Team Atmosphere

Alice Ni, Moody Rahman, Joseph Yusufov, David Wang

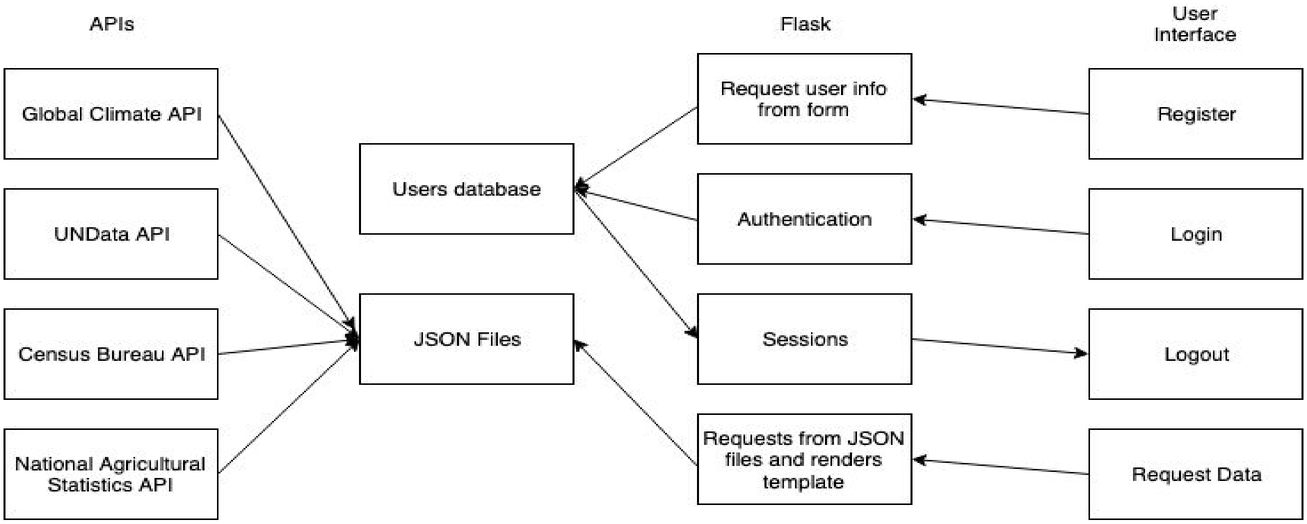
ROLES:

Alice - PM, bootstrap
Moody - Database, python
Joseph - API, bootstrap
David - Python, front end

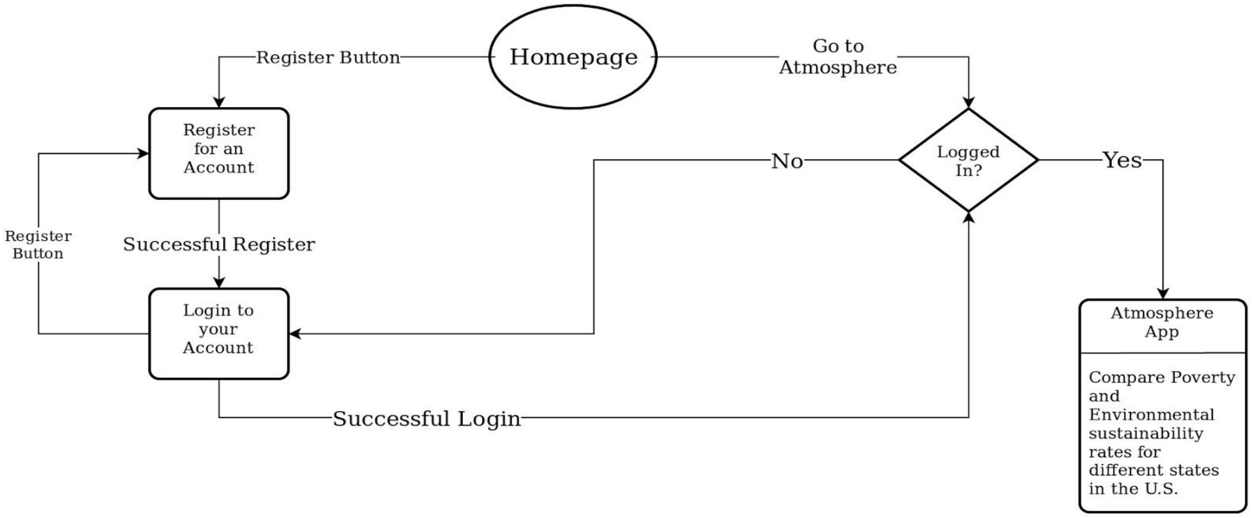
Functionality

- Interactive website that allows users to view and compare data from different states
- A few built-in data variables, stored as JSON files in a JSON file folder
  - Population per state
  - Poverty rates
  - Carbon emissions
  - Crop yield
- Users can choose the variables and the states they want to compare, making a request to a Flask app that reads from the JSON files. If the variables are not in the folder, the app makes a request to the API that has the specified data.
- The data will be stored in databases relative to each API used. API information will be cached into the databases as they are requested by users.
- Scattergrams and charts will be generated by the parameters specified by the user. Displays information in a table numerically as well.
- Sliders on the scattergram allow the user to choose which years they want to see. They can move across time periods to see how data trends shifted.
- Implements bootstrap
  - Navbar
  - Sliders
  - Scattergrams
  - Charts
  - Fixed sidebar
  - https://getbootstrap.com/docs/4.0/examples/dashboard/ - good example of what ours should look like

Component Map



Site Map



Atmosphere Site Map

Alice Ni, Moududur Rahman,
David Wang, Joseph Yusufov

Database Layout

Users

username TEXT	password TEXT
“jimbob”	“cooljoe23”
“hamlet”	“macbeth”

username: displayed name for each account, entered by the user
password: entered by the user

Census

Agriculture

Climate

Front End

- landing.html
  - User **must** login or register for an account before viewing the site
  - Buttons to register or login
- login.html
  - Form for submitting an existing username or password
- register.html
  - Form for creating an account
- home.html
  - Home page that displays real-time data for the U.S. as a country (total U.S. population, carbon emissions, etc.)
  - Option for user to select a single state and view all the available statistics for that state via a form at the bottom of the page
  - Option for user to select and compare two states via a form at the bottom of the page
  - Submit button brings the user to another page that displays all the state-specific data
- statesData.html
  - Page that displays the specified state(s) and data in a table
  - Sliders that allow the user to move around different years
  - Generates a scattergram showing relation between selected data
  - Form that lets the user change between different states / selected data types
  - A log for the user’s most recent data comparisons (needs a database)

    - If user wants to request specific data, they can enter it into a form
        - If data does not exist within APIs, flash a “data does not exist” message
        - If data does exist in APIs, add data to the JSON file folder

Back End

- app.py
    - Login system
    - Registration system
    - Routes
        - “/”
            - Renders “landing.html” if user not logged in
            - Renders “home.html” if user logged in
        - “/login”
            - Renders “login.html”
            - Redirect to “/home”
        - “/register”
            - Renders “register.html”
            - Redirect to “/”
        - “/home”
            - If user is not logged in, redirect to “/”
            - Renders “home.html”
            - Displays user specific info
        - “/states”
            - Renders “statesData.html”
            - Displays user’s recently viewed comparisons
        - “/logout”
            - Removes user from Sessions
            - Redirects to “/”

Functions

- search_state()
    - @param: name of state
    - @param: name of requested data
    - Returns one specific data for one specific state
- get_data()
    - @param: requested data variable
    - If not found in existing JSON files, makes request to API for relevant data
- login()
    - @param: username
    - @param: password
- register()

- @param: username
- @param: password
- Cannot register a username already in use

Important Links:

National Agricultural Statistics API - https://quickstats.nass.usda.gov/api

Key: 79900EE9-743F-3CBA-AD8A-26063F956065

https://quickstats.nass.usda.gov/api/get_param_values/?key=79900EE9-743F-3CBA-AD8A-26063F956065&param=sector_desc

Example of GET request of all the corn produced by Virginia since 2012

http://quickstats.nass.usda.gov/api/get_counts/?key=79900EE9-743F-3CBA-AD8A-26063F956065&commodity_desc=CORN&year__GE=2012&state_alpha=VA

Returns: {"count":13048}

Global Climate API --
https://datahelpdesk.worldbank.org/knowledgebase/articles/902061-climate-data-api

Working request:

http://climatedataapi.worldbank.org/climateweb/rest/v1/country/mavg/tas/1980/1999/FRA

General format:

http://climatedataapi.worldbank.org/climateweb/rest/v1/country/**type/var/start/end/ISO3[.ext]**

UNData API -- https://unstats.un.org/SDGAPI/swagger/

https://unstats.un.org/SDGAPI/v1/sdg/Indicator/Data?indicator=1.1.1&areaCode=1&timePeriod=2017&dimensions=%5B%7Bname%3A%22Age%22%2Cvalues%3A%5B%2215%2B%22%5D%7D%2C%20%7Bname%3A%22Sex%22%2Cvalues%3A%5B%22BOTHSEX%22%5D%7D%5D

M49 codes: https://unstats.un.org/unsd/methodology/m49/

Census Bureau:
https://api.census.gov/data/2018/acs/acs1?get=NAME,group(B01001)&for=us:1&key=07626e3b3578edd0e55ba15cb38770a85aedd31d

https://www.census.gov/data/developers/data-sets/acs-1year.html

Bureau for Economic Analysis:
https://apps.bea.gov/API/signup/index.cfm