Crazy Joe Speaks Frogs:: Kevin Wang (PM), Shreya Roy, Faiyaz Rafee, Justin Mohabir
SoftDev
P01 -- ArRESTed Development
2022-12-4
time spent: 20 hrs

Target ship date: W **2022-12-21**

Design V.0 for GOOGOL MAPS!

Goal:

Create boot-leg Google Maps. An address inputted from the client is then converted into coordinates to be passed into the yelp restaurants API and overpass API. Returns nearest restaurants and amenities within 5000 meters of the client's inputted address. Also allows users to update the maps with their own amenities and restaurants, and an admin to approve it!

Roles:

API stuff: Shreya
Frontend: Justin
DB management: Faiyaz
PM & moral support: Kevin

Frontend Framework Chosen: Bootstrap

- Why:
  - Bootstrap is easier to work with & looks cleaner.
  - Bootstrap containers are epic
  - Foundation is bad :D
    - Using only flexbox rather than grid/flex grid/etc
- How:
  - Containers to make sure everything is neatly organized in accordance with screen size
  - Forms to accept user input for a location or address

APIs Utilized:

- https://github.com/stuy-softdev/notes-and-code/blob/main/api_kb/411_on_PositionStack.md
  - Input Address -> output coordinates

    - 25000 quota & key
- https://github.com/stuy-softdev/notes-and-code/blob/main/api_kb/411_on_Yelp.md
    - Input coordinates -> output nearest restaurants
    - Maximum of 5,000 calls/day
- https://github.com/stuy-softdev/notes-and-code/blob/main/api_kb/411_on_OpenStreetMap_overpass-api.md
    - Input coordinates -> output nearest transit & places
    - no quota no key
- https://github.com/stuy-softdev/notes-and-code/blob/main/api_kb/411_on_geoapify_static_maps.md
    - Input coordinates -> output static image of map
    - 3000 per day quota & key

DB:

Authentication

Username	**Password**
devo1	hehehaha
devo3	gabagoo

Past Searches

username	**Past Searches**
ThonnySinclair	345 Chambers Street
ThonnySinclair	1 Empire Avenue

Admin Panel

username	**admin**	**isRequested**	**isRejected**
admin	True	False	False
devo1	False	True	False

Amenities

id	**name**	**latitude**	**longitude**	**proposer**	**approvedby**

1	Tribeca Bridge	3	3	kevin	
2	Effiel Tower	5	5	admin	admin

Restaurants

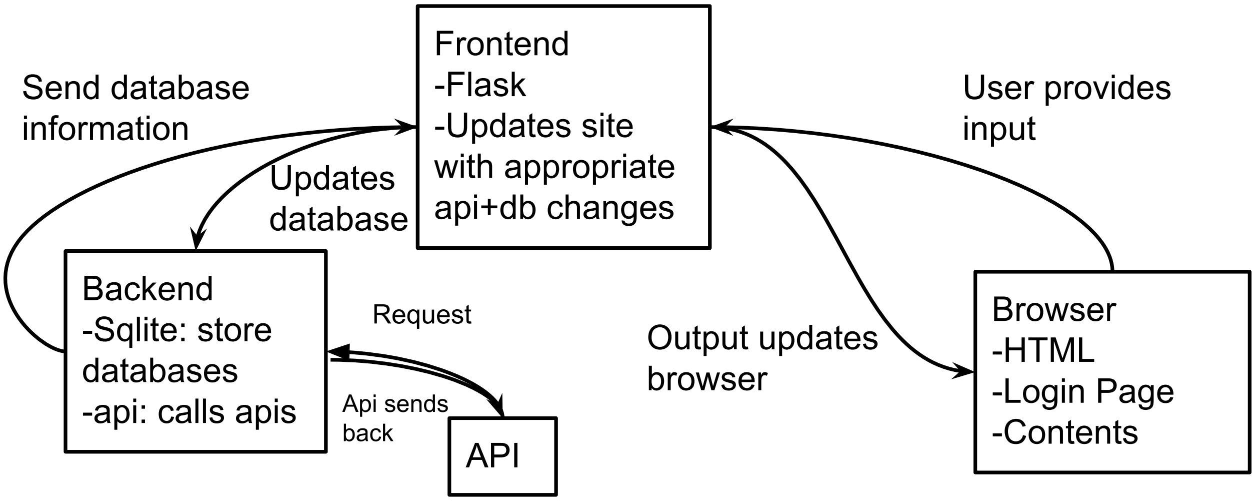
id	name	stars	imgname	latitude	longitude	proposer	approvedby
1	Ferries	4	ferries.jpg	3	3	kevin	
2	Terries	5	terries.jpg	5	5	admin	admin

Program Components:

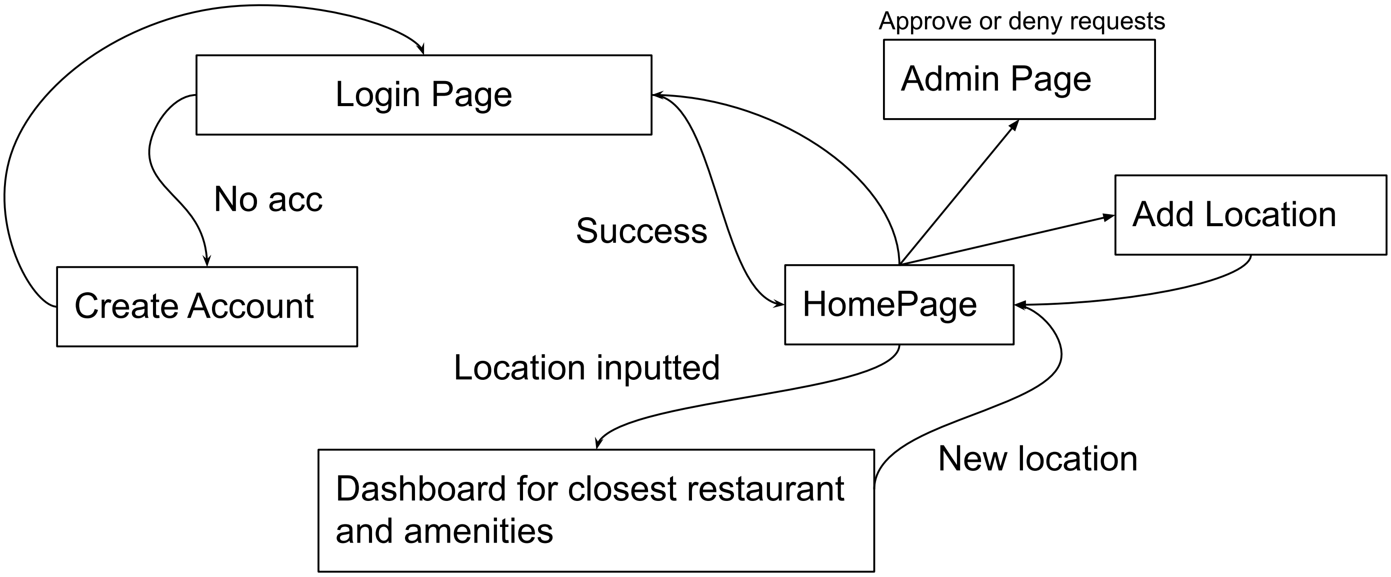
- __init__.py
  - Where the flask magic happens
  - Renders templates
  - Uses database.py to interact with the db and handle username/password authentication, as well as past searches
  - Takes in user changes -> sanitizes it from methods in api.py -> retrieves data from apis from methods in api.py -> renders templates from it
- database.db
  - A table for usernames and passwords
  - A table for storing username and their searches
  - A table for Admin Panel requests
  - A table for Amenity requests
  - A table for Restaurant requests
- db.py
  - Methods for updating, viewing, and creating new entries in the db
  - Responsible for admin authentication
  -
- api.py
  - retrieving info from apis

   - Cleans and parses data from api response to be used in `__init__.py`

Component Map:



Site Map:



Templates:

login.html

- Bootstrap form asking for username and password
- Displays error if incorrect inputs

register.html

- Bootstrap form asking for username and password
- Displays error if the username is already taken
- Password/username checking

home.html

- Has a nav bar
- Shows previously searched locations
- Bootstrap form for navigating to a new location
    - Accepts: address (best) / place name (worse)

dashboard.html

- Has a nav bar
- Map showing the relative directions of restaurants, transit/amenities. (relative to the central point of the searched address/location)
- Names of amenities on the side
- Restaurant carousel on the side
- Relevant cool statistics & data included in the apis

adminPanel.html

- Approve or deny account requests for admin
- Approve or deny added amenities or restaurants

add.html

- Request a new amenity or restaurant to be added to the map!

requestAdmin.html

- Regular users may request administration

navbar.html

- Navigation bar present on all pages except for login/register.

Flask:

__init__.py

/home:

- Access argument username and all searches
- Display all search results in home.html template

/login:

- Access arguments username, password
- Checks with form inputs
- Logs in/error message if incorrect username/pswd

/register:

- Accesses arguments username, password and appends that data to users table
- Ensures that username/password meets requirements

/dashboard:

- Shows closest locations
- Shows relative distances to restaurants and amenities on a map
- Shows restaurant names and information
- Cool map with relative spaces

/logout:

- Signs user out of their account

/admin:

- If already an admin:
    - Approves other admins
    - Approves amenities
    - Approves restaurants
- Otherwise, request to be an admin
- Directs users to either admin panel or request admin page, depending on their status

/add:

- Directs users to add their own amenities or restaurants.