

Milestone 4
Purple Nut
Team 6

1. List of features
 - a. Completed Features
 - i. Create user accounts
 - ii. Create events
 - iii. Sign Up for events
 - iv. Filter Events
 - b. Features to add
 - i. Follow users
 1. Sort by popular with those you follow
 - ii. Notifications
 - iii. Host on AWS
 - iv. Google maps API
 - v. General HTML improvement
2. Architecture diagram
 - a. Django handles the management of all parts of our architecture using a 'View-Model-Controller'
 - b. Django routes requests with view functions. View functions pull data from the database (currently sqlite, will migrate to postgres) and will pass the data to templates. Templates are HTML files with special Django syntax to render the variables passed to it.
3. Front end design
 - a. Need a datetime picker in create event form
 - b. Should use a modal for listing attendees (currently an ugly table)

Front End Design

Create Account

To access the main features of our website, the user needs to make an account. If they do not have an account they can not access features such as signing up for events and creating events. So to have access to those features, they need to create a profile with the fields below.

Create User

Username*

ct

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email*

ct@gmail.com

Password*

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation*

Enter the same password as before, for verification.

Sign Up

Already have an Account? [Log In](#)

View events, sign up for events, edit events

Once logged in, the homepage of the website displays the list of upcoming events that have been posted by a user. Each listing displays the details of the event. If logged in, the user can sign up for any event by simply clicking the signup button. Also, if they are the creator of the event they have the ability to edit the event, and an edit event button will appear. The user in this instance had not created any events, so that is why no edit event button appears.



Events

search for events

☐ show old events


Search

Morning Run [Signup](#)

Running

posted by:  admin

 The Rec Center

 Jan. 1, 2020, 8:20 p.m.

Is your fridge running?

Number signed up: 2

[admin](#)


[athlete](#)

Created: Oct. 23, 2019, 1:09 a.m. Last Modified: Nov. 8, 2019, 12:41 a.m.

View Profile

If the user clicks the profile picture in the top right corner of the homepage, they have the ability to view their profile details.

Edit Profile



ct
ct@gmail.com

Signed Up Events

[Morning Run](#)

Edit Profile

If the user wishes to change their profile name, email, or password they can do so by clicking the edit profile button on top right corner of view profile. This view will be presented to them.

Edit Profile

Username*

ct

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email address

ct@gmail.com

Password

No password set.

Raw passwords are not stored, so there is no way to see this user's password, but you can change the password using [this form](#).

Image:Currently: [default.jpg](#)

Change: No file selected.

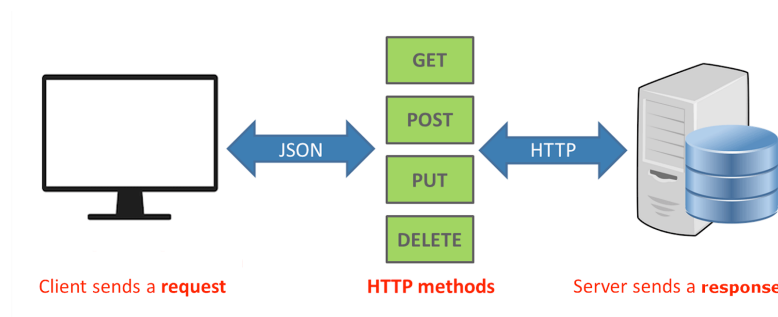
4. Web services design

- Google maps API to generate a thumbnail for each event
- Amazon Web Services

1. Message format: SOAP, XML, JSON, etc.
2. Request syntax: URI, Parameters & Data types
3. Actions on the server: named methods, HTTP verbs
4. Security: authentication (username & password)
5. Response format: SOAP, XML, JSON, etc.

AWS uses REST API - an abstract concept that defines the characteristics and features you would find in a web service request built according to the REST style

- Uses uniform interfaces. Resources are handled using http POST, GET, PUT, DELETE operations
- Stateless. Every request is an independent request. Each request from client to server must contain all the information necessary to understand the request



Dynamic Web Server - Consists of a static web server plus extra software, most commonly an *application server* and a *database*. It's dynamic because the application server updates the hosted files before sending them to your browser via the HTTP server.

Database Design

- Currently we are using sqlite3 but will transfer our database over to postgresql
- 1 to 1 between users and profiles
- many to many between users and events

Relations

i. Users

1. ID
2. Django class handles
 - a. username
 - b. password
3. Profile extends user to add profile picture and followed users
4. email
5. name

ii. Events

1. id
2. Name
3. Description
4. Location
5. Time
6. Category
7. author_id

iii. Profile

1. id
2. image
3. user_id