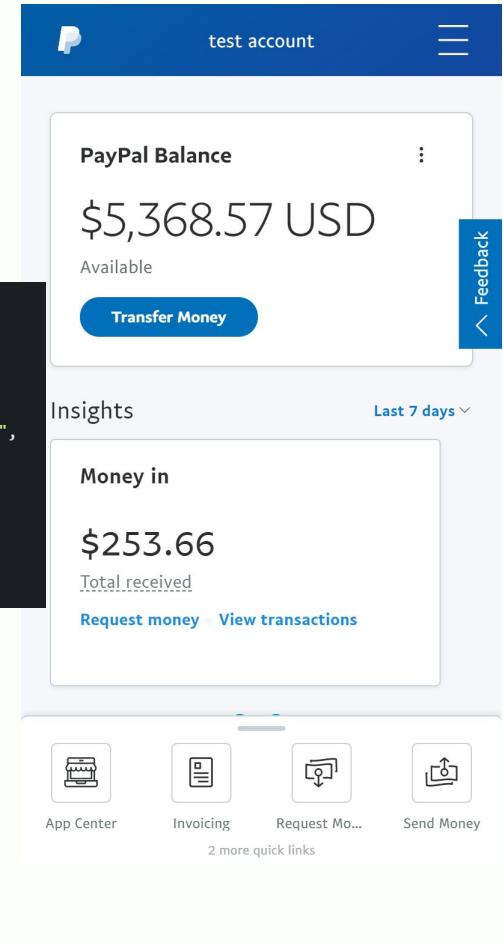
return this.invoiceService.batchCreate(createInvoiceDtos);
```



PayPal and Invoices

- PayPal Orders API utilizes OAuth 2.0
- Custom token service creates a bearer token, replaces expiring tokens, and retries requests on authentication errors
- Orders API only cares about cost (in \$)
- PayPal Smart Buttons flow:
 - a. Server **creates** an order, 1 unit per user.
 - b. User confirms order and popup closes
 - c. Server **captures** the order, then creates the invoice by mapping items to users
- Refunds credit the account, no direct refunds are made to the user

```
{  
    "purchase_units": [  
        {  
            "amount": {  
                "currency_code": "USD",  
                "value": "15.00"  
            }  
        }  
    ]  
}
```



Event Attendance

- Attendance kiosk at events will maintain an updated list of users and registrations through websockets
- Nest.js provides Socket.IO integration through a gateway interface
- Gateways accept subscribe messages used to join a room for the event
- Room is injected into account creation and registration methods to push changes
- Secured through WSS (SSL/TSL) and modified authentication flow

```
@WebSocketGateway(80, { namespace: 'events' })
export class EventAttendanceGateway {
  @SubscribeMessage('join')
  onJoin(
    @MessageBody() { eventId }: JoinEventRoomDto,
    @ConnectedSocket() client: Socket,
  ) {
    client.join(`event-${eventId}`)
  }
}

// Elsewhere...A user was just registered

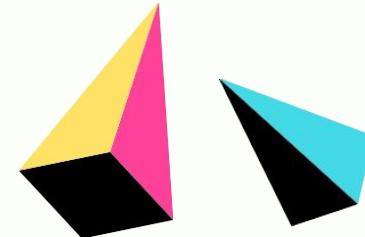
@WebSocketServer()
server: Server;

this.server.emit(`event-${registration.event.id}`, registration)
```

Courses & Projects

- Courses are a unique group of events that describe payment modality
- Projects are reusable metadata for events and courses to describe types of events
- Retrieved for the projects page to display metadata, or get payment requirements
- Names are on a “Working Title” basis

Project Programming Courses
Course Intro to JavaScript
Events Meeting Times



Volunteering

- Volunteers pick jobs during event registration
- Separate permissions on events for volunteers
- Jobs are associated with projects or events
- Project work is recorded during attendance
- Assumes work hours are completed, but can be deducted by admins in the panel
- Volunteer role is added through a volunteer handbook form in the account panel
- Generalized work is created through an approval process on a separate form

```
async create(eventId: number, user: User, jobId: number) {  
  const event = await this.eventService.findOneOrFail({  
    id: eventId,  
    jobs: {  
      id: jobId  
    }  
  }, ['jobs']);  
  
  // Check for existing registrations, invoices, and permissions  
  
  const work = this.projectWorkService.create(user, jobId);  
  
  user.work.push(work);  
  
  const registration = this.registrationRepository.create({ user, event })  
  
  await this.registrationRepository.persist(registration).flush();  
  
  return registration;  
}
```

Emailing & Notifications

- AWS Simple Email Service (SES)
- Rate-limiting through the **Bull** library
- Redis-backed for ensuring email delivery
- JSON store for counting bounces and complaints for automatic notification shutoff
- Nest.js module for cron-style scheduling
- Sponsors researching email service with more features for the organization

```
@Injectable()
export class EmailScheduler extends NestSchedule {
  constructor(@InjectQueue('email') private readonly queue: Queue) {
    super();
  }

  // Every 15th minute
  @Cron('*/15 * * *')
  enqueueNotifications() {
    this.queue.add('send-email-notifications', { attempts: 1 });
  }
}

@Processor('email')
export class EmailQueue {
  @Process({ name: 'send-email-notifications', concurrency: 1 })
  private async sendEmailNotifications(job: Job) {
    // Find event registrations and send email notifications
  }
}
```

File Uploads

- Files are uploaded using the *multipart/form-data* content type
- Nest.js provides a *Multer* library interceptor that extracts files to the disk
- File size and MIME type enforcement
- Files served statically for admin download
- File names hashed and not indexed for security
- Generalized feature that could be expanded with future development

```
@Post('upload')
@UseInterceptors(FilesInterceptor('form', 1))
uploadForm(@User() user, @UploadedFiles() files) {
  return this.uploadService.create(user, files)
}
```

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.example.com`

```
map $sent_http_content_type $expires {
    "text/html"                      epoch;
    "text/html; charset=utf-8"        epoch;
    default                           off;
}

server {
    listen      80;
    listen      443;
    server_name example.com;

    gzip        on;
    gzip_types  text/plain application/xml text/css application/javascript;
    gzip_min_length 1000;

    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://127.0.0.1:8000; # Frontend port
    }
}
```

Budget



vs.



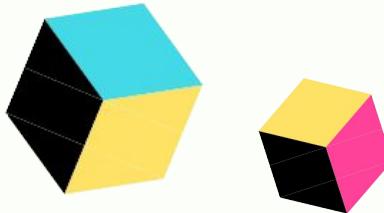
- Simple dashboard that does not require technical expertise to operate
- Tutorials and documentation that can be used for student education
- Best price for performance compared to AWS, Google Cloud Compute, Microsoft Azure, and other leading VPS providers.
- AWS features such as the Simple Email Service, would be cheaper
- More services yields more educational opportunities for students
- Cheaper reserved instances require 1-3 year commitments
- AWS increases learning curve for students

VM Instance	vCPU	RAM	Transfer	Storage	Cost
General Purpose Droplet	2 vCPUs	2 GB	3 TB	60 GB	\$15/mo
AWS EC2 t3.small	2 vCPUs	2 GB	\$0.09/GB	\$0.10/GB	\$17+/mo

Current Progress

Backend		Frontend	
100%	Database	100%	Registration / Login
90%	Authentication	100%	Accounts / User Selection
90%	Project / Event Dependencies	50%	Home Page
80%	Calendar API	25%	Admin Panel
50%	Emailing	50%	Projects / Event Page Layout
25%	Volunteers API	0%	Volunteer Hours Portal
30%	Final Testing	0%	Final Testing

Next Steps



- This upcoming week we will be giving a calendar feature demo to our sponsors and collecting feedback
- October will be spent rounding out minor features and implementing the volunteering systems
- Later tonight we meet with the high school students to gauge their skills and get them involved with the project and with our documentation website
- Switch from local to remote deployments for quality and accessibility testing on different kinds of mobile devices

Questions?

