Luke Barker

IoD Capstone Project - TurnUp

Application for Event and Venue Management

Introduction

People across the world are constantly looking for events to attend or venues to visit. Many of these people turn to social media to find events or venues. However, information about events or venues is often scattered, either across different event promotion or management accounts, or across different social media platforms entirely.
TurnUp aims to solve this by compiling information about events and venues in one place to be more accessible to the user.

With TurnUp, each **venue** can host **events**. By only associating events with venues, rather than promotional accounts or other third parties, events are much easier to discover.

Stakeholders in this solution include:

- People planning ahead for an event
- People spontaneously looking to attend an event
- Event planners
- Venue managers

By using TurnUp, the stakeholders can expect a more refined and clear-cut method of finding or hosting the events they want, or better yet, what they didn't know they wanted.

Product Description

The three primary entities of TurnUp are:

1. Users
2. Venues
3. Events

Users can own **venues**. **Events** are hosted by **venues. Venues** and their associated **events** can only be created, modified or deleted by the **user** that owns the **venue.**

These entities are reflected in the database logical model, as well as their foreign references to support features such as event/venue saving and event discussions and updates (see Appendix A).

User Stories

In order to design the flow of data within TurnUp, some user stories were considered. This ensures that features the stakeholders would need are implemented.

Luke Barker

Post completion note: rows that are marked in green have their user story satisfied with a feature that is implemented into the application and tested.

No#	User story	Available action to satisfy	Priority
1	"I'm out and about I know some venues near me, I want to know if there's any events happening right now at those venues."	They can search by name for each venue, and can see at a glance whether an event is happening now via an animated indicator.	High
2	"I know I'm available for a specific date range, I want to know what events are on in that range, regardless of venue."	The browse page displays all events, omitting those which have already ended. These events can be filtered between two dates	High
3	"I have found an event I'm interested in. I want to access it later without having to find it again."	This person can create a TurnUp account, which will allow them to save the event to their account for easy access later.	High
4	"I know a venue that I really like, I want to save it so I can easily see what events are on there."	This person can use their TurnUp account to save the venue to their account for easy access later.	High
5	"I am out and about and need to access the application from my mobile phone."	All site elements resize and flow such that all features and data is accessible on small screen sizes.	High
6	"I operate a venue and have created it on TurnUp. I don't want another user also creating my venue."	Venue locations are defined by Google's place ID. Venues' place IDs must be unique. This prevents two venues being created at the same location.	High
7	"I found an event and I want to discuss it with other users"	Any user that is signed in can post to an event's discussion, and view posts made by other users."	Medium
8	"I own a venue and am hosting an event. I want to keep users updated about the event."	Venue owners can update their events by posting to the events update feed.	Medium
9	"I created a venue and an event but some of the details are wrong, I need to edit them!"	Venue owners can edit the details of their venue and event. However, the location of a venue cannot be changed after creation.	Medium
10	"I want a live feed of users discussing an event I'm interested in."	**None -** a live feed is not a high priority for the user experience. The discussion posts are updated on each page load which is	Low

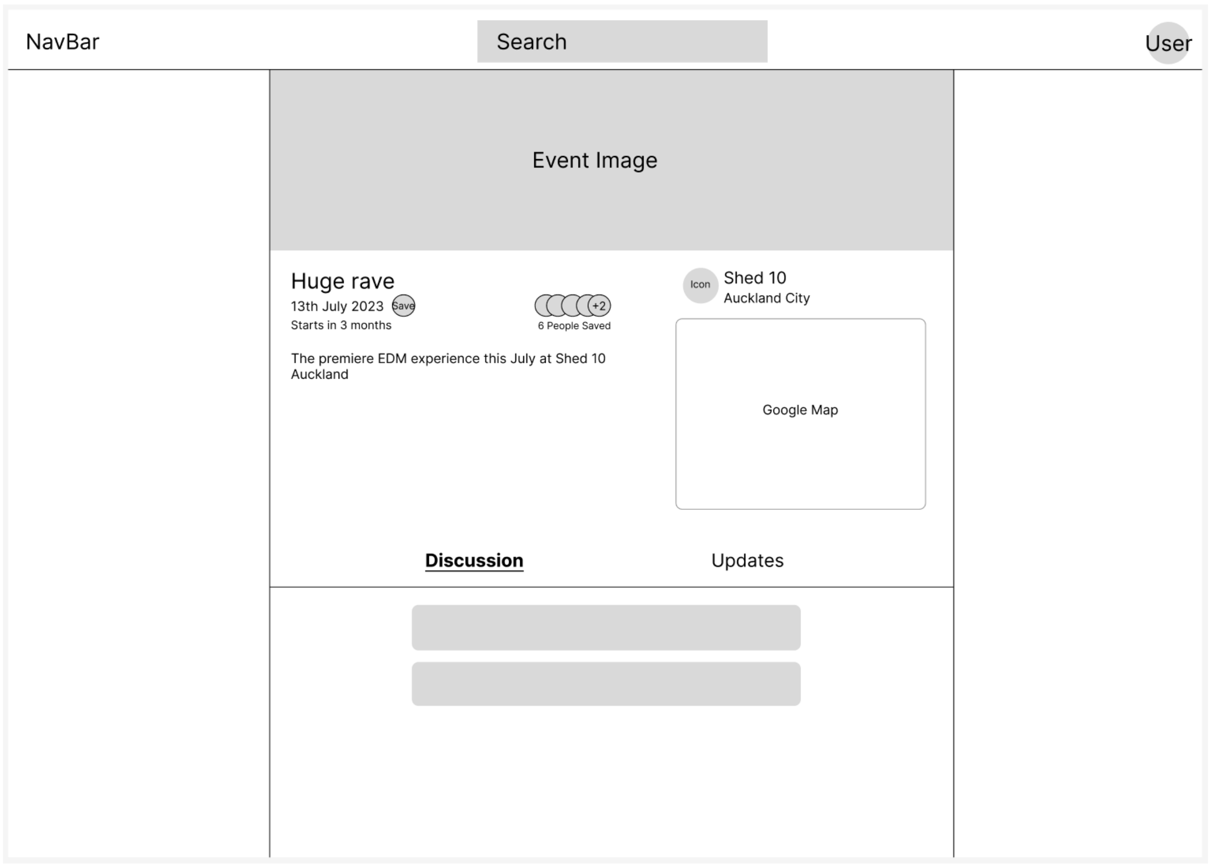
		sufficient	
11	“I want to find venues that are near me by letting the application know my location.”	**None -** other services such as Google provide these location-aware services.	Low
12	“I loved that event! I want to share my experience.”	**None -** Although users would appreciate this feature, the event and venue discovery aspect of the application is more important than the social aspect. Therefore, this feature is low priority.	Low

These user stories and their associated features can be viewed as each entity relating to each other (see Appendix B). These relations define the user flow of interaction.

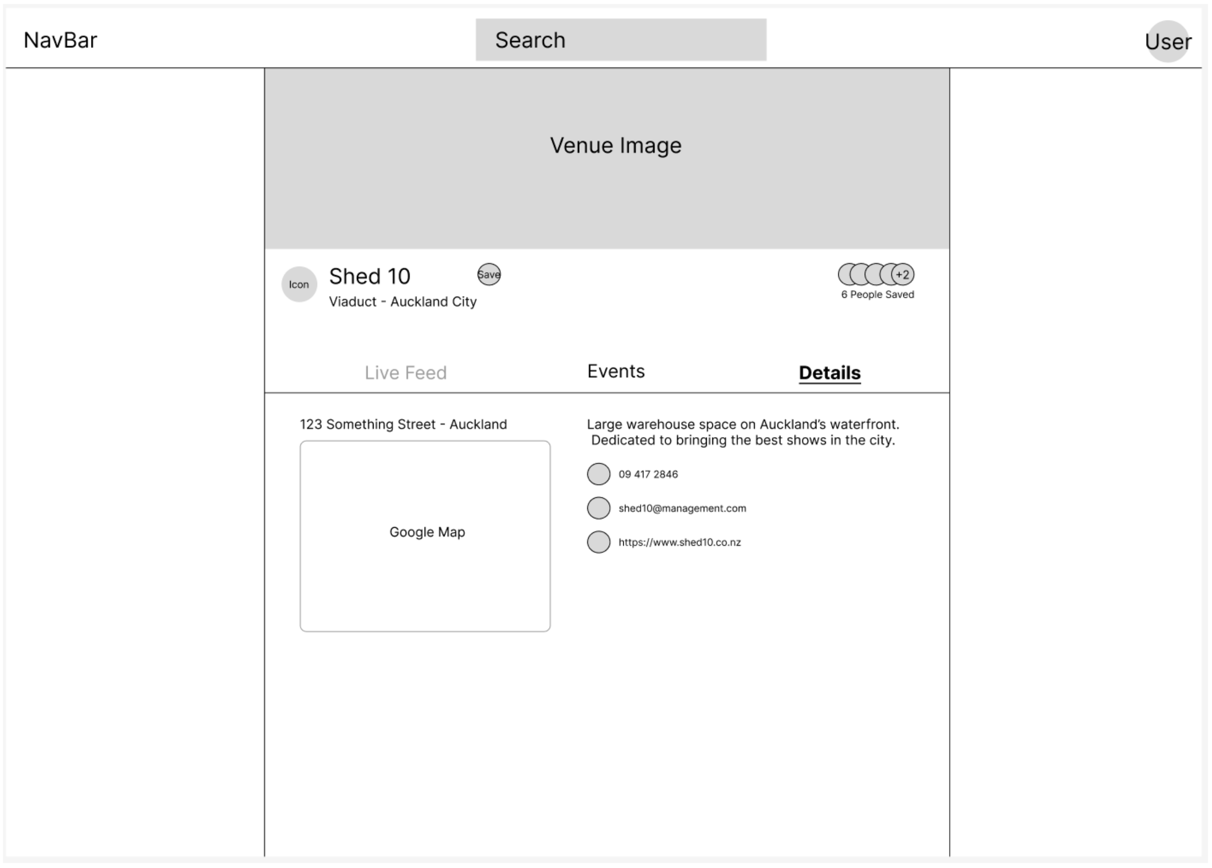
Design

Key pages such as the venue and event pages were designed with wireframe designs. Doing these wireframe designs first ensures that the user interface displays the data in an optimised layout.
Some minor changes exist between the initial wireframe designs and the final interface but the core layout has remained.

Single event page:



Single venue page:



Out of Scope

Some features are out of scope for this project. These features do not provide a contribution that is meaningful enough to the feature-set.

- **HTTPS.** An HTTPS connection to the backend is out of scope. It is common practice to send plaintext user information to the backend, such as sign in details, so long as it is over an encrypted HTTPS connection. Since establishing this connection is out of scope for the project, this poses a security risk but one that is not relevant enough within the scope.
- **Caching.** Caching queries from the database can speed up responsiveness of the application and improve efficiency of the server. This is not required for the core features of this application.
- **Venue owner verification.** Any user is permitted to create a venue so long as the location is not already used by another venue. This feature allows users to create venues for locations they own such as their own houses. However, issues could arise if a user creates a venue for a location they are not the actual owner of. Should this application be put into production, a method of verifying venue owners would need to be implemented. Such a method is outside the scope of this project.

By omitting these features, more time can be delegated to improving the more meaningful features of TurnUp.

Non-functional Requirements

The application has some requirements that do not necessarily improve the overall feature-set as far as the user is concerned. However, by implementing these requirements, users will be more comfortable while they use the core features.

User authentication

A smooth and robust user authentication process is crucial for the user experience, but also for security and the protection of users' information.

To provide this, TurnUp's backend uses JSON Web Tokens (JWT) and http only cookies to verify users (see Appendix C). This ensures tokens and their payloads are harder to compromise, by making them inaccessible to javascript.

All requests to the backend for restricted actions (Eg. get user's saved events, edit/delete entities) are protected by the validity of the JWT stored in the client's browser cookie (see Appendix D).

Ease of use

UI Cues

Users are most comfortable with an interface that is already familiar. To provide familiarity, some UI cues that are common among other applications must be implemented.

These cues include:

- **Card-based layout for sets of entities** such as displaying multiple events.
- **Venue and event text on cards clickable** and link to their corresponding page.
- **Hover reactive images** to suggest a gateway to more information.
- **Icon theme application-wide.** The same event and venue icons are used alongside each entity across the site, so users are able to identify the type of each entity at a glance.
- **Pop-up notifications** to confirm actions such as creating or editing an entity.
- **Tooltip on hover** to describe a button action.
- **User profile pictures** help users visually distinguish other users. This is particularly important in the context of the event discussions.
- **Logo links to home page.** The logo in the top left of the page is clickable and will direct users back to the home page.

These familiar UI cues will help users intuitively navigate the application.

Google Places API

In order to create venues, users must select their location. Google's places API streamlines this process by using their location lookup service to select locations.
By using Google's API, it also allows unique venue locations to be enforced by Google's unique place ID.

Reliability

A crash or unexpected error interrupts the user experience at best and ruins it at worst. To mitigate the risk of this, some features must be implemented to reduce the risk of application failures.

Some of these features include:

- **Verifying the user before attempting to display the application.** By doing this, pages already know whether or not the user is allowed to access them and can redirect the user accordingly. This is far less jarring than the user being denied access due to an authentication error.
- **Loading states and visualisation.** By implementing loading states, pages do not attempt to access data that does not yet exist. Attempting to display undefined data has the potential to cause errors. A loading indicator should also be displayed to the user while data is loading so the user knows to wait.
- **Re-enter password on sign up.** Requiring users to enter their password twice and disabling submission unless they match will reduce user error.

By incorporating these non-functional requirements, users can use the key features of the site and be more comfortable while doing so.

Project Planning

Each task that is required to complete the application features is listed with a date for completion. By tracking all these tasks, a timeline can be formed to ensure the project is complete by the deadline.
Trello is used to track these tasks (see Appendix E).

Each completed and pending task was separated into two categories: frontend and backend. This was to make sure backend tasks that the frontend relied on could be completed before attempting to integrate them.

Testing Strategy

In order to ensure all required features are functional, a testing strategy must be implemented.

The testing strategy for TurnUp is to identify each CRUD operation on each entity that must be functional in order to achieve the required application features. Then, identify the restrictions for each operation (eg. only the signed in user can update their account).

Once each operation and their associated restrictions are identified, they are tested to verify functionality twice:

1. Backend only - using HTTP request from Postman to verify the restrictions are enforced, and the correct data is in the response if they are complied with.

2. From TurnUp client frontend - The client must be able to interact with each operation. The UI must reflect the restrictions enforced by the backend (eg. redirect a user trying to modify an event they don't host).

Operations must pass the first backend test, in order to be valid for the second frontend test. *Note: All object property names are distinguished by being **bold** in the testing results. Medium priority features; Event/venue saving, event discussions and event updates are not documented here due to their lower priority. Regardless, they have undergone the same testing strategy and are implemented in the application.*

Backend

All features and their corresponding request to a backend operation were tested. Each operation enforced the restrictions successfully and returned the correct information if the restrictions were complied with. All backend tests passed (see Appendix F).

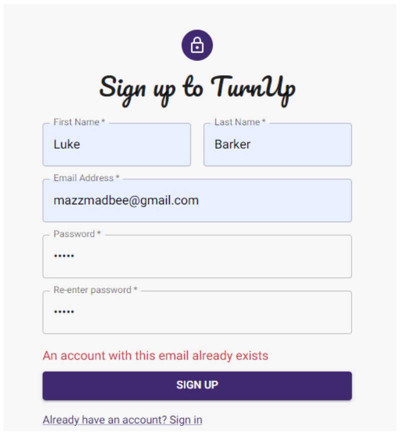
Frontend

Now that all backend operations have passed testing, the frontend must interact with them. The restrictions in place for each operation must also be reflected in the UI.

Each tested action and their corresponding UI integration is presented as follows:

Users

Action	**Restriction**
Create (Sign up) - POST	**firstName, lastName, email, password** - all required - **email** must be unique



Sign up to TurnUp

First Name *: Luke

Last Name *: Barker

Email Address *: mazzmadbee@gmail.com

Password *: •••••

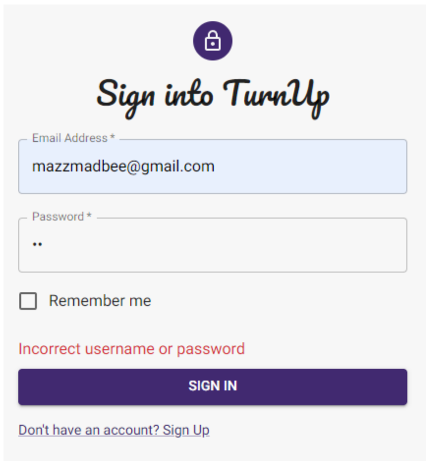
Re-enter password *: •••••

An account with this email already exists

All sign up fields are required in the form

User is notified if an email already exists and signup fails.

Action	Restriction
Read (Sign in) - POST	**email, password** - all required



All sign in fields are required in the form

User is notified of incorrect credentials

Sign in is successful if the restrictions are met

Action	Restriction
Update (Edit account) - PUT	Valid cookie must be provided

The account edit page of a user is completely inaccessible to any user that does not own it. Users that are not signed in are redirected to the sign in page.

Action	Restriction
Read (Verify account) - GET	Valid cookie must be provided

On page load, the application automatically attempts to verify the browser cookie, if it is invalid, the user is not signed in. If the cookie is valid, the user is signed in successfully.

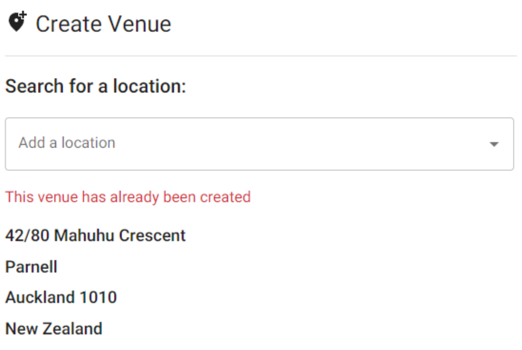
Venues

Action	Restriction
Create (Create venue) - POST	**address, placeId, ownerUserId, name** - all required - valid cookie provided - **placeId** unique

The create venue form cannot be submitted until restrictions are complied with. If the user is not signed in, they are redirected to the sign in page. If the required data is provided, the venue is created successfully.

Luke Barker

Users are notified if the location is not available, therefore **placeId** would not be unique.

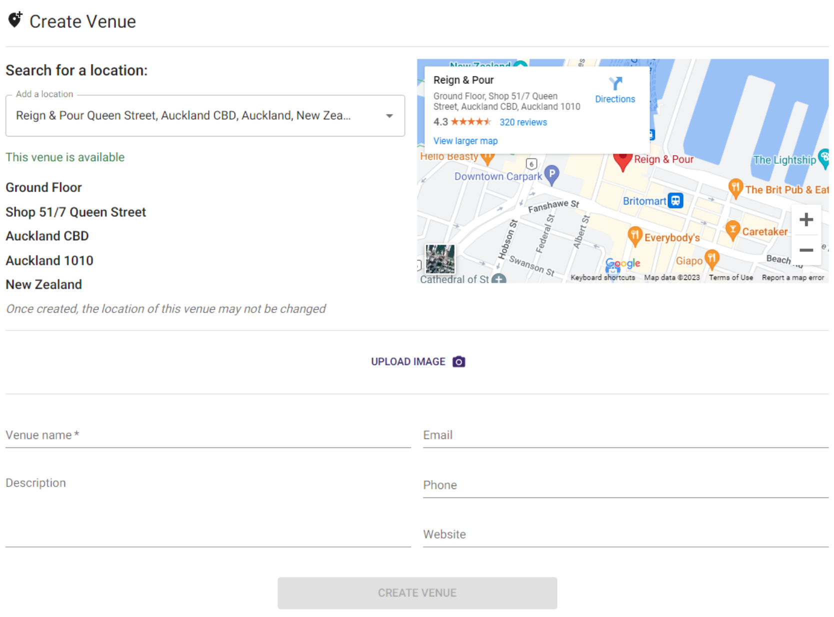

Create Venue

Search for a location:

Add a location

This venue has already been created

42/80 Mahuhu Crescent
Parnell
Auckland 1010
New Zealand


Create Venue

Search for a location:

Add a location: Reign & Pour Queen Street, Auckland CBD, Auckland, New Zea...

This venue is available

Ground Floor
Shop 51/7 Queen Street
Auckland CBD
Auckland 1010
New Zealand

Once created, the location of this venue may not be changed

Reign & Pour
Ground Floor, Shop 51/7 Queen Street, Auckland CBD, Auckland 1010
4.3 ★★★★★ 320 reviews
View larger map

UPLOAD IMAGE

Venue name *

Email

Description

Phone

Website

CREATE VENUE

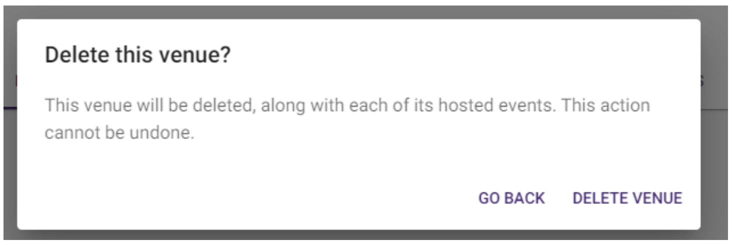
Users are warned that their venue location cannot be changed.

Here - the venue name is missing and the submission button is disabled.

Action	**Restriction**
Edit (Edit venue) - PUT	Valid cookie provided - **userId** must match **ownerUserId**. **placeId, address, addressArray** cannot be changed.

If the signed in user does not own the venue, they are redirected from the edit page. No UI is present that allows the user to send a request that attempts to change **placeId, address** or **addressArray.** If the required data is provided, the venue is edited successfully.

Action	**Restriction**
Delete (Delete venue) - DELETE	Valid cookie provided - **userId** must match **ownerUserId**.

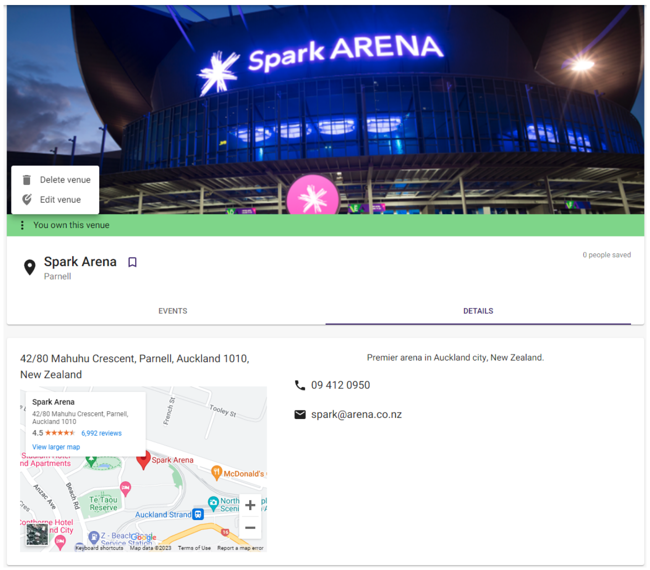

Delete this venue?

This venue will be deleted, along with each of its hosted events. This action cannot be undone.

GO BACK DELETE VENUE

Venue deletion is inaccessible to a user that does not own the venue. For the user that does own the venue, the venue can be deleted successfully, with a confirmation warning dialog to reduce user error.

Action	**Restriction**
Read (Get venue page data) - GET	**venueId** required



Delete venue
Edit venue
You own this venue
Spark Arena
Parnell
0 people saved
EVENTS
DETAILS

42/80 Mahuhu Crescent, Parnell, Auckland 1010, New Zealand

Premier arena in Auckland city, New Zealand.

09 412 0950

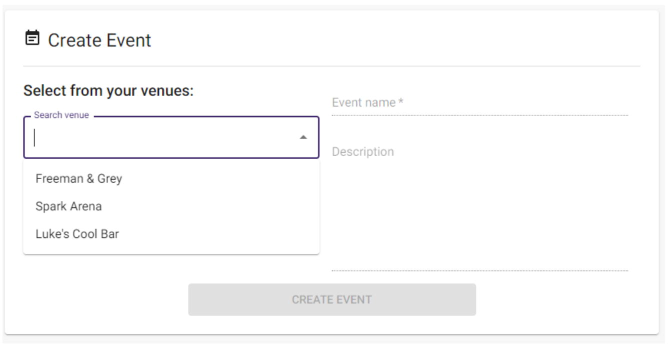
spark@arena.co.nz

The single venue page displays the data of the given **venueId** successfully.

Note - here you can see this user owns the venue and can access the edit and delete options.

Events

Action	**Restriction**
Create (Create event) - POST	Valid cookie provided - **userId** must match **ownerUserId** of venue. **venueId** must exist. **name, venueId, date, endDate** all required



Create Event

Select from your venues:

Search venue

Freeman & Grey
Spark Arena
Luke's Cool Bar

Event name *

Description

CREATE EVENT

Users only have the venues they own available to them.

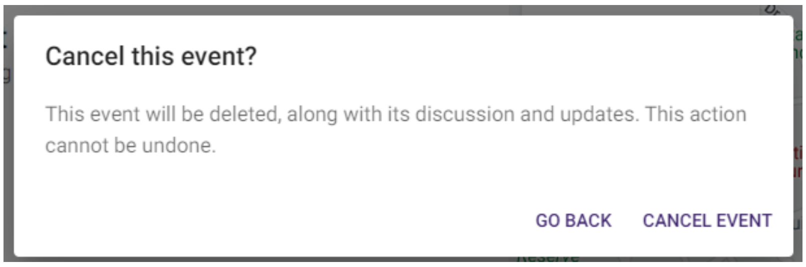
The create event form submission is disabled until the required data is provided.

Events are successfully created if the required data is provided.

Action	**Restriction**
Update (Edit event) - PUT	Valid cookie provided **- userId** must match **ownerUserId** of venue. New **venueId** must exist. **name, venueId, date, endDate** all required

The event edit page is inaccessible to a user that does not own the venue that hosts it. Such users are redirected to the sign in page. Only venues that are owned by the user are available to select from. The edit event form submission is disabled if the required data is not provided.

Action	**Restriction**
Delete (Delete venue) - DELETE	Valid cookie provided **- userId** must match **ownerUserId**.



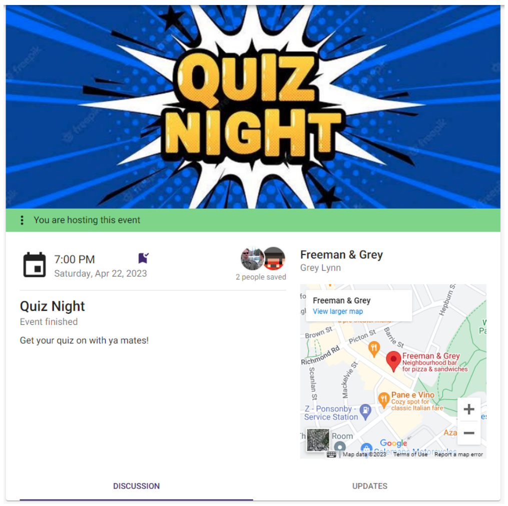
Cancel this event?

This event will be deleted, along with its discussion and updates. This action cannot be undone.

GO BACK CANCEL EVENT

Event deletion is inaccessible to a user that does not own the venue that hosts it. If a user owns the venue that hosts the event, they may delete it via a confirmation dialog to reduce user error.

Action	**Restriction**
Read (Get event page data) - GET	**eventId** provided



You are hosting this event

7:00 PM
Saturday, Apr 22, 2023

2 people saved

Freeman & Grey
Grey Lynn

Quiz Night
Event finished
Get your quiz on with ya mates!

The single event page displays the data of the given **eventId** successfully, along with its populated venue data.

Edge Cases

There are some known edge cases that the application will have to handle during usage:

A user attempting to access a restricted resource or action.
The most likely edge case to occur is a user attempting to access or modify an entity they know they are not permitted to. By ensuring each action is restricted on both the backend and the frontend, this edge case is unlikely to produce issues with the application by throwing errors.

Unexpected user flow.
A user could navigate the application in a way that is not intended.
This could include:

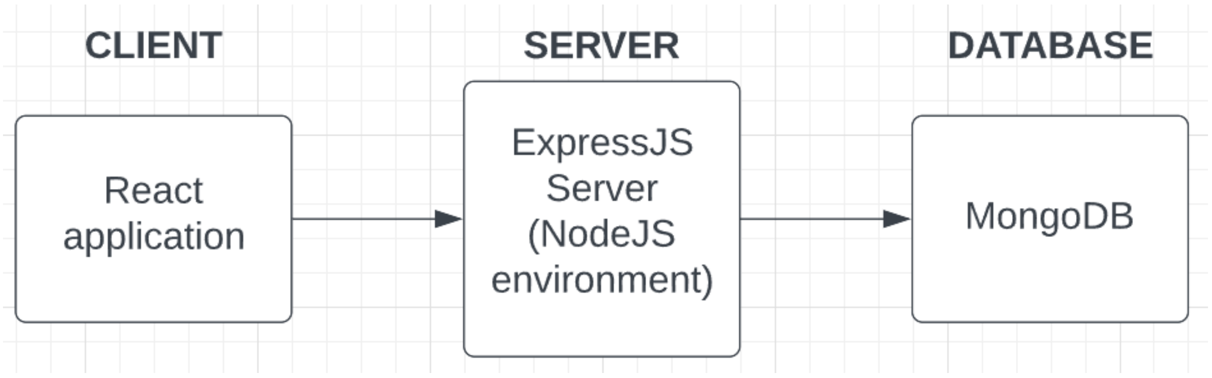
- Navigating to an entity that doesn't exist
- Accessing resources directly from the browser address bar.

Navigations such as these are handled by having failed requests to the backend caught and displaying an error message in the appropriate section of the page.
Accessing resources from the address bar will trigger a page reload and reinitialise the user and all data, which will mitigate any errors.

By undergoing each of these tests, it can be ensured that the users of TurnUp will have a smooth and fully functional user experience.

Implementation

The application solution uses a React application for the client, an ExpressJS server running in a NodeJS for the backend and MongoDB for the database. The application stack is visualised below:



End-to-end solution

The end application solution met all high and medium priority objectives effectively.
Omitting the low-priority and out-of-scope features had little impact on the final usability and features of the solution.
Functional and nonfunctional requirements have been successfully integrated to provide a smooth, intuitive and stable user experience.

References

Some external libraries, frameworks and APIs were used to construct the application. They are as follows:

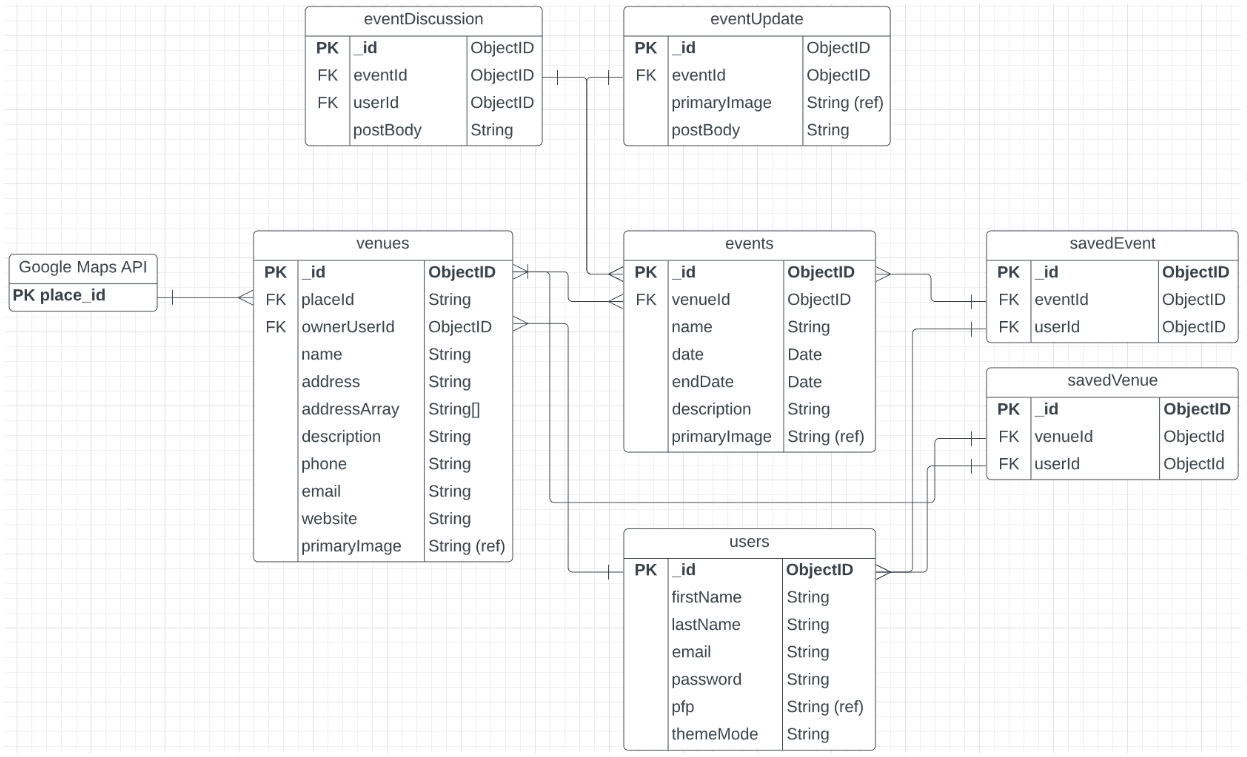
- React
- MUI
- React-toastify
- Axios
- Express JS
- Jsonwebtoken
- Cookie-parser
- Multer
- Bcrypt
- Mongoose
- MongoDB
- Google Places API

Code bases:

Server: https://github.com/MazzomesClone/IoD-Capstone-TurnUp-Server
Client: https://github.com/MazzomesClone/IoD-Capstone-TurnUp-Client

Appendices

A: Logical database model



eventDiscussion

PK	_id	ObjectID
FK	eventId	ObjectID
FK	userId	ObjectID
	postBody	String

eventUpdate

PK	_id	ObjectID
FK	eventId	ObjectID
	primaryImage	String (ref)
	postBody	String

Google Maps API

PK place_id

venues

PK	_id	ObjectID
FK	placeId	String
FK	ownerUserId	ObjectID
	name	String
	address	String
	addressArray	String[]
	description	String
	phone	String
	email	String
	website	String
	primaryImage	String (ref)

events

PK	_id	ObjectID
FK	venueId	ObjectID
	name	String
	date	Date
	endDate	Date
	description	String
	primaryImage	String (ref)

savedEvent

PK	_id	ObjectID
FK	eventId	ObjectID
FK	userId	ObjectID

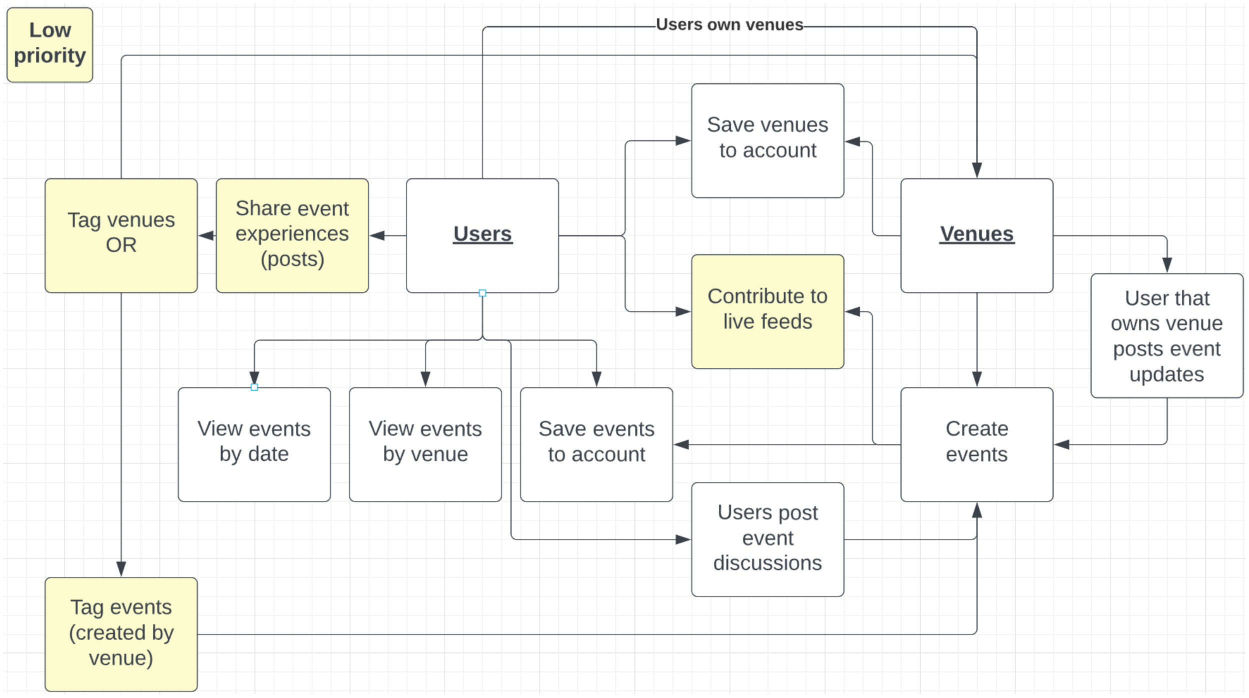
savedVenue

PK	_id	ObjectID
FK	venueId	ObjectId
FK	userId	ObjectId

users

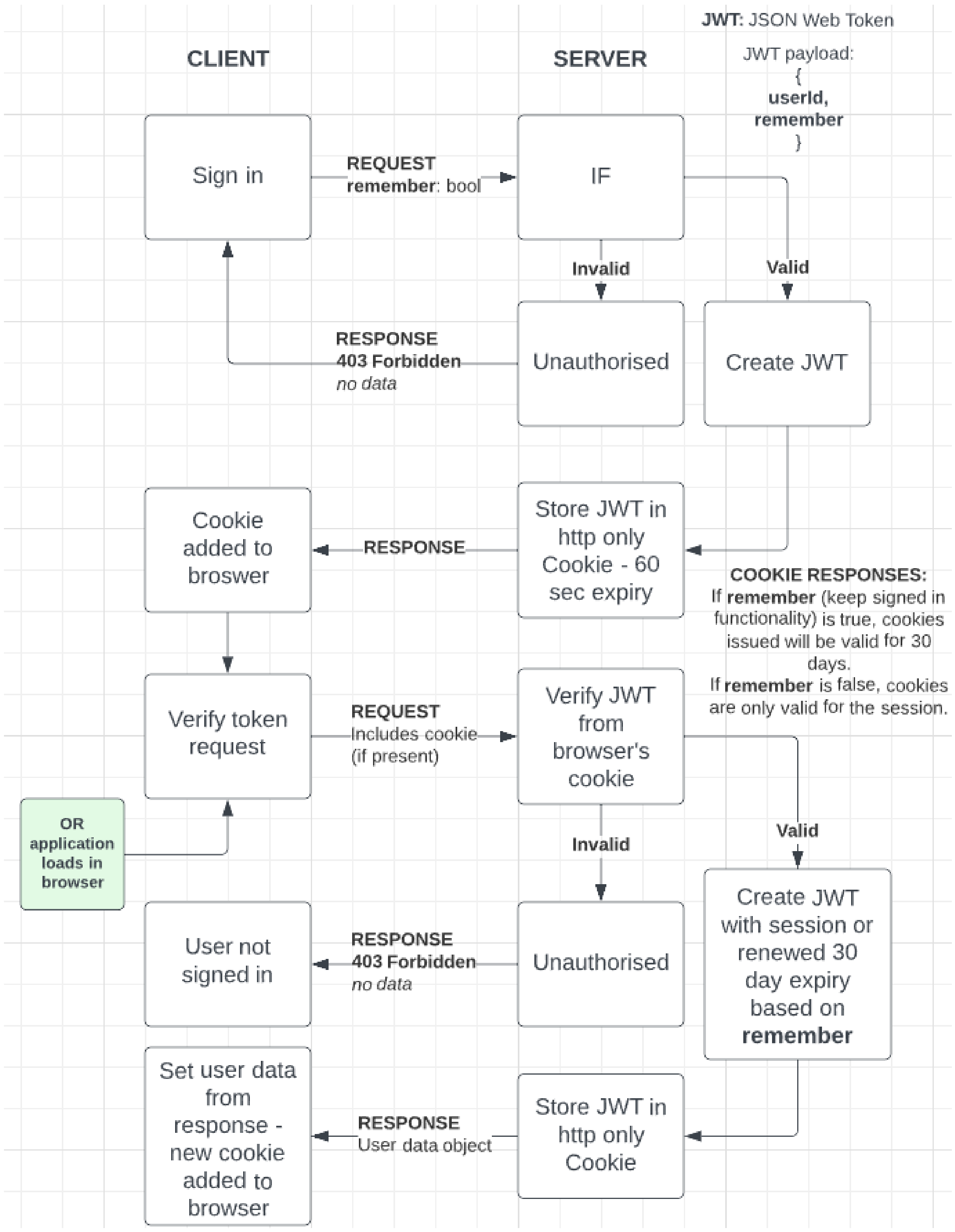
PK	_id	ObjectID
	firstName	String
	lastName	String
	email	String
	password	String
	pfp	String (ref)
	themeMode	String

B: Entity interactions



Luke Barker

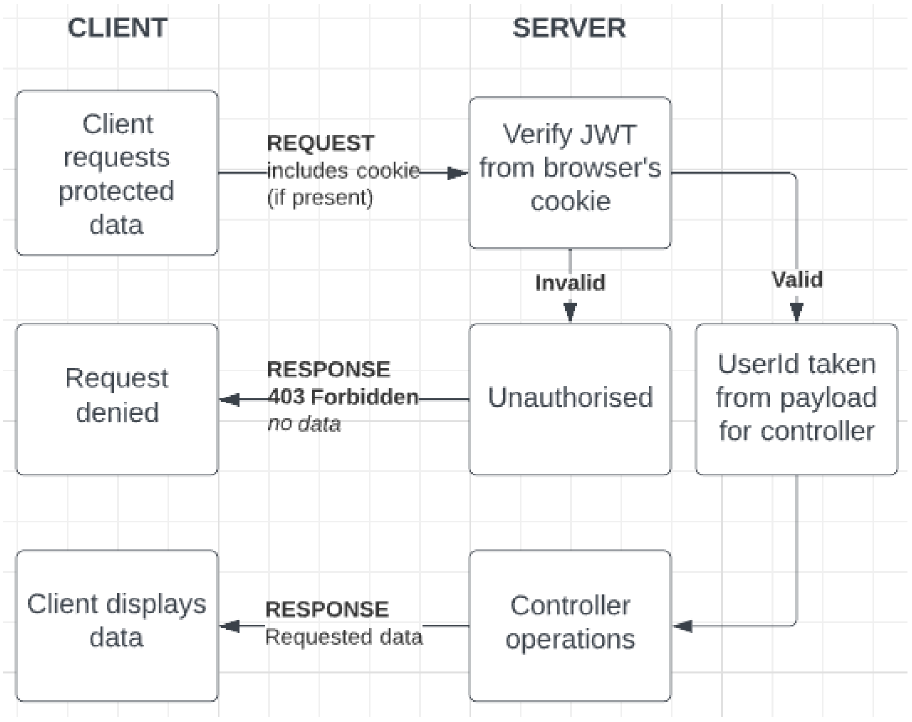
C: Authentication flow



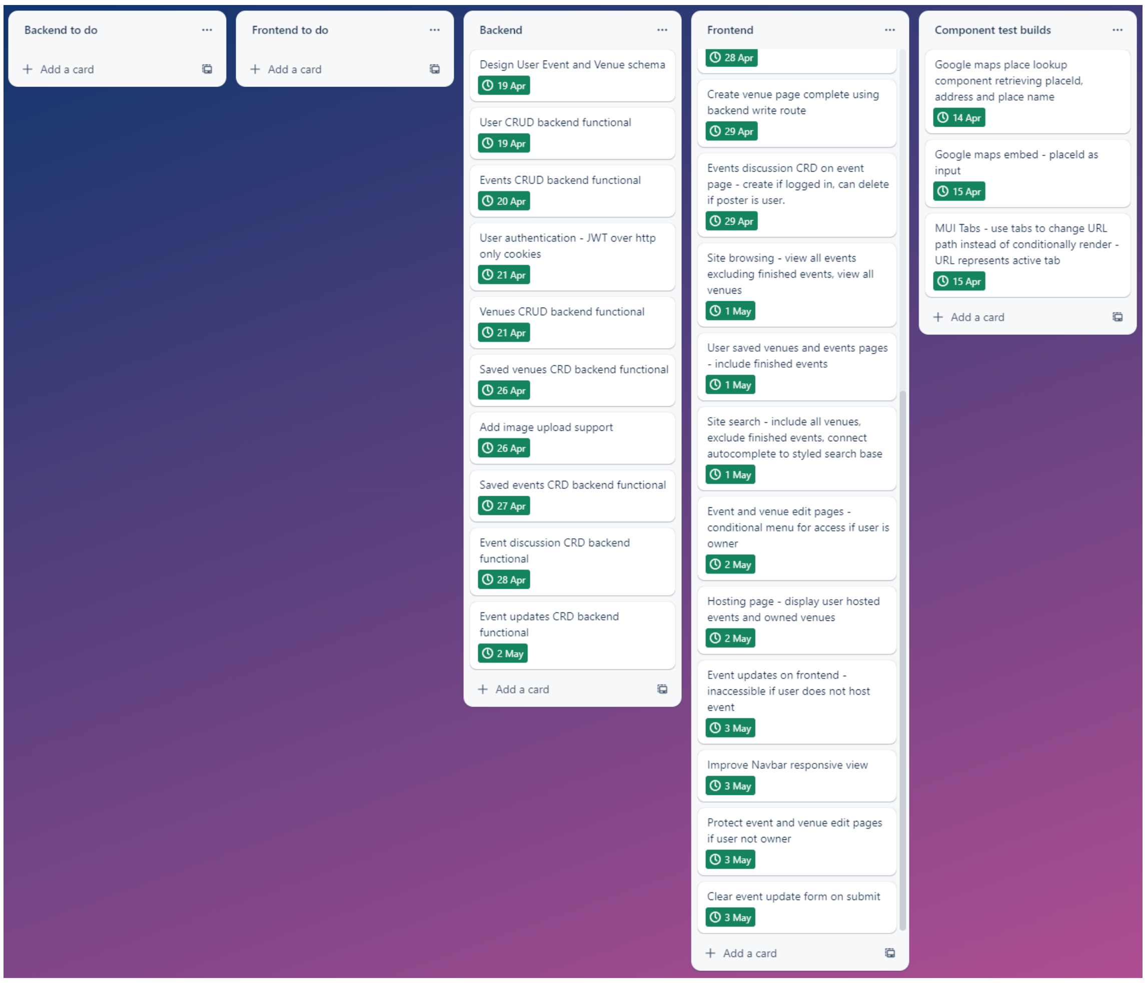
JWT: JSON Web Token
JWT payload: { userId, remember }
CLIENT
SERVER
Sign in
REQUEST remember: bool
IF
Invalid
Valid
Unauthorised
Create JWT
RESPONSE 403 Forbidden no data
Cookie added to broswer
RESPONSE
Store JWT in http only Cookie - 60 sec expiry
COOKIE RESPONSES:
If remember (keep signed in functionality) is true, cookies issued will be valid for 30 days.
If remember is false, cookies are only valid for the session.
Verify token request
REQUEST Includes cookie (if present)
Verify JWT from browser's cookie
OR application loads in browser
Invalid
Valid
User not signed in
RESPONSE 403 Forbidden no data
Unauthorised
Create JWT with session or renewed 30 day expiry based on remember
Set user data from response - new cookie added to browser
RESPONSE User data object
Store JWT in http only Cookie

Luke Barker

D. Client requests restricted data



E. Trello tasks


Backend to do

+ Add a card

Frontend to do

+ Add a card

Backend

- Design User Event and Venue schema — 19 Apr
- User CRUD backend functional — 19 Apr
- Events CRUD backend functional — 20 Apr
- User authentication - JWT over http only cookies — 21 Apr
- Venues CRUD backend functional — 21 Apr
- Saved venues CRD backend functional — 26 Apr
- Add image upload support — 26 Apr
- Saved events CRD backend functional — 27 Apr
- Event discussion CRD backend functional — 28 Apr
- Event updates CRD backend functional — 2 May

+ Add a card

Frontend

- 28 Apr
- Create venue page complete using backend write route — 29 Apr
- Events discussion CRD on event page - create if logged in, can delete if poster is user. — 29 Apr
- Site browsing - view all events excluding finished events, view all venues — 1 May
- User saved venues and events pages - include finished events — 1 May
- Site search - include all venues, exclude finished events, connect autocomplete to styled search base — 1 May
- Event and venue edit pages - conditional menu for access if user is owner — 2 May
- Hosting page - display user hosted events and owned venues — 2 May
- Event updates on frontend - inaccessible if user does not host event — 3 May
- Improve Navbar responsive view — 3 May
- Protect event and venue edit pages if user not owner — 3 May
- Clear event update form on submit — 3 May

+ Add a card

Component test builds

- Google maps place lookup component retrieving placeId, address and place name — 14 Apr
- Google maps embed - placeId as input — 15 Apr
- MUI Tabs - use tabs to change URL path instead of conditionally render - URL represents active tab — 15 Apr

+ Add a card

Luke Barker

F. Backend testing and results

Users

Action	**Restriction**
Create (Sign up) - POST	**firstName, lastName, email, password** - all required - **email** must be unique

Test input	**Required result**	**Pass**
Required credential missing	**400** Missing registration fields	✓
All fields present but email not unique	**409** Account already exists	✓
All fields present, email unique	**201** User registered - writes to database	✓

Action	**Restriction**
Read (Sign in) - GET	**email, password** - all required

Test input	**Required result**	**Pass**
Required credential missing	**401** Missing credentials	✓
Invalid email or password	**403** Invalid credentials	✓
Valid email, correct password	**200** Successfully logged in - cookie with token issued	✓

Action	**Restriction**
Update (Edit account) - PUT	Valid cookie must be provided

Test input	**Required result**	**Pass**
Invalid cookie	**403** Authentication failed	✓
Valid cookie, new user data and new profile picture	**200** User data updated in database - image saved, reference saved to user data	✓

Action	Restriction
Read (Verify account) - GET	Valid cookie must be provided

Test input	Required result	Pass
Invalid cookie	**403** Authentication failed	✓
Valid cookie	**200** User data in response	✓

Venues

Action	Restriction
Create (Create venue) - POST	**address, placeId, ownerUserId, name** - all required - valid cookie provided - **placeId** unique

Test input	Required result	Pass
Invalid cookie	**403** Authentication failed	✓
Required data missing - valid cookie	**500** Mongoose error thrown and caught - error message response	✓
Required data present, placeId already exists - valid cookie	**201** User registered - writes to database - image saved if present	✓

Action	Restriction
Edit (Edit venue) - PUT	Valid cookie provided - **userId** must match **ownerUserId**. **placeId, address, addressArray** cannot be changed.

Test input	Required result	Pass
Invalid cookie	**403** Authentication failed	✓
Valid cookie - user does not own venue	**403** User does not own venue	✓

Valid cookie - user owns venue - changes exist including changes to **placeId, address,** or **addressArray**	**200** Venue edited successfully - new data updated in database EXCEPT **placeId, address,** and **addressArray**	✓

Action	**Restriction**
Delete (Delete venue) - DELETE	Valid cookie provided - **userId** must match **ownerUserId**.

Test input	**Required result**	**Pass**
Invalid cookie	**403** Authentication failed	✓
Valid cookie - user does not own venue	**403** User does not own venue	✓
Valid cookie - user owns venue	**200** Venue deleted - venue and saved venues deleted - events deleted - related saved events, event discussions, event updates deleted.	✓

Action	**Restriction**
Read (Get venue page data) - GET	**venueId** provided

Test input	**Required result**	**Pass**
Invalid **venueId**	**404** Invalid Venue ID	✓
Valid **venueId**	**200** Venue data in response	✓

Events

Action	**Restriction**
Create (Create event) - POST	Valid cookie provided - **userId** must match **ownerUserId** of venue. **venueId** must exist. **name, venueId, date, endDate** all required

Test input	**Required result**	**Pass**
Invalid cookie	**403** Authentication failed	✓
Valid cookie - user does not own venue	**403** User does not own venue	✓
Valid cookie - user owns venue - venueId does not exist	**400** Invalid venue ID	✓
Valid cookie - user owns venue - venueId exists - missing required data	**500** Mongoose error thrown and caught - error message response	✓
Valid cookie - user owns venue - venueId exists - all required data	**201** Event created successfully - event saved to database - image saved if present	✓

Action	**Restriction**
Update (Edit event) - PUT	Valid cookie provided - **userId** must match **ownerUserId** of venue. New **venueId** must exist. **name, venueId, date, endDate** all required

Test input	**Required result**	**Pass**
Invalid cookie	**403** Authentication failed	✓
Valid cookie - user does not own venue	**403** User does not own venue	✓
Valid cookie - user owns venue - venueId attempted to edit does not exist	**400** Invalid venue ID	✓
Valid cookie - user owns venue - new venueId does not exist	**400** Invalid new venue ID	✓
Valid cookie - user owns venue - new venueId exists	**201** Event edited successfully - updated event data saved to database - new image saved if present and reference updated	✓

Action	**Restriction**
Delete (Delete event) - DELETE	Valid cookie provided - **userId** must match **ownerUserId** of venue.

Test input	**Required result**	**Pass**
Invalid cookie	**403** Authentication failed	✓
Valid cookie - user does not own venue	**403** User does not own venue	✓
Valid cookie - user owns venue	**200** Event deleted - event discussion, updates deleted - saved events deleted	✓

Action	**Restriction**
Read (Get event page data) - GET	**eventId** provided

Test input	**Required result**	**Pass**
Invalid **eventId**	**404** Invalid Event ID	✓
Valid **eventId**	**200** Event page data populated with associated venue data in response	✓