RedDacks: Grace, Nahi, Sophia, Jackson
P#01 - ArRESTed Development
SoftDev1 Pd9
2019-11-18

RedDack Daily

Overview:

We are creating a site that displays recent information about a location. After searching for a location, a user can find the following: crime data, current news, nearby ticketed events, and upcoming holidays. Information will be pulled from the API's every time a new location is searched and will be stored in the databases. Timestamps will also be stored for each search, and location information will be updated only when a certain time frame is passed (in order to not surpass quotas). The appeal of our website stems from the human interest to keep up with news, whether it's actual news, pop culture, etc.

Our final goal is to use the six APIs listed below, two of which are nested within one topic. Using Foundation and HTML templates with Jinja to build our website, we would make an interactive site stemming from Flask and Python for registered users to look things up on, like Google on a smaller scale. As for search queries, users would be allowed to choose which API they want to search from, as well as search from any combination of them. Results listed would have been pulled from either the database or the API (and if pulled from the API, stored into the database). With primary functions in mind, the first stages of the website would allow users to search by location, plugging their query into the API URL and pulling the data.

APIs Overview:

- Crime Data API
    - https://crime-data-explorer.fr.cloud.gov/api
    - Key needed
    - JSON file returned using state and city to search (FBI data only for USA)
    - Demographic data on arrests, victims, preliminary report data etc
- News API
    - https://newsapi.org/
    - Key needed
    - JSON file returned using country and/or keyword to search (other functions also available)
    - Title, author, description, url, image, and content returned
        - https://open-platform.theguardian.com/documentation/ The Guardian
        - https://developer.nytimes.com/ New York Times
        - Above are two more trustworthy sites for news, which users can choose to browse instead

- Ticketmaster API
    - https://developer.ticketmaster.com/products-and-docs/apis/discovery-api/v2/
    - Key needed
    - JSON file returned using locale or country, multiple query parameters can be used
    - Returns event specific data with price ranges, description, accessibility, external links, etc
- Calendars/Holiday API
    - https://calendarific.com/api-documentation
    - Key needed
    - JSON file returned using location, however more specific queries can be made by using parameters such as month, day, or type of holiday
    - Returns name, description, and date of holiday

These are for fun
https://random.dog/
https://aws.random.cat/meow

User Experience:

Users would need to login or signup to be able to access our database information. Once the session begins, the homepage will display the search bar and filter options in which the user can select with APIs they want to search from, and type in a location that they want to learn about. The search results returned would be sorted into the different APIs, and users would be able to click into these results to pull from the API or the database.

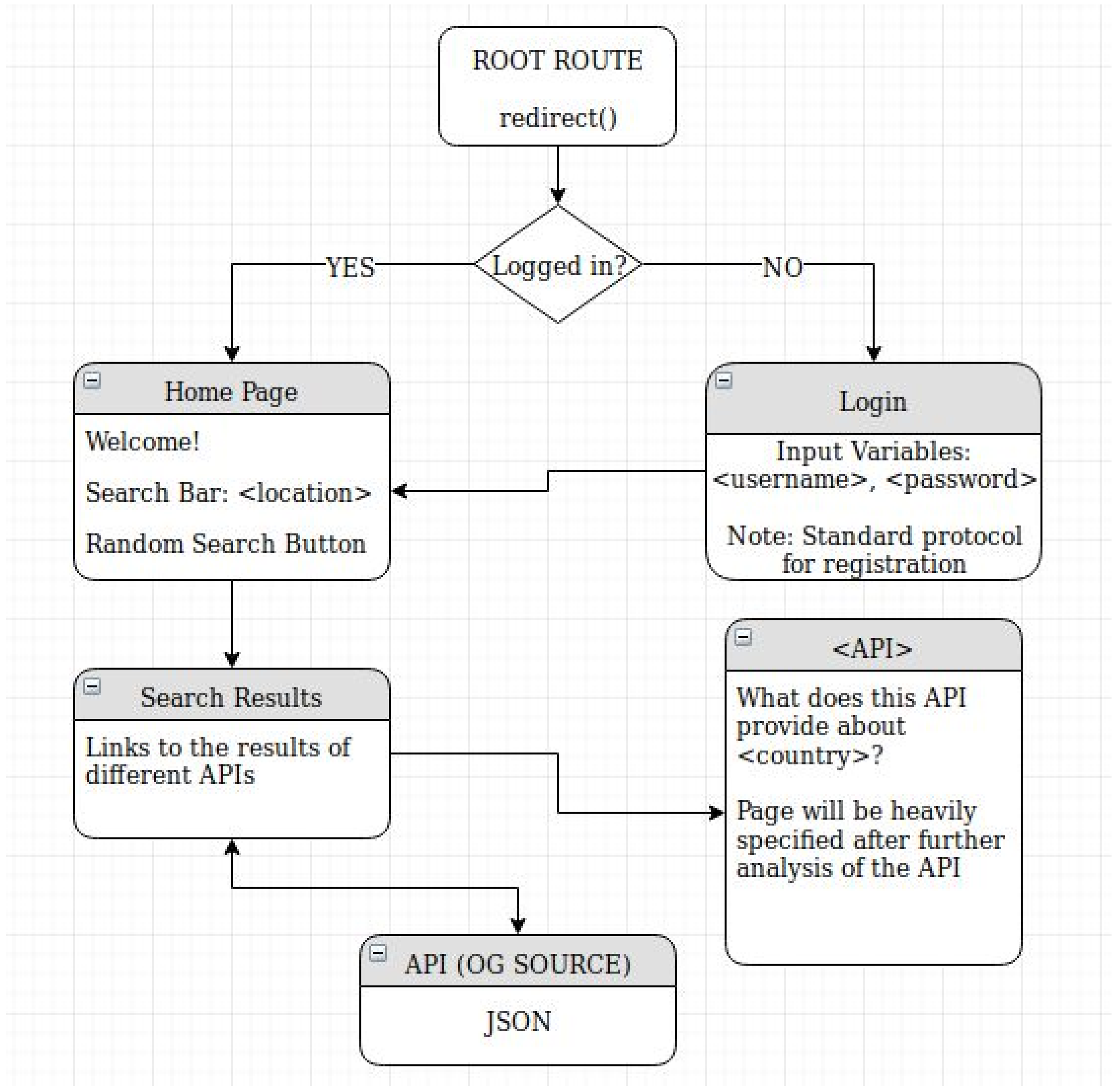
When the user searches something that has been searched before in the last 24 hours, the database will have cached this information and the template will be rendered by retrieving information from the database. Otherwise, a new call to the API will be made, and this new data will be stored into the database and then rendered onto the page for the user to view.

Every page will have at least one external page in order for the user to get more information, and if they don't log out before going to one of these external links, their session will still be there when they get back.

Roles:

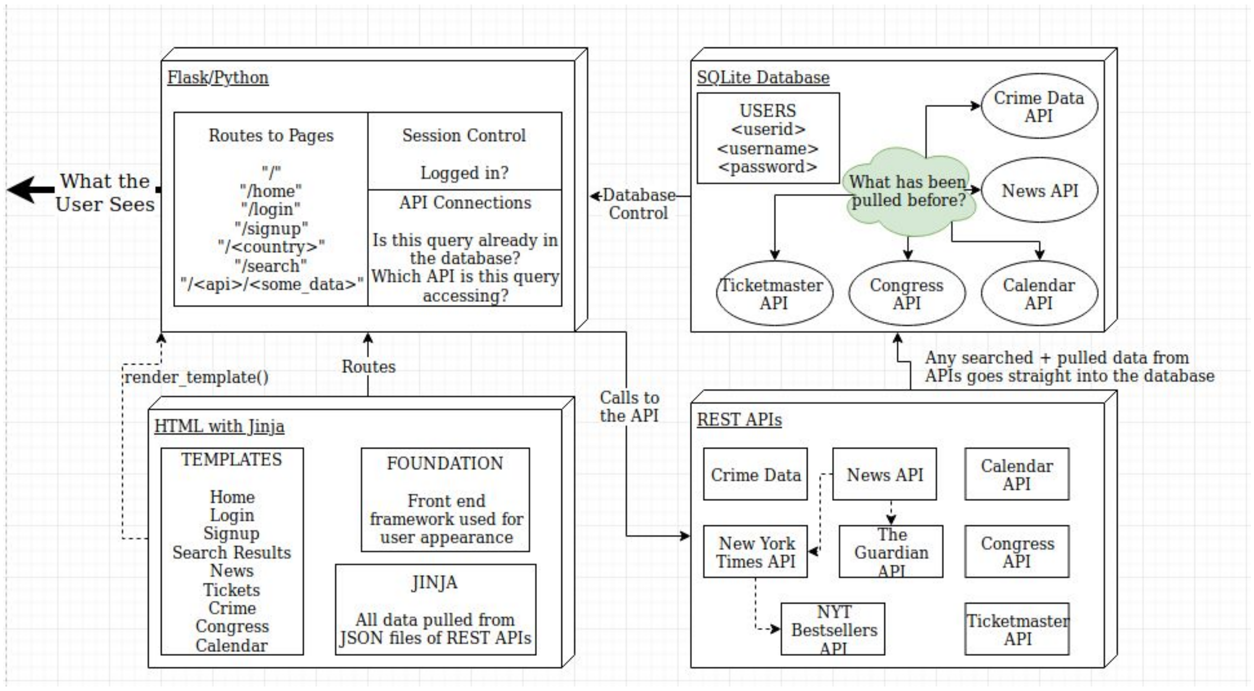
- Grace:
    - Project Manager
        - Revise design doc
        - Assign tasks
        - Facilitate communication
    - Templates using Foundation
        - Make a template that will store and format the title, headers, images, and forms to be displayed on a given page
    - Troubleshoot issues and complete minor coding tasks as necessary
- Nahi:
    - Route Functions
        - Renders the template for each page requested
    - Building Forms
        - Implement old code for login / signup
        - Handle buttons in the templates (Home button on each page, submit buttons etc)
    - Creating Cards (bit.ly/apikb1920)
        - Create cards for each API used to store in the Knowledge Base
- Sophie:
    - Initialize Database
        - Make a table of users that will store the usernames, ids, passwords of each user
        - Make a table of favorite places for each user
        - Make a table for each API
    - Database Operations Module (Insert to Table, Edit a Row, Create a Table)
        - Facilitate any changes that the user will request, such as adding a new row in a database when the user pulls information from the API to store that information and facilitate faster access in the future
- Jackson:
    - API Keys
        - Obtain access to keys for all APIs
    - Pull info from APIs
        - Open the url for each API every time a call is made and pull and store needed information from the API in the appropriate database
        - Optimization: keep times stored in a DB to check whether or not to make another pull request (in order to not fill quotas)

Site Map:


ROOT ROUTE
redirect()
Logged in?
YES
NO
Home Page
Welcome!
Search Bar: <location>
Random Search Button
Login
Input Variables: <username>, <password>
Note: Standard protocol for registration
<API>
What does this API provide about <country>?
Page will be heavily specified after further analysis of the API
Search Results
Links to the results of different APIs
API (OG SOURCE)
JSON

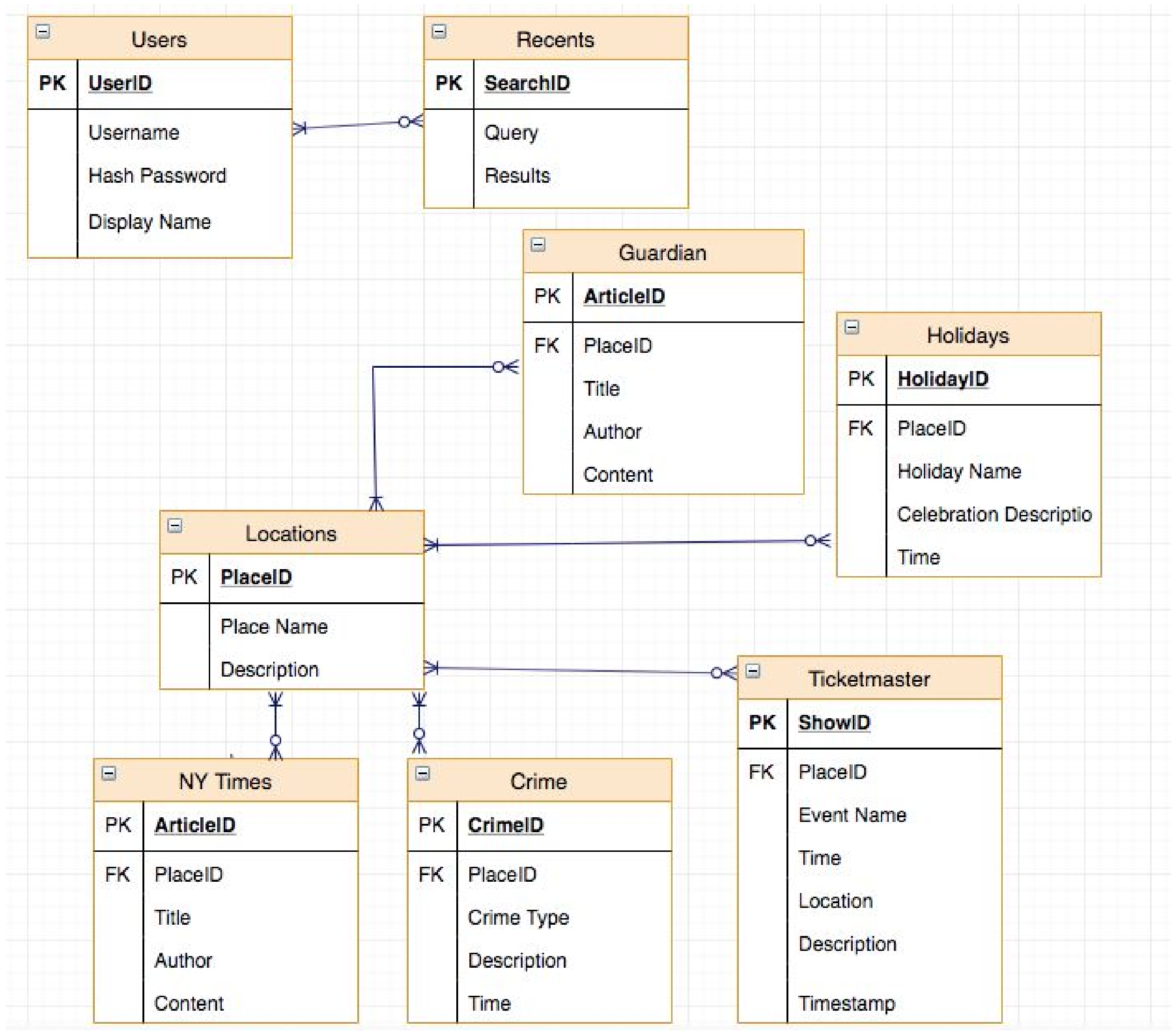
Note for team: Sitemap up for major revision after delving further into each API

Component Map:



Each of these components will be dealt with by a different member of the team, so that in the end they can be pulled together to form the entire website. Communication will be required between SQLite and the API, as well as the HTML and Flask code in order to render templates correctly.

Database Diagram:



Note: All database tables would need to be linked with LOCATION, in order for the search query to function correctly.

The **Primary Key** for each table shows that each entry is required to have this item; it cannot be null or empty.

The **Foreign Key** for the tables connected to an API is the LOCATION item that will connect each entry to the search results.