DOCUMENTAZIONE PROGETTO INFORMATICA - Blettur

BREVE DESCRIZIONE OBBIETTIVI DEL PROGETTO:

In questo progetto gli obbiettivi sono quelli di gestione di dati geografici dell’utente e non, ed anche di gestione di dati su sql server per cercare di creare un database senza ridondanze e che possa prendere più dati possibili da quello che fa l’utente.

COSA SERVE IL PROGETTO:

Il progetto serve a far sì che l’utente si possa registrare e loggare dentro il sito, dove questi log e registrazione vengono tutti salvati su sql per la gestione di informazioni dell’utente. Dopo la registrazione e log dell’utente, ci troviamo nella richiesta di consentire la posizione dell’utente per far sì che il programma possa poi funzionare ed andare avanti. Dopo la pagina di richiesta di posizione troviamo la pagina della mappa che, gestendo i dati dei punti vaccinali lato server e mandandoli nel lato client, possiamo vedere sulla mappa tutti i centri vaccinali e in più, la posizione attuale dell’utente che aveva consentito in precedenza. Questi punti vaccinali sono in un raggio di 4km, un raggio che tutte le persone possono raggiungere anche per le persone più anziane, i punti vaccinali possono essere premuti e danno il nome del centro vaccinale ed anche le indicazioni su come arrivarci. La funzione del premere il centro vaccinale serve per prendere le informazioni per poi metterle in relazione con il log. Nella pagina della mappa (“index”), troviamo una “float navbar” dove troviamo le possibili route che possiamo raggiungere. Infatti, troviamo “grafici” che è una pagina dove possiamo vedere visivamente la gestione dei dati del covid-19. Infine abbiamo sempre nella navbar una funzione di logout, che permette all’utente di sloggarsi dal sito e su sql vengono salvate le informazioni di quando si è sloggato, per poi abbinarlo insieme al login, così da sapere quell’utente quando è entrato, quando è uscito e per quanto tempo è rimasto attivo nel nostro sito.

LIBRERIE UTILIZZATE:

* Flask: (Libreria python) Base del progetto è la libreria che ci permette di creare delle route dove mettere il nostro codice server e permettere reindirizzare anche file html sempre per il passaggio server client così quando si va su quella route troviamo il nostro html con dentro i dati ricavati lato server.
* Leaflet: (Libreria JavaScript), Libreria gratuita che ci permette di utilizzare la mappa, aggiungere marker, aggiungere un raggio, e con il passaggio server client tramite flask questo processo è molto facile.
* Leaflet-Routing-Machine (Libreria JavaScript), ci permette di avere le indicazioni da un punto ad un altro punto, nel caso del nostro sito dalla nostra posizione fino al punto vaccinale selezionato dall’utente.
* Chart.js (Libreria JavaScript) , permette di creare grafici in modo molto semplice che viene semplificato a sua volta dal passaggio sever client di flask, si possono creare tanti diversi tipi di grafici con molte diverse animazioni e colori a noi basta soltanto gestire i dati via server e passarla alla libreria che penserà a metterli in forma grafica
* Bootstrap (Libreria Html) permette di trasformare il nostro sito in mobile first ma anche ad aiutarci con l’aspetto del sito e facilitarci le cose grazie alle sue classi prestabilite, l’utilizzo di questa libreria dipende da persona a persona dipende come si vuole il sito.
* Jquery (Libreria JavaScript) serve per la gestione del DOM e di tutte le parti di javascript che abbiamo nel nostro sito e anche html.
* Pandas (Libreria Python) permette la gestione dei dataframe molto facilmente ed intuitivamente, con questa libreria possiamo giostrare il dataframe come vogliamo e passare quello che ci serve ad html o js (ad esempio con i grafici)
* Geopandas (Libreria Python) permette la gestione di GeoDataframe cioè i dataframe geospaziali, anche in questo caso è molto intuitiva la libreria che permette di giocare sempre con i dati e passare quello che ci serve alle librerie js per poi visualizzarlo (ad esempio con la mappa)

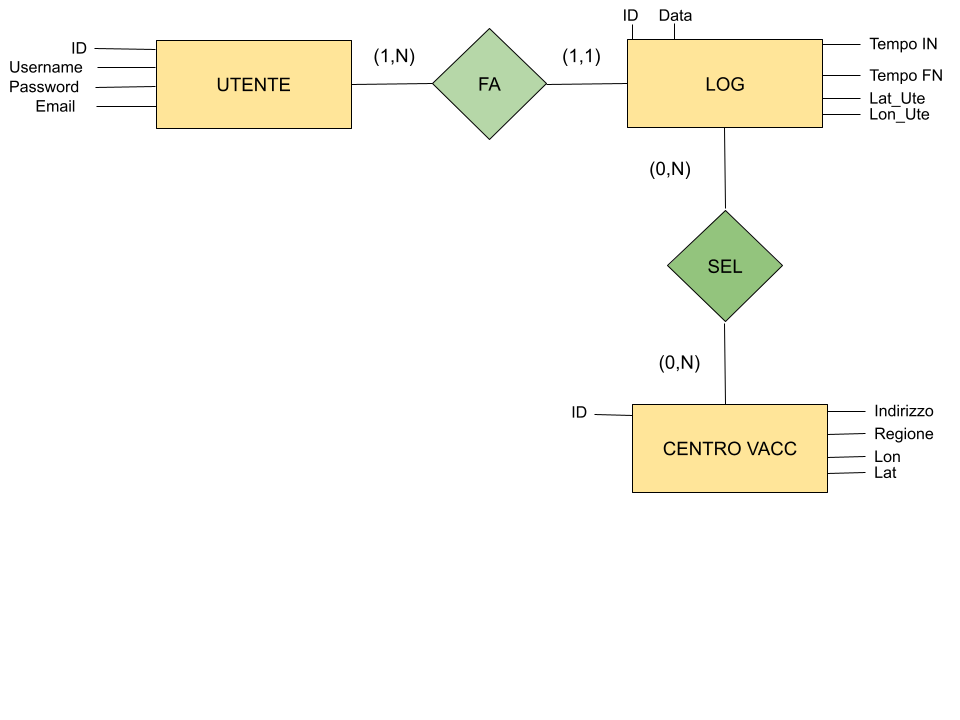
SOFTWARE UTILIZZATI NELLA REAIZZAZIONE DEL PROGETTO:

Visual Studio code

Anaconda

BREVE DESCRIZIONE DELL’ARCHITETTURA DEL PROGETTO:

Il progetto si sviluppa con front-end che sarebbe tutta la parte di javascript e html, mentre abbiamo una parte di back-end che sarebbe python flask che ci permette di gestire i dati facilmente, e infine abbiamo un lato database con sqlserver dove andiamo a mettere tutti i dati che vogliamo dall’utente.

SCHEMA DATABASE UTILIZZATO NEL PROGETTO: ER - LOGICO

UTENTE (ID,Username,Password,Email)

LOG (ID,Data,Tempo IN, Tempo FN, Lat\_Ute, Lon\_Ute,ID\_UTENTE)

CENTRO VACC (ID,Indirizzo, Regione, lat, lon)

SELEZIONA (IDLOG,IDCENTROVACC)

AZIONARE:

Ci sono 2 modi per azionare il sito la prima è fare un venv e scommentare in pezzo di codice per far partire tutto, l’altro è visitare il sito vero e proprio

<https://blettur-web.herokuapp.com/>

VENV – Io utilizzo anaconda i codici per creare un ambiente con già installato geopandas sono:

0) Ricordarsi di usare Anaconda pormt usar win button per trovarlo

1) - conda create -n geo\_env -> per creare l'ambiente virtuale (il nome è a piacere "geo\_env")  
2) conda activate geo\_env -> per attivare l'ambiente virtuale  
3) conda config --env --add channels conda-forge  
4) conda config --env --set channel\_priority strict  
5) conda install geopandas -> per installare geopandas  
6) scrivere "python"poi premere “invio” e provare a fare l'import (come se fossimo in python) per poi uscire dalla scrittura in python usare "exit()"  
7) conda install jupyter notebook -> per installare il nostro jupyter in questo ambiente  
8) python -m ipykernel install --name geo\_env -> aggiungere al nostro ambiente il notebook

Dopo aver creato questo ambiente virtuale (si può utilizzare su vs code)

Avrembo il nostro ambiente virtuale con solo geopandas e le librerie prestandard di python -> nel caso si voglia far funzionare il progetto bisognerà installare per sicurezza se non ci siano già:

pymssql,

json, -> dovrebbe già esserci

datetime, -> dovrebbe già esserci

re, -> dovrebbe già esserci

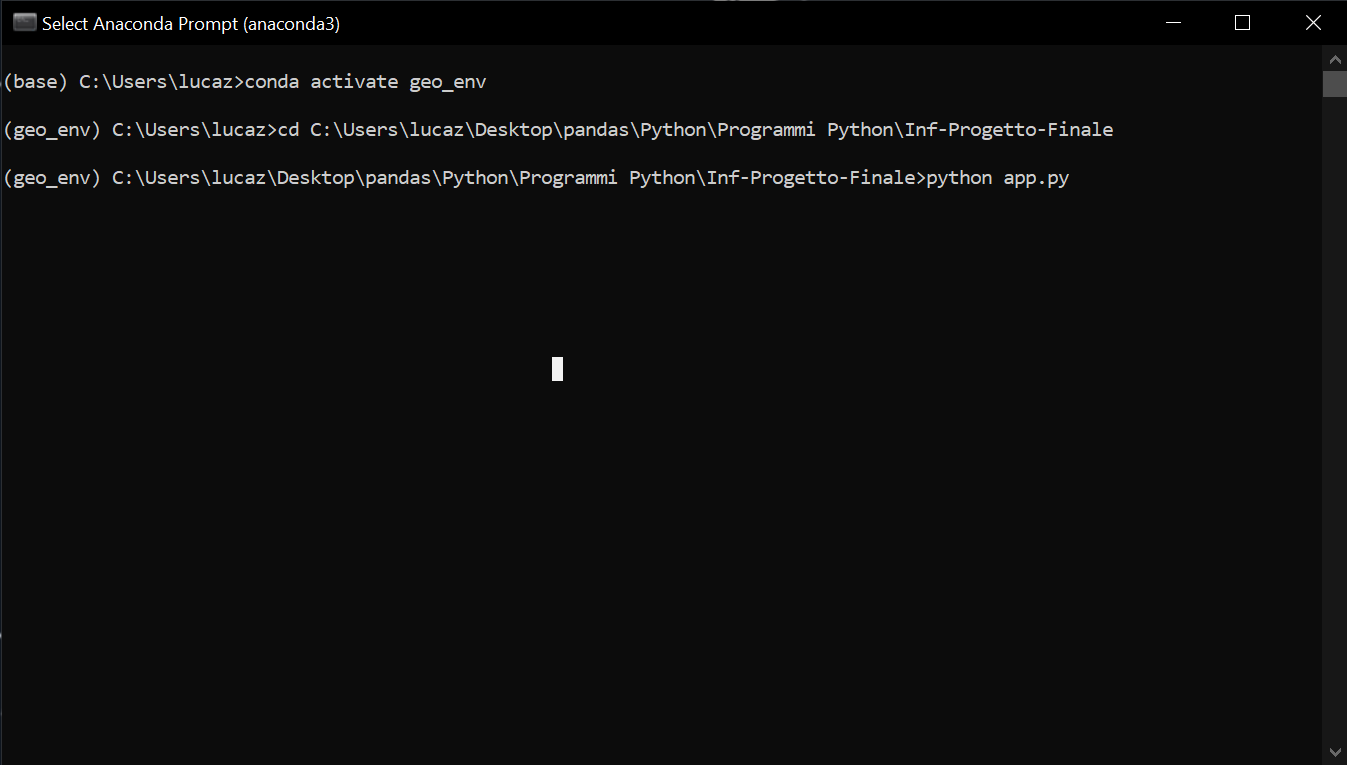
pandas, -> dovrebbe già esserci

flask

matplotlib

numpy

* Nell’ambiente creato si installano o con il comando:
  + conda install nome\_libreria
  + oppure: python -m pip install nome\_libreria

per azionare il programma vero e proprio basterà andare nell’ambiente virtuale andare nella path del programma e azionarela esempio: app.py è il nome del mio main file

Questo ambiente di sviluppo si può utilizzare anche su jupiter di anaconda.

Poi se si ha già un ambiente virtuale proprio si può utilizzare benissimo quello.

PER CREARE SITO

Innanzitutto dipende da quando si legge questa documentazione bisogna fare github for student per chi volesse che da molte opportunità e software che sarebbero a pagamento come ad esempio pycharm. Si può anche non fare (da chiedere alla preside grande consiglio)

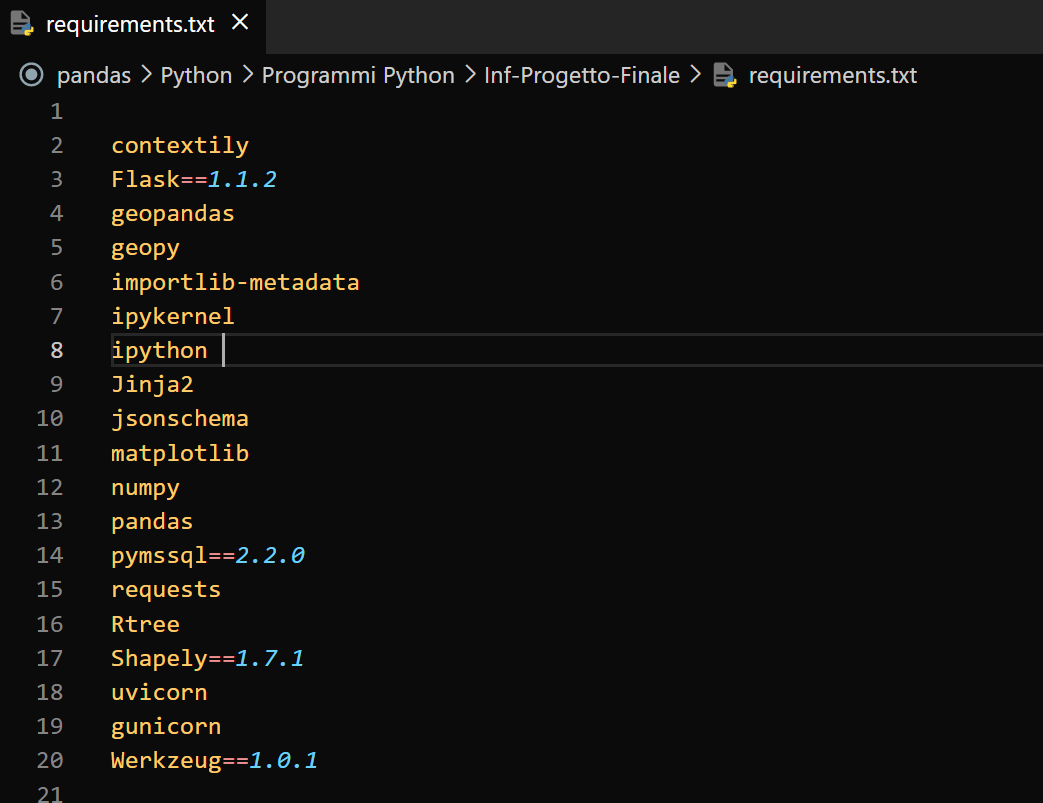
Utilizzerremo heroku, prima di tutto bisogna registrrsi e loggarsi al sito, una volta dentro ci verra detto che ossiamo creare un nuovo progetto, si può inserire il nome che si vuole o lo mette in automatico dipende dai casi. Ci troveremo davanti alla dashboard, ci dirigiamo verso la sezione deploy e consioglio di collegarsi con github creato all’inizio dell’anno istituzionalmente e dopo di che specifichiamo la nostra repository del progetto e andiamo a fare il deploy branch ma naturalmente non serve solo il nostro progetto con python e html, necessitiamo anche di altri file per far si che funzioni online il sito:

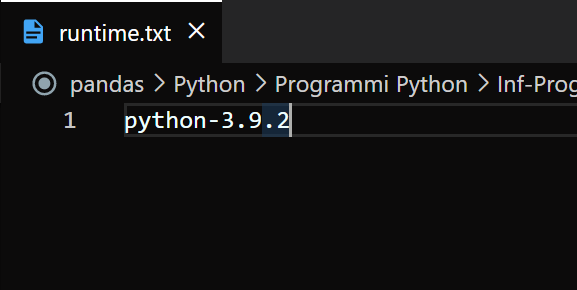
requirements.txt: -> se dovesse dare errore credo bisogni soltanto creare il file e poi eseguire il codice. Eseguire il codice all’interno del nostro ambiente virtuale dentro la path del nostro progetto

CODICE: pip freeze > requirements.txt

Un altro modo è quello di scrivere gli import del progetto dentro il file requiremetns.txt (manualmente) – questo è solo un esempio ma gunicorn e uvicorn mettterli sempre per sicurezza (forse uvicorn non è necessario) anche il relese della libreria non serve per forza.

Ma alla fine il risultato deve essere come questo senza path strane date dal freeze



Oltre al requirements.txt serve il runtime.txt che sarebbe la versione di python per eseguire il codice

L’ultimo file che serve per deployare tutto è il Procfile -> si dovrebbero trovare codici su internet di come riuscirlo ad avere ma il modo che ho usato per far funzionare il tutto è stato questo: app:app dovrebbe essere il nome del file main consiglio di chiamare il main app.py

ROTTE:

@app.route(“/”) per flask serve per andare ad una specifica route su sito web

@app.route(“/login”) l’utente viene reindirizzato a questa pagina come pagina principale e serve all’utente di loggarsi nel caso abbia già una account per entrare nel sito vero e proprio

@app.route(“/register”) serve per registrarsi al sito e vengono fatti tutti i controlli per vedere se il nome corrsiponde ecc.

@app.route(“/cookie”) pagina semplicemente bianca dove viene chiesto all’utente di consentire l’accesso alla posizione via javascript una volta completato questo reindirizza all’index

@app.route(“/index”) esce la mappa con tutti i punti vaccinali la nostra posizione la navbar e tutto quello che c’è nel codice tutto commentato

@app.route(“/graph”) serve per andare a vedere i grafici del covid 19.

@app.route(“/logout”) viene fatto il logout e vengono salvati tutti i dati su database

@app.route(“/login\_amministrator”) per far si che l’amministratore possa accedere

@app.route(“/graph\_accounts\_amministrator”) serve per far vedere la tabella con tutte le innerjoin che potrebbero servire all’amministratore.

FINALE:

Questo sito non è stato fatto apposta per la sicurezza vera e propria ho cercato con i try ed exept di risolvere alcuni problemi, ricordarsi non andare troppo veloci nel premere sul sito anche perché potrebbe andare in overload.

Spero vada bene il sito e la documentazione, quello che non ci dovesse essere nella documentazione la si può trovare nei commenti del codice.