RedDacks: Grace, Nahi, Sophia, Jackson
P#01 - ArRESTed Development
SoftDev1 Pd9
2019-12-04

RedDack Daily

Overview:

We are creating a site that displays updated news tracked by location. Searching by location, the user will be able to find news articles from that location from multiple sources, mainly The Guardian and the New York Times. Put together with the News API, the website would pull information from these sources every time a location is searched that has not already been searched in the last 24 hours (in order to not surpass quotas). This information will be stored in the database.

Our final goal is to use the four APIs listed below to create a collective news site. 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 by location. As for search queries, users would be allowed to choose what results they would like to see, based on category (i.e Sports). 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:

- 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
- The Guardian API
    - https://open-platform.theguardian.com/documentation/
    - Key needed
    - JSON file returned using country to search
    - Title, author, url returned, ***content not
- New York Times API
    - https://developer.nytimes.com/
    - Key needed
    - JSON file returned, has **multiple** APIs
        - Article Search, Books, Most Popular, Geographic API

- Countries API
    - https://restcountries.eu/
    - No key needed
    - JSON file returned with country name, code, border information, flag etc
    - Used in this project to establish country codes (i.e Canada = CA) and provide background information on countries to users
    - Note: All countries from this API are pulled **once a day**, the first time the website is loaded that day

This is for fun
https://random.dog/

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 in which the user would type in a location that they want to learn about. The search results returned would be sorted into the different categories returned by our APIs, such as Sports or Technology. Users would be able to click into these results to view specific articles.

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.

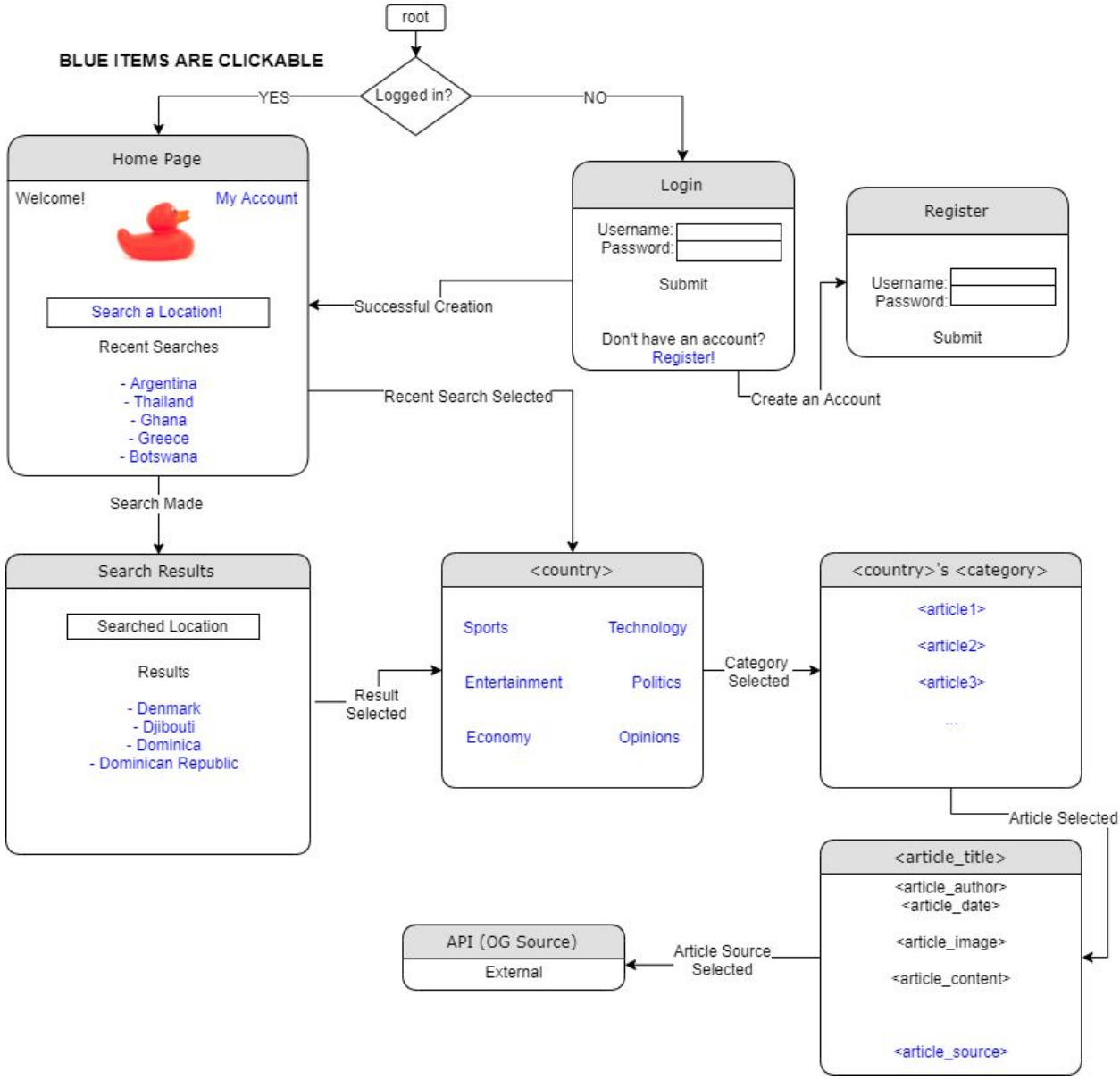
Every page will have at least one external page in order for the user to get more information, and if they don't log out before going to one of these external links, their session will still be there when they get back.

Roles:

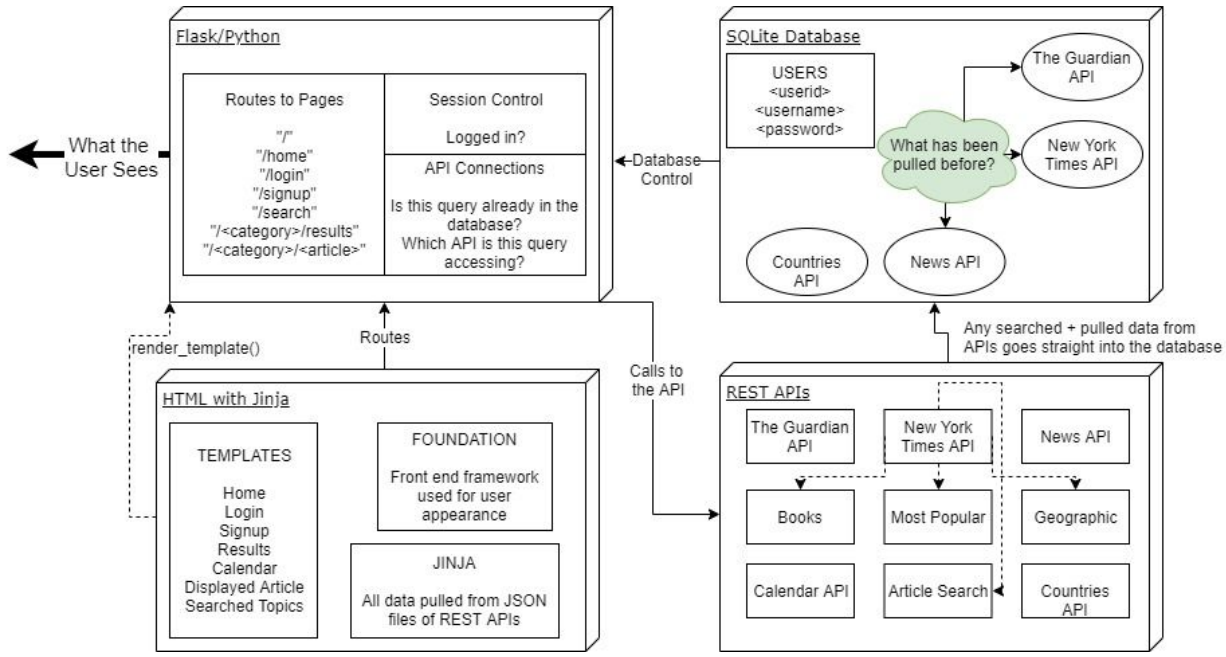
- Grace:
    - Project Manager
        - Revise design doc
        - Assign tasks
        - Facilitate communication
    - Creating Cards (bit.ly/apikb1920)
        - Create cards for each API used to store in the Knowledge Base
    - Facilitate the final steps of building the app, adding all components together to form the large “app.py” python file
    - 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 templates using Foundation
        - Write the HTML code for each template using Foundation as the front end framework
- 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:

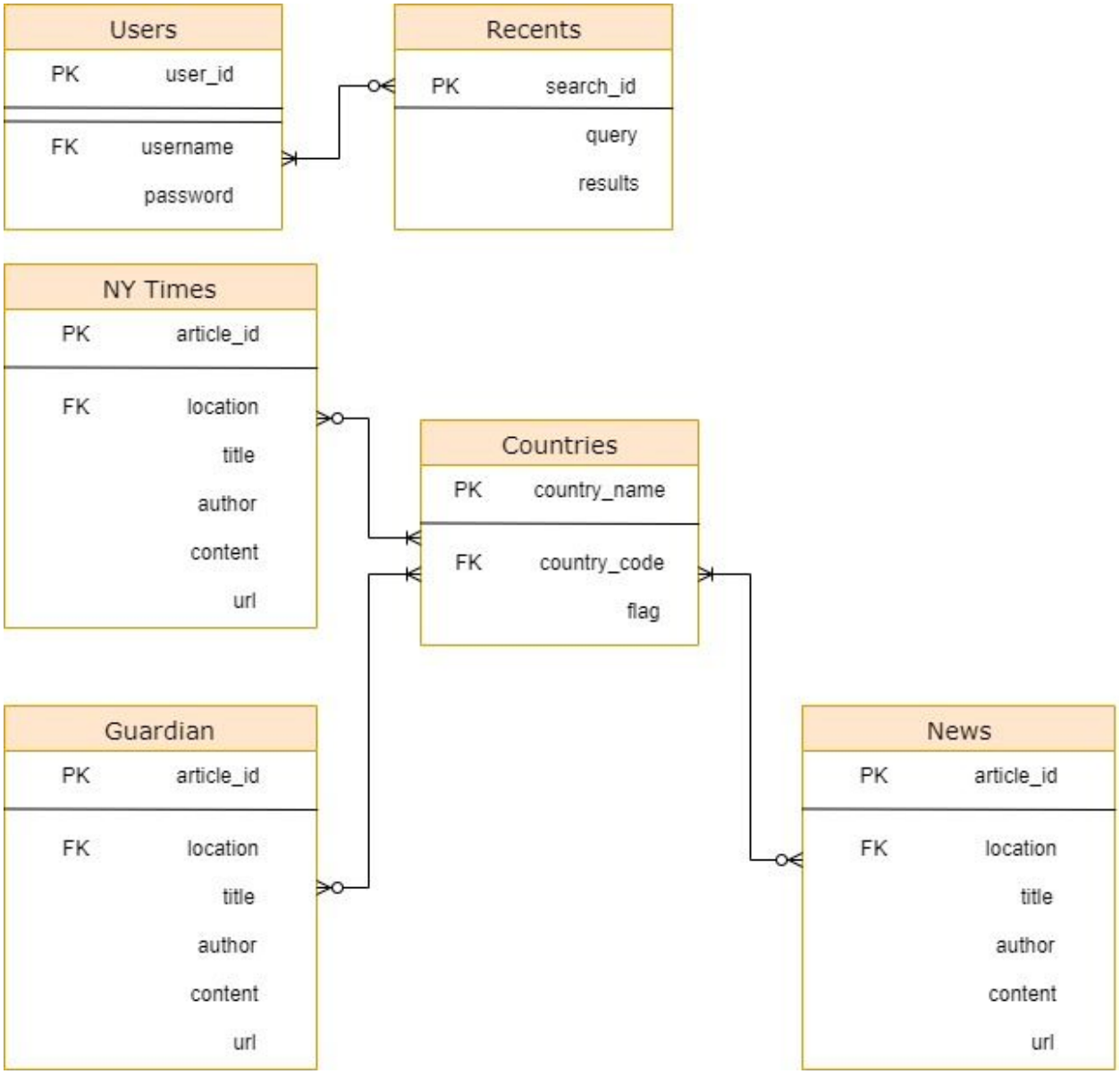


Component Map:



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.