SPORTANA

Low-Level Design

kiwimango

LIBRARIES

FRONTEND

Angular:

MVC framework. 2-way data-binding.

HTML composed via partials.

Angular-UI:

View the application as a state machine rather than worrying about URL structure.

Bootstrap:

Responsive design + Predefined CSS classes.

BACKEND

Express:

Node Web Development framework

PG:

Object Oriented Database system for storing and handling user data

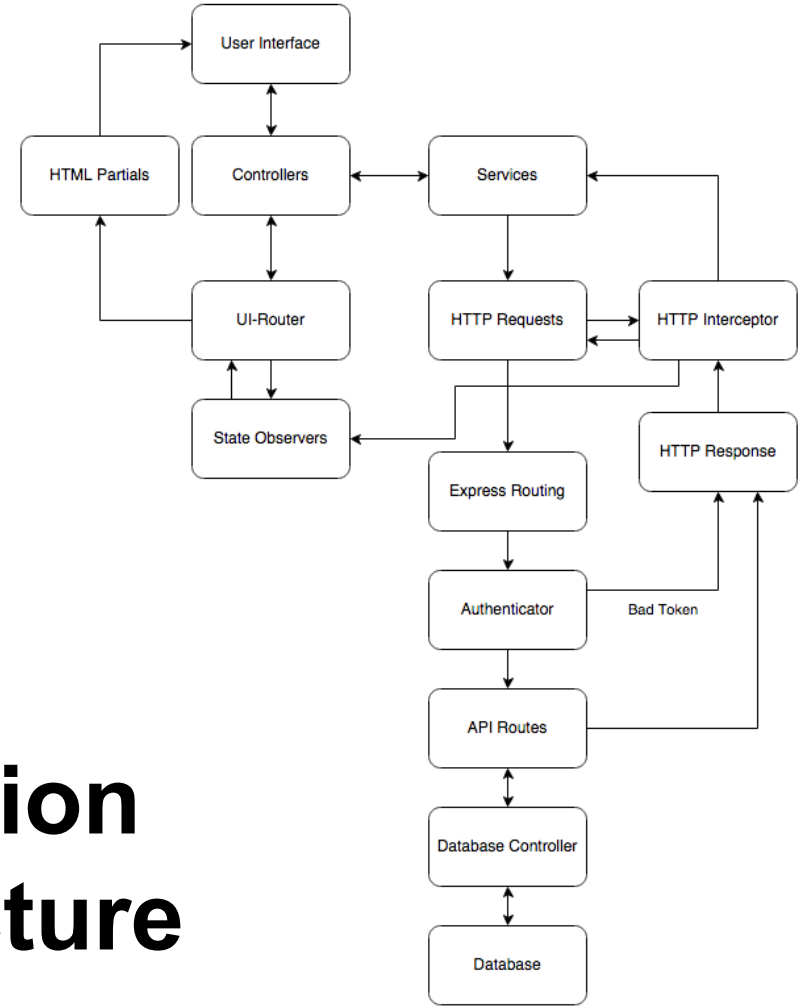
Base64URL:

Authentication token encoder

Crypto:

Generates random tokens

Application Architecture



@Brandon

High Level Overview - Front End

- UI-Router injects HTML partials into application view on state changes.
- DOM is manipulated via Controllers which provide 2-way data-binding.
- Changes are persisted by calling Services that expose HTTP API requests. Services make the calls to the backend API routes.
- User-authentication Service ensures application fidelity.
- State Observers watch for authentication errors and re-route when necessary.

@Brandon

Component Descriptions - Front End

Authentication Service: An authentication token is saved into a cookie via a Session Service on successful login. This token is passed on subsequent server requests to verify the user. If the backend responds with a mismatched token, then the user is redirected to login where they cannot access the application until they have logged in.

HTTP Interceptor: Auth tokens are appended to all requests via the interceptor. Failed token matches will be caught here on a server response and an AUTH_FAILED message will be broadcast where it is caught by the state observer where they will be redirected to login.

State Observer: When a user tries navigating to any page (except login/signup), their token is checked before granting them access. This step improves the user experience so that invalid requests are prevented before they can be sent.

@Brandon

Component Descriptions - Front End

Find Games (Queue): User provides game criteria: SPORTS, AVAILABILITY, LOCATION, AGE RANGE, NUMBER OF PLAYERS, and CASUAL or COMPETITIVE. A sidebar provides info and edit links for the preferences so far while the main content view allows the user to edit a specific preference at a time. The information is stored in a Queue Service that will make an API call to look for a game matching the criteria, or add the user's queue profile to the Database.

GAME FOUND → Join (drop from queue) or decline (stay in queue → Notified when game available)

NO GAME FOUND → Stay in queue OR Create your own game

Create Game: A user will be able to modify the information they entered for their queuing preferences, but will additionally set a start and end time for the game and a location. Here they may also choose to invite friends. Setting the game to PRIVATE will only allow friends to accept invites, but PUBLIC will open it up to all users who are queuing up. Once the user is finished, the game cannot be modified beyond adding additional friends. Creating will also send out friend invites.

@Brandon

Component Descriptions - Front End

Games List: The games list is a view where a user can view the games where they are currently joined as well as the notifications for being invited to a game. The games list is retrieved by making an http request to the server and then renders the information in the view. The same is true for the notifications except there is an accept and decline button. When a user hits “Accept” on any game notification, the game moves to the current games list below and then sends a post request to the server to add the game to the users list in the database. If a user click “Decline” then the notification is removed from the list and a post request is sent to remove the notification from the database.

The information that is displayed for each game includes the name of the sport, date, location, a link to the game itself, and how many people are going and invited. Likewise, the notifications include all of the above in addition to the player that invited the user.

@Alec

Component Descriptions - Front End

View Game:
A user can find the view of a game they are in from the Games List view. All of the information that is displayed on the page will have been rendered from a http GET request to the server. This page includes all of the information about a specific game including but not limited to the name, sport, who is going, date, and location. The page would also include buttons to leave the game and invite other players. Another main feature of this page is the chat board, this will be the main communication between the players of a game. For the implementation of the chat board we will be using websockets.

Friends: The friends list is a view where a user can look at a list of all their friends. The information to be rendered comes from an http GET request to the server and includes the name, profile picture, age, city, and link to the profile. A user can click on the link which will send the user to the corresponding profile view.

@Alec

Component Descriptions - Front End

View Profile: The view profile component allows one to view the details of another player. This includes name, profile picture, age, current city, ratings, and favorite sports. This page will also alter itself based on the relationship of the user that is logged in vs the user who's profile page it is. If it is your own profile, you will have a button that allows you to edit your profile. This will alter the page to have input fields for your profile details and save and cancel buttons. If the profile is of and friend, an unfriend button and possibly invite to game button will appear. If the profile is of a non-friend, a request friend button will appear.

Sidebar: Much like the search bar, once a user is logged in, they will see the sidebar on every page they visit(Mobile users will have a button on the search bar that expands/hides the menu). The sidebar is a very important feature of the site as it is needed to get from page to page since it handles the major navigation. It will have links to the dashboard, profile of current user, joining as game, viewing current games, settings, and a logout button.

@Ryan

Component Descriptions - Front End

Searchbar: Once a user is logged in, they will have access to the searchbar on the top of every page they visit. For our first version, this will only allow for searching players by name. A user can type in a name and press enter and then will be redirected to a search results page which is described in the next section.

Search Results: The search results page is accessed when when a user enters a name into the searchbar and presses enter. When the page is requested, it reads in the query parameter from the URL and sends an Ajax get request to the server. The server then returns a list of all matching users. Each one of these results will give a brief overview of a user (profile picture, name, age, city, sports they play) and when clicked will redirect to that users profile page.

@Ryan

High Level Overview - Back End

- Front End makes a request to a route handler
- The handler will authenticate the user through the authentication middleware
- The handler will make a request to the database through the database controller
- The database controller will access the database and query for information and return it to the handler
- The handler will wrap up the data into JSON format and return it to the front end

Component Descriptions - BackEnd

Index - responds with index.html page to be rendered by front end

Database Controller - responsible for all interactions with the database via RESTful API handlers

Component Descriptions - Back End

Authentication middleware - responsible for assigning unique authentication tokens to authenticated users and later deserializing those tokens to identify the user.

Component Descriptions - Back End

Friends - Responsible for interactions involving friends including getting a list of friends and removing a particular friend

Login - Responsible for logging in a user. Returns an authentication token when a user logs into Sportana successfully

Requests - Responsible for interactions involving friend requests, game invitations, and queue invitations as well as accepting and declining requests

Component Descriptions - Back End

Games - handles all interactions involving games such as creating a game, listing the games a user is attending and information about specific games such as location, time, and date.

Users - handles all interactions possible by a Sportana user such as user profiles, creating a new account, rating players, etc.