NSA_Advice

Tyler Huang, William Lin, Coby Sontag, Hilary Zen

Framework:
FOUNDATION

Roles
Tyler: Back end
William: Project Manager
Coby: Back end
Hilary: Front end

APIs Used
Open Weather Map
Key: da19d101a993403bd4ab9a3284ec0f0d
Google Cloud
Key: AIzaSyA4Rb84cl3x6kVw6AZuPrhQgP9teGyPN6A
Location IQ
Key: 5d138e217570a3

Core Functionality:
Login system, and accounts that store user information
Return data collected through APIs like age range, gender, city, weather, job, election data. Plots location on a map...

Databases
Users

id	username	password	firstName	lastName	location	address
Integer	Text	Text	Text	Text	Text	Text

- Information used to keep track of accounts
- Stores the information that the user gives us, like their name and email, along with data that we collected through the APIs, like their age
- Not all fields are required. Blank fields will just have a NULL placeholder.

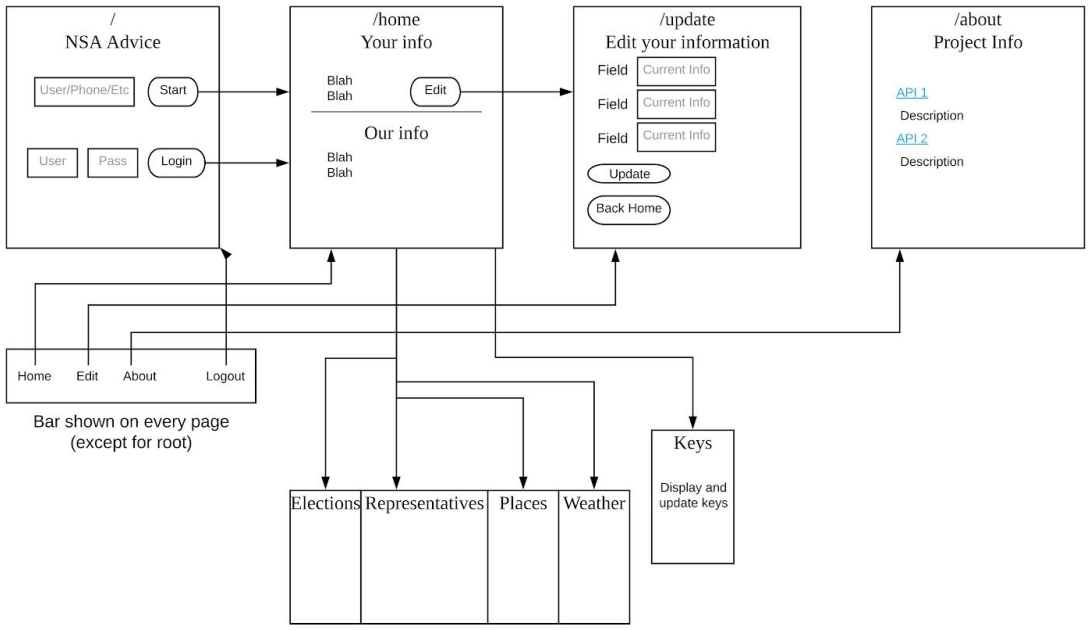
Api Keys

id	openWeather	googleCloud	locationIQ

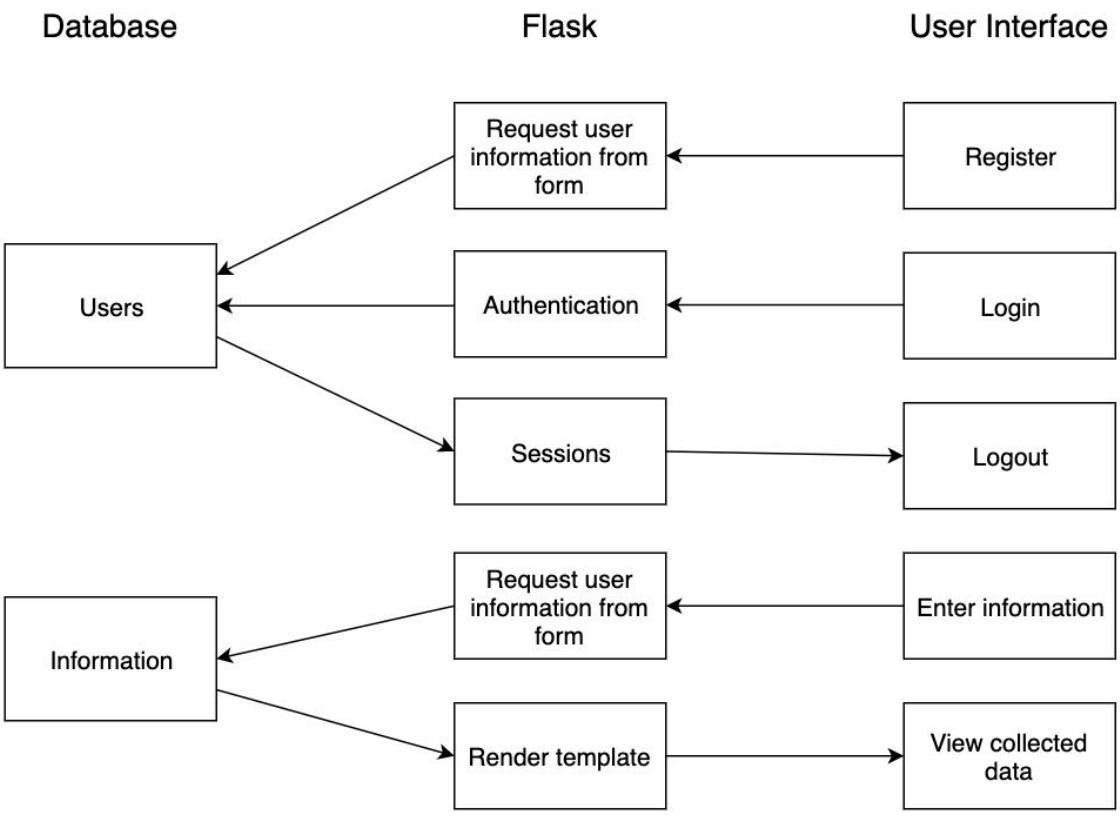
Integer	Text	Text	Text

- Stores the API keys of users
- Not all fields are required. Blank fields will just have a NULL placeholder.

Site Map



Component Map



Task List

- Create pages for user to login and register (William)
- Set up authentication system (Tyler)
- Create form for user to enter information about themselves (William)
- Store info in database and use it in our three APIs (Coby)
- Build user's homepage, where they can view the data taken from the APIs (Hilary)
- Take the user's information from the database to the template (Tyler)
- Make a form for users to edit the information they have provided (Hilary)
- Update their changes in the database, using the APIs again (Coby)

Egoless Programming

- Although the industry looks for “detached” programmers to enjoy being left alone to be creative, they are very attached to the programs they write. Our code often becomes an extension of themselves, and when there are errors we blame them on other factors, or even ignore them completely.
- Open, shared programming > secretive, possessive programming
- The article explained very clearly why it can be so difficult to ask for help and debug code in groups. We were shocked by the examples throughout the text that showed how new programmers working together could quickly surpass others who worked alone, even though they had years of experience.
- We learned that it isn’t enough to just have one person check code, since they also become attached to the debugging work they have done. To be most effective, even though all of us are focusing on separate areas of the application, we will review all the code written each day, making sure that we understand it and checking for errors or things that could be done better.
- We will also work on not viewing our errors as a reflection of our programming ability. When reviewing code, we will make sure to give constructive feedback and think of

concrete solutions together, rather than simply pointing out errors and leaving each other to struggle alone.