ASTANA IT UNIVERSITY

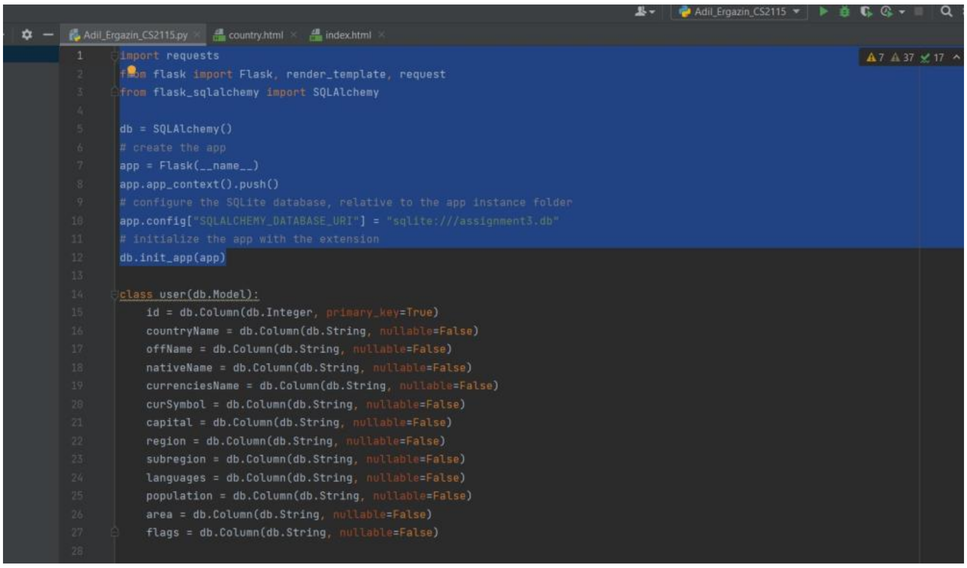
# Assingment-3: Flask, sqlalchemy, sqlite

Name: Adil Ergazin CS-2115

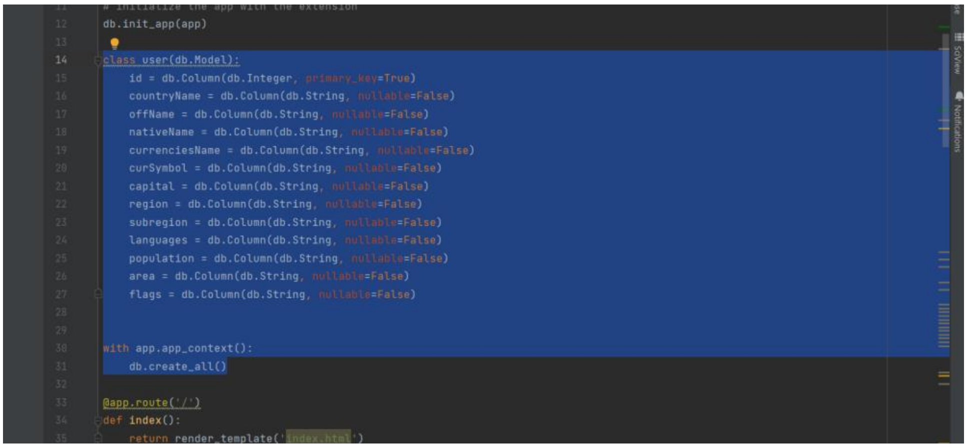
# Assignment-3

## Connect to the site’s documantation by API and Flask SqlAlchemy and connect to the database by sqlite

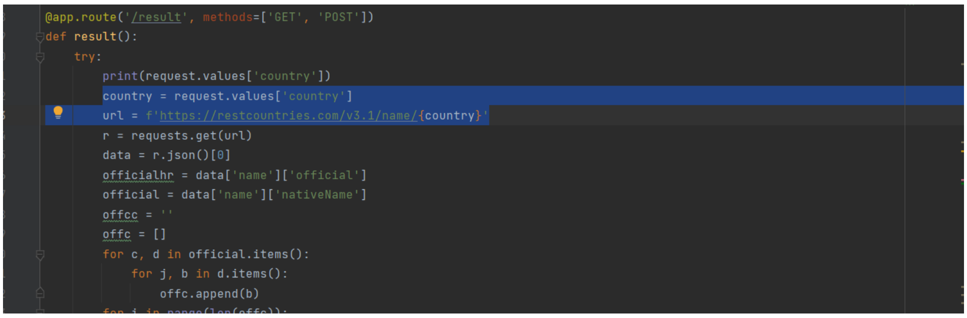
1. **Step:** Connect to the site’s information documantation by API(Rescountries, weather) and create the database connection.



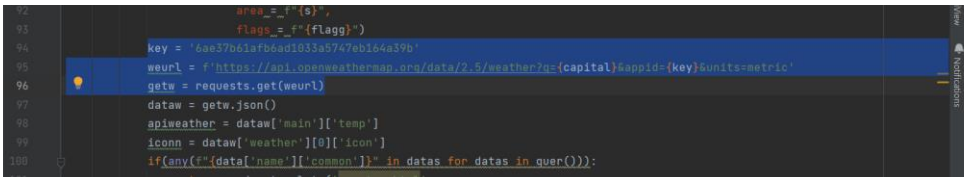
```python
import requests
from flask import Flask, render_template, request
from flask_sqlalchemy import SQLAlchemy

db = SQLAlchemy()
# create the app
app = Flask(__name__)
app.app_context().push()
# configure the SQLite database, relative to the app instance folder
app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///assignment3.db"
# initialize the app with the extension
db.init_app(app)

class user(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    countryName = db.Column(db.String, nullable=False)
    offName = db.Column(db.String, nullable=False)
    nativeName = db.Column(db.String, nullable=False)
    currenciesName = db.Column(db.String, nullable=False)
    curSymbol = db.Column(db.String, nullable=False)
    capital = db.Column(db.String, nullable=False)
    region = db.Column(db.String, nullable=False)
    subregion = db.Column(db.String, nullable=False)
    languages = db.Column(db.String, nullable=False)
    population = db.Column(db.String, nullable=False)
    area = db.Column(db.String, nullable=False)
    flags = db.Column(db.String, nullable=False)

```



```python
# initialize the app with the extension
db.init_app(app)

class user(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    countryName = db.Column(db.String, nullable=False)
    offName = db.Column(db.String, nullable=False)
    nativeName = db.Column(db.String, nullable=False)
    currenciesName = db.Column(db.String, nullable=False)
    curSymbol = db.Column(db.String, nullable=False)
    capital = db.Column(db.String, nullable=False)
    region = db.Column(db.String, nullable=False)
    subregion = db.Column(db.String, nullable=False)
    languages = db.Column(db.String, nullable=False)
    population = db.Column(db.String, nullable=False)
    area = db.Column(db.String, nullable=False)
    flags = db.Column(db.String, nullable=False)


with app.app_context():
    db.create_all()

@app.route('/')
def index():
    return render_template('index.html')
```

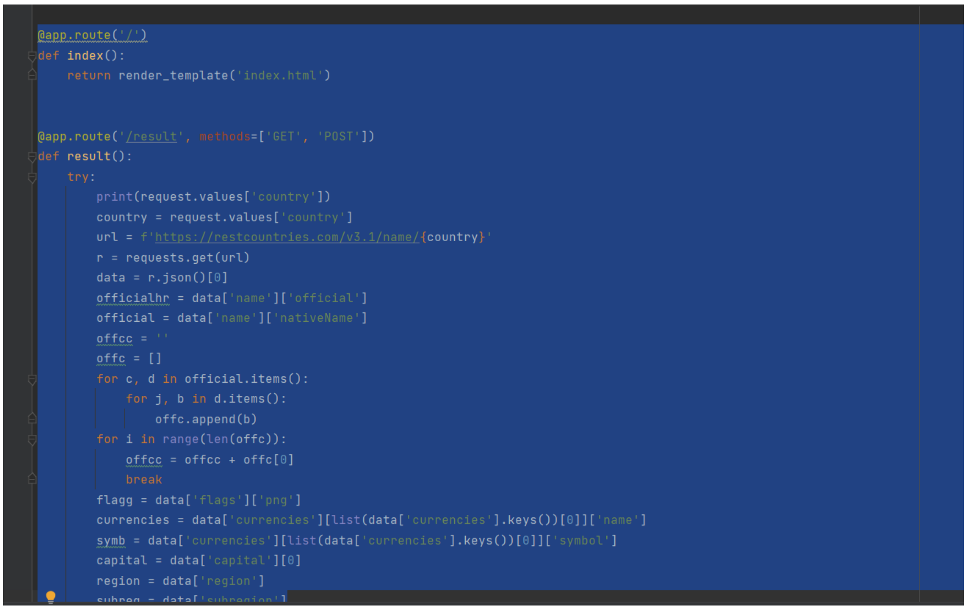


```
@app.route('/result', methods=['GET', 'POST'])
def result():
    try:
        print(request.values['country'])
        country = request.values['country']
        url = f'https://restcountries.com/v3.1/name/{country}'
        r = requests.get(url)
        data = r.json()[0]
        officialhr = data['name']['official']
        official = data['name']['nativeName']
        offcc = ''
        offc = []
        for c, d in official.items():
            for j, b in d.items():
                offc.append(b)
        for i in range(len(offc)):
```

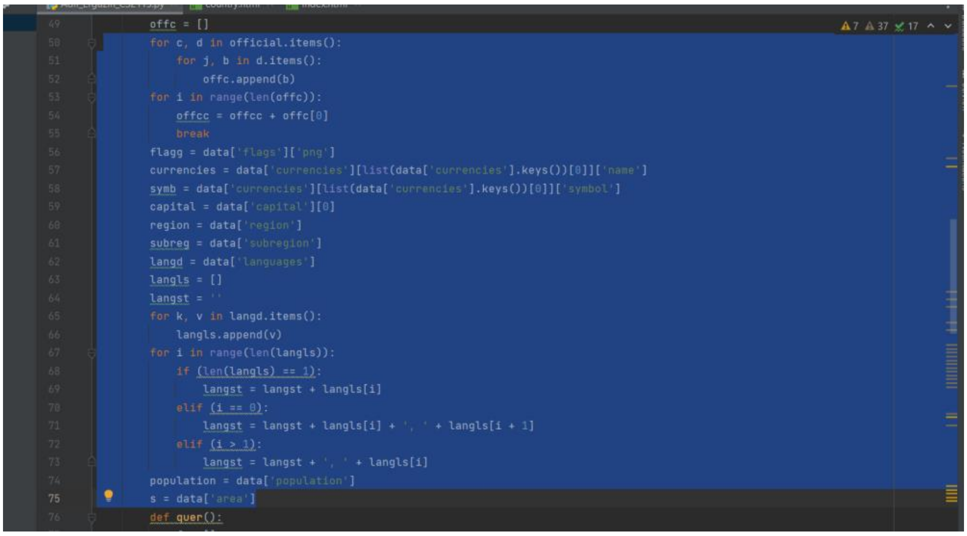


```
                    area_=_f"{s}",
                    flags_=_f"{flagg}")
        key = '6ae37b61afb6ad1033a5747eb164a39b'
        weurl = f'https://api.openweathermap.org/data/2.5/weather?q={capital}&appid={key}&units=metric'
        getw = requests.get(weurl)
        dataw = getw.json()
        apiweather = dataw['main']['temp']
        iconn = dataw['weather'][0]['icon']
        if(any(f"{data['name']['common']}" in datas for datas in quer())):
```

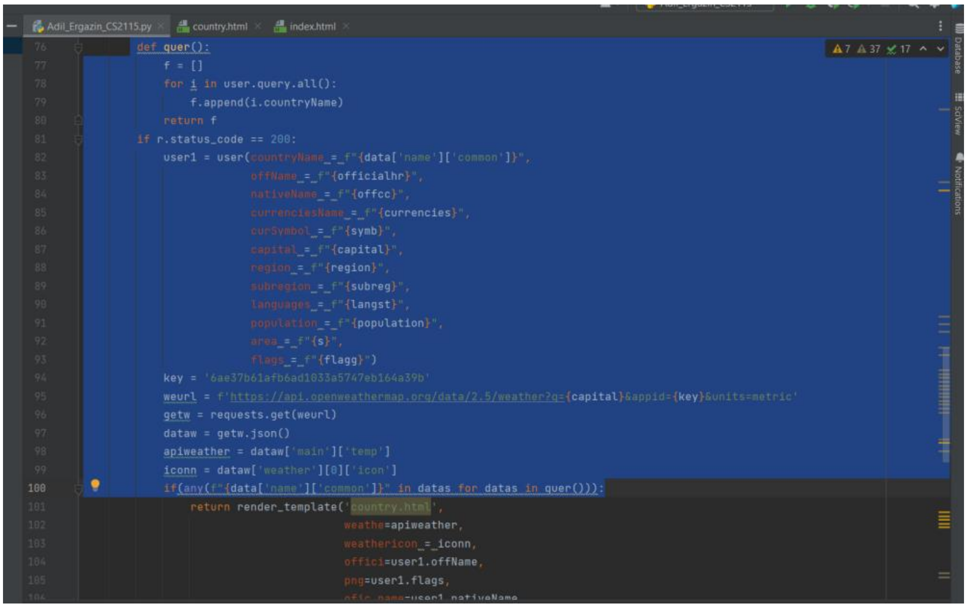
2. **Step2:** By “***route”*** mapping of the URL to the Python function and by “***GET, POST”*** methods workin with codes which are going by flask to the sql as a query of sql commands.



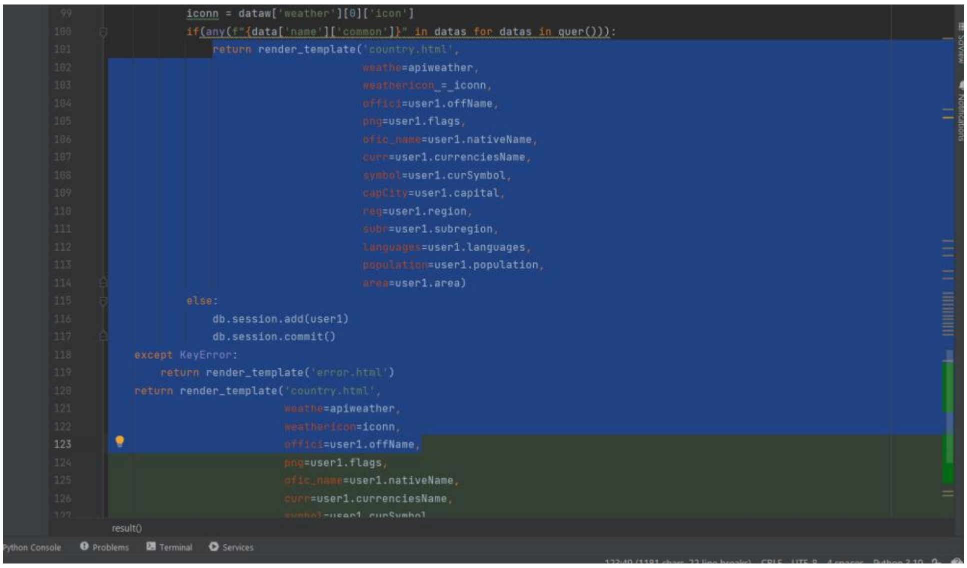
```
@app.route('/')
def index():
    return render_template('index.html')


@app.route('/result', methods=['GET', 'POST'])
def result():
    try:
        print(request.values['country'])
        country = request.values['country']
        url = f'https://restcountries.com/v3.1/name/{country}'
        r = requests.get(url)
        data = r.json()[0]
        officialhr = data['name']['official']
        official = data['name']['nativeName']
        offcc = ''
        offc = []
        for c, d in official.items():
            for j, b in d.items():
                offc.append(b)
        for i in range(len(offc)):
            offcc = offcc + offc[0]
            break
        flagg = data['flags']['png']
        currencies = data['currencies'][list(data['currencies'].keys())[0]]['name']
        symb = data['currencies'][list(data['currencies'].keys())[0]]['symbol']
        capital = data['capital'][0]
        region = data['region']
        subreg = data['subregion']
```



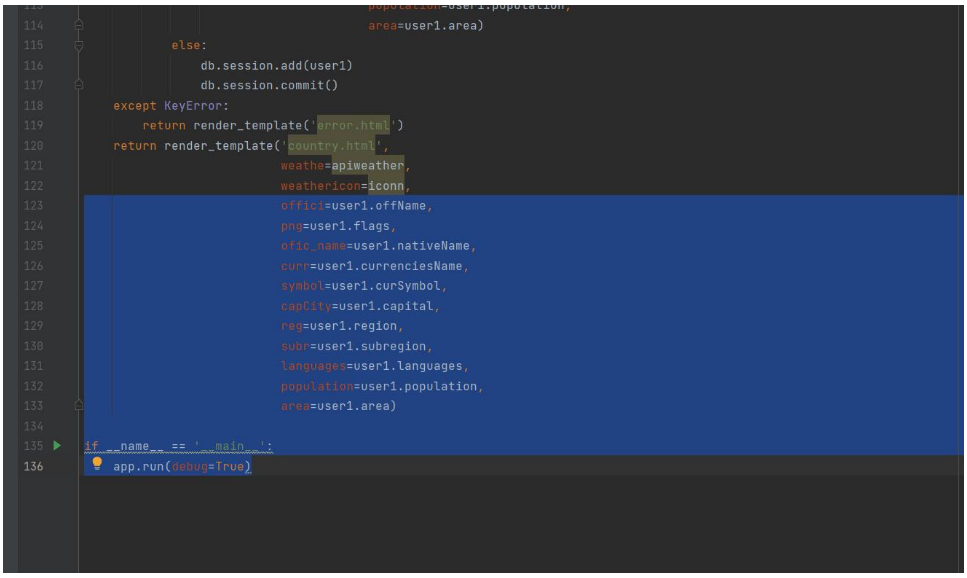
```
    offc = []
    for c, d in official.items():
        for j, b in d.items():
            offc.append(b)
    for i in range(len(offc)):
        offcc = offcc + offc[0]
        break
    flagg = data['flags']['png']
    currencies = data['currencies'][list(data['currencies'].keys())[0]]['name']
    symb = data['currencies'][list(data['currencies'].keys())[0]]['symbol']
    capital = data['capital'][0]
    region = data['region']
    subreg = data['subregion']
    langd = data['languages']
    langls = []
    langst = ''
    for k, v in langd.items():
        langls.append(v)
    for i in range(len(langls)):
        if (len(langls) == 1):
            langst = langst + langls[i]
        elif (i == 0):
            langst = langst + langls[i] + ', ' + langls[i + 1]
        elif (i > 1):
            langst = langst + ', ' + langls[i]
    population = data['population']
    s = data['area']
    def quer():
```



```
    def quer():
        f = []
        for i in user.query.all():
            f.append(i.countryName)
        return f
    if r.status_code == 200:
        user1 = user(countryName = f"{data['name']['common']}",
                     offName = f"{officialhr}",
                     nativeName = f"{offcc}",
                     currenciesName = f"{currencies}",
                     curSymbol = f"{symb}",
                     capital = f"{capital}",
                     region = f"{region}",
                     subregion = f"{subreg}",
                     languages = f"{langst}",
                     population = f"{population}",
                     area = f"{s}",
                     flags = f"{flagg}")
        key = '6ae37b61afb6ad1033a5747eb164a39b'
        weurl = f'https://api.openweathermap.org/data/2.5/weather?q={capital}&appid={key}&units=metric'
        getw = requests.get(weurl)
        dataw = getw.json()
        apiweather = dataw['main']['temp']
        iconn = dataw['weather'][0]['icon']
        if(any(f"{data['name']['common']}" in datas for datas in quer())):
            return render_template('country.html',
                                   weathe=apiweather,
                                   weathericon = iconn,
                                   offici=user1.offName,
                                   png=user1.flags,
                                   ofic_name=user1.nativeName
```

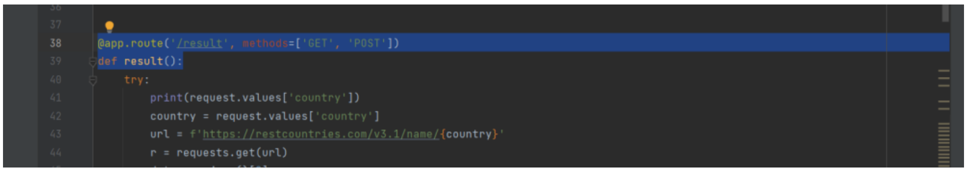


```
        iconn = dataw['weather'][0]['icon']
        if(any(f"{data['name']['common']}" in datas for datas in quer())):
            return render_template('country.html',
                                   weathe=apiweather,
                                   weathericon = iconn,
                                   offici=user1.offName,
                                   png=user1.flags,
                                   ofic_name=user1.nativeName,
                                   curr=user1.currenciesName,
                                   symbol=user1.curSymbol,
                                   capCity=user1.capital,
                                   reg=user1.region,
                                   subr=user1.subregion,
                                   languages=user1.languages,
                                   population=user1.population,
                                   area=user1.area)
        else:
            db.session.add(user1)
            db.session.commit()
except KeyError:
    return render_template('error.html')
return render_template('country.html',
                       weathe=apiweather,
                       weathericon=iconn,
                       offici=user1.offName,
                       png=user1.flags,
                       ofic_name=user1.nativeName,
                       curr=user1.currenciesName,
                       symbol=user1.curSymbol
```

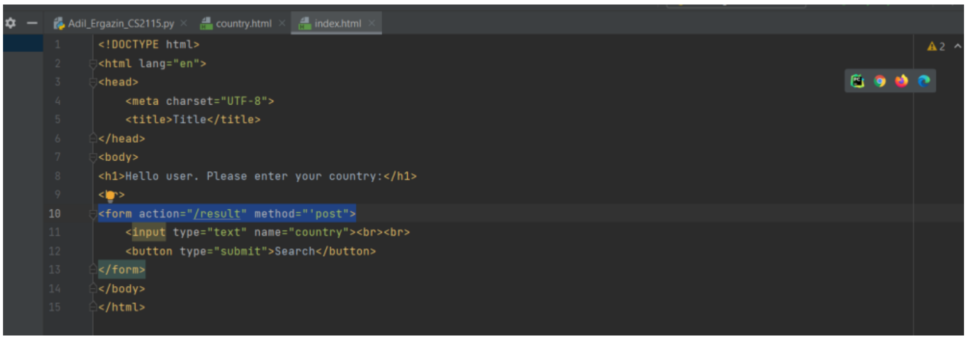


```python
                                       population=user1.population,
                                       area=user1.area)
            else:
                db.session.add(user1)
                db.session.commit()
    except KeyError:
        return render_template('error.html')
    return render_template('country.html',
                           weathe=apiweather,
                           weathericon=iconn,
                           offici=user1.offName,
                           png=user1.flags,
                           ofic_name=user1.nativeName,
                           curr=user1.currenciesName,
                           symbol=user1.curSymbol,
                           capCity=user1.capital,
                           reg=user1.region,
                           subr=user1.subregion,
                           languages=user1.languages,
                           population=user1.population,
                           area=user1.area)

if __name__ == '__main__':
    app.run(debug=True)
```

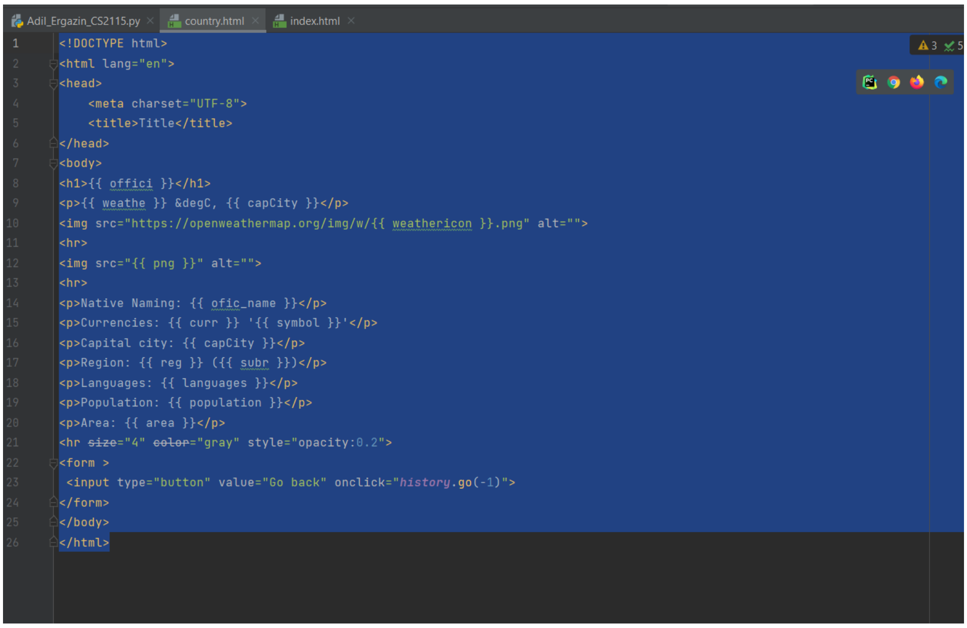
3. **Step3:** Getting information by result fucntion from “***route result”*** and by “***index.html”*** and “***country.html”*** we can see our results joining to the host ***“http://127.0.0.1:5000”*** from site’s documentation(Restcountries, weather). In order to get info fast we are using our database and it’s(from API docs) automatically inserting and by data from database it will search faster.



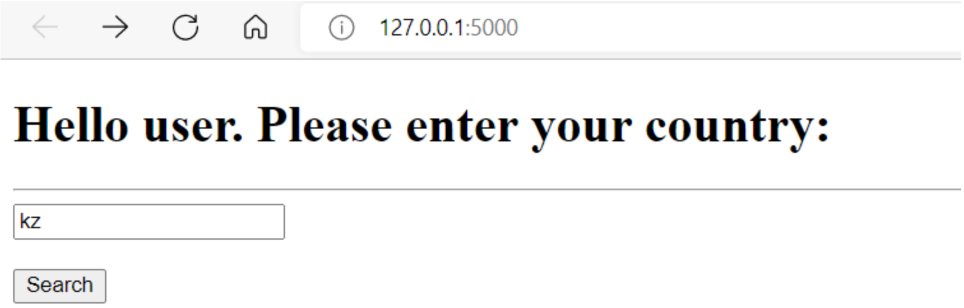
```python
@app.route('/result', methods=['GET', 'POST'])
def result():
    try:
        print(request.values['country'])
        country = request.values['country']
        url = f'https://restcountries.com/v3.1/name/{country}'
        r = requests.get(url)
```



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
<h1>Hello user. Please enter your country:</h1>
<br>
<form action="/result" method="'post">
    <input type="text" name="country"><br><br>
    <button type="submit">Search</button>
</form>
</body>
</html>
```



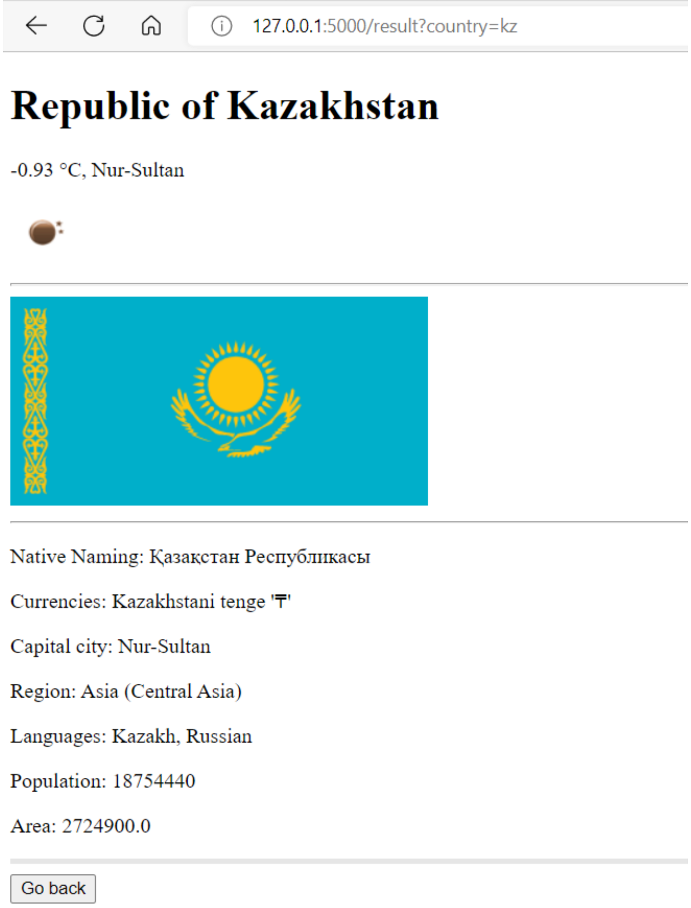
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
<h1>{{ offici }}</h1>
<p>{{ weathe }} &degC, {{ capCity }}</p>
<img src="https://openweathermap.org/img/w/{{ weathericon }}.png" alt="">
<hr>
<img src="{{ png }}" alt="">
<hr>
<p>Native Naming: {{ ofic_name }}</p>
<p>Currencies: {{ curr }} '{{ symbol }}'</p>
<p>Capital city: {{ capCity }}</p>
<p>Region: {{ reg }} ({{ subr }})</p>
<p>Languages: {{ languages }}</p>
<p>Population: {{ population }}</p>
<p>Area: {{ area }}</p>
<hr ~~size~~="4" ~~color~~="gray" style="opacity:0.2">
<form >
 <input type="button" value="Go back" onclick="history.go(-1)">
</form>
</body>
</html>
```



127.0.0.1:5000

# Hello user. Please enter your country:

kz



127.0.0.1:5000/result?country=kz

# Republic of Kazakhstan

-0.93 °C, Nur-Sultan

Native Naming: Қазақстан Республикасы

Currencies: Kazakhstani tenge '₸'

Capital city: Nur-Sultan

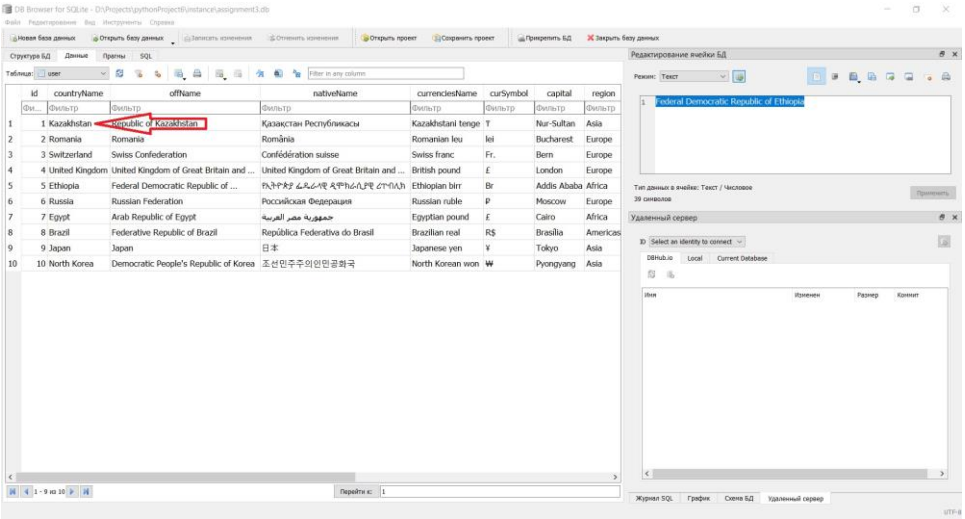
Region: Asia (Central Asia)

Languages: Kazakh, Russian

Population: 18754440

Area: 2724900.0

Go back

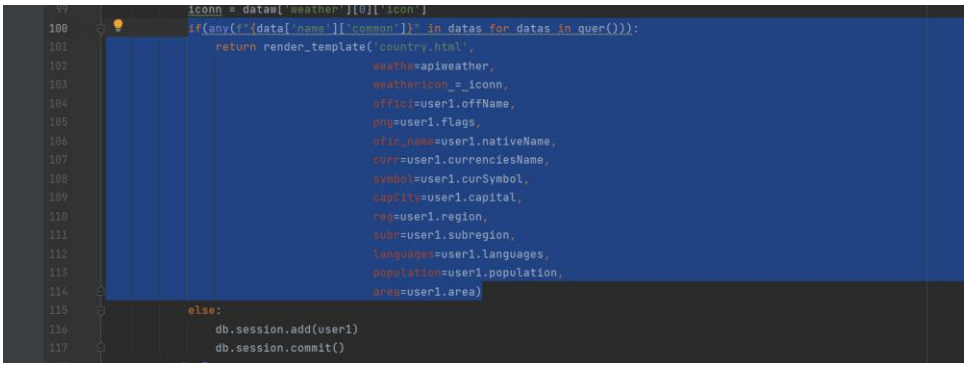


| | id | countryName | offName | nativeName | currenciesName | curSymbol | capital | region |
|---|---|---|---|---|---|---|---|---|
| | Фи... | Фильтр | Фильтр | Фильтр | Фильтр | Фильтр | Фильтр | Фильтр |
| 1 | 1 | Kazakhstan | Republic of Kazakhstan | Қазақстан Республикасы | Kazakhstani tenge | ₸ | Nur-Sultan | Asia |
| 2 | 2 | Romania | Romania | România | Romanian leu | lei | Bucharest | Europe |
| 3 | 3 | Switzerland | Swiss Confederation | Confédération suisse | Swiss franc | Fr. | Bern | Europe |
| 4 | 4 | United Kingdom | United Kingdom of Great Britain and ... | United Kingdom of Great Britain and ... | British pound | £ | London | Europe |
| 5 | 5 | Ethiopia | Federal Democratic Republic of ... | የኢትዮጵያ ፌዴራላዊ ዲሞክራሲያዊ ሪፐብሊክ | Ethiopian birr | Br | Addis Ababa | Africa |
| 6 | 6 | Russia | Russian Federation | Российская Федерация | Russian ruble | ₽ | Moscow | Europe |
| 7 | 7 | Egypt | Arab Republic of Egypt | جمهورية مصر العربية | Egyptian pound | £ | Cairo | Africa |
| 8 | 8 | Brazil | Federative Republic of Brazil | República Federativa do Brasil | Brazilian real | R$ | Brasília | Americas |
| 9 | 9 | Japan | Japan | 日本 | Japanese yen | ¥ | Tokyo | Asia |
| 10 | 10 | North Korea | Democratic People's Republic of Korea | 조선민주주의인민공화국 | North Korean won | ₩ | Pyongyang | Asia |

1 Federal Democratic Republic of Ethiopia

Тип данных в ячейке: Текст / Числовое
39 символов

Применить



```
iconn = dataw['weather'][0]['icon']
if(any(f"{data['name']['common']}" in datas for datas in quer())):
    return render_template('country.html',
                           weathe=apiweather,
                           weathericon_=_iconn,
                           offici=user1.offName,
                           png=user1.flags,
                           ofic_name=user1.nativeName,
                           curr=user1.currenciesName,
                           symbol=user1.curSymbol,
                           capCity=user1.capital,
                           reg=user1.region,
                           subr=user1.subregion,
                           languages=user1.languages,
                           population=user1.population,
                           area=user1.area)
else:
    db.session.add(user1)
    db.session.commit()
```