Klaviyo Weather App:
Kaveesha Shah
ks2379@cornell.edu
5714572734

Project: WeatherAppKaveesha
App: weather
URL: http://localhost:8000/home/

Files and description:

1. Urls.py:
    a. This has the home/ path defined which points to the home method in views.py
2. Models.py:
    Defined the 2 models: User and Location. Location consists of a field called City. User has 2 fields-email_id and foreign key to a Location object.
3. Forms.py
    1. As a part of this file, I have used the forms module of django to create a SubscriberForm which has 2 fields - email which is of type EmailField and city which is of type ModelChoiceField since we need a dropdown for the form and the drop-down options are basically the cities in the Location model. We obtain this by passing a query set to the ModelChoiceField objects.
    2. By using the form module, the validation is handled by it and hence we don't need extra checks in our codebase.
4. Home.html
    1. I have used the form attributes of the form passed as context through the render method of views.py
    2. I have also included messages for successful subscription.
    3. Once the subscription is successful, it blanks all the fields and is ready to accept other values.

5. Sendemails.py
    1. I have created this file as a command and hence can be run as python manage.py sendEmails. I have created an email id klaviyoweatherapp6@gmail.com . Please use klaviyo@weather as the password.
    2. As an enhancement, I first ran across all the locations that have been referenced by Users and kept them in a dictionary. This is helpful because if 3 users have the same location, we won't need to call the API thrice.
    3. I have used the WeatherBit api to get the weather information. I have used the weather codes to see the condition like cloudy skies, snow etc. Precipitation can be snow, rains and includes thunderstorms as well. A lot of codes satisfy this condition namely range200,300,500,600 and 900. Hence I have used the weather code%100 and matched it with 2,3,5,,6 and 9.

4. I have created a customized email message for each which consists of the current and the forecasted temperature(please note that the temperature is in Celsius) and the weather description. I have also included a reference to the Klaviyo website and logo.

5. I have used the SMTP server for sending the mails.

Other topics:

1. Scalability: In this app, the access of models(databases) will be a bottleneck. Apart from enhancing the hardware, we can enhance this by using indexing and caching frequently accessed data.
2. Security: I have tried covering some security concerns in this app by making the API key an environment variable. We can further encode the accepted password for connection to the mail server.
3. Re-usability: We can create a base.html which will be extended by all other pages in the app. We can also create a css which is shared by all components.
4. Re-inventing the wheel: I have tried reusing some modules like forms so that I don't need to add other checks for validation of email and location. I have done this by using the forms module and emailField. I have also avoided writing select tags for location because ModelChoiceField does that for me. Also since the weather API already existed, I didn't have to do anything but just call the api.
5. Usability: We could have an attractive UI that draws attention and could embed a chatbot that gives more insight to the users.