بسمه تعالی

پروژه نهایی مهندسی اینترنت

1402

رضا شیرالی

9912814050

ساخت یک صفحه لاگین به وسیله Google

و استفاده از Social Media برای اشتراک گذاری

مقدمه:

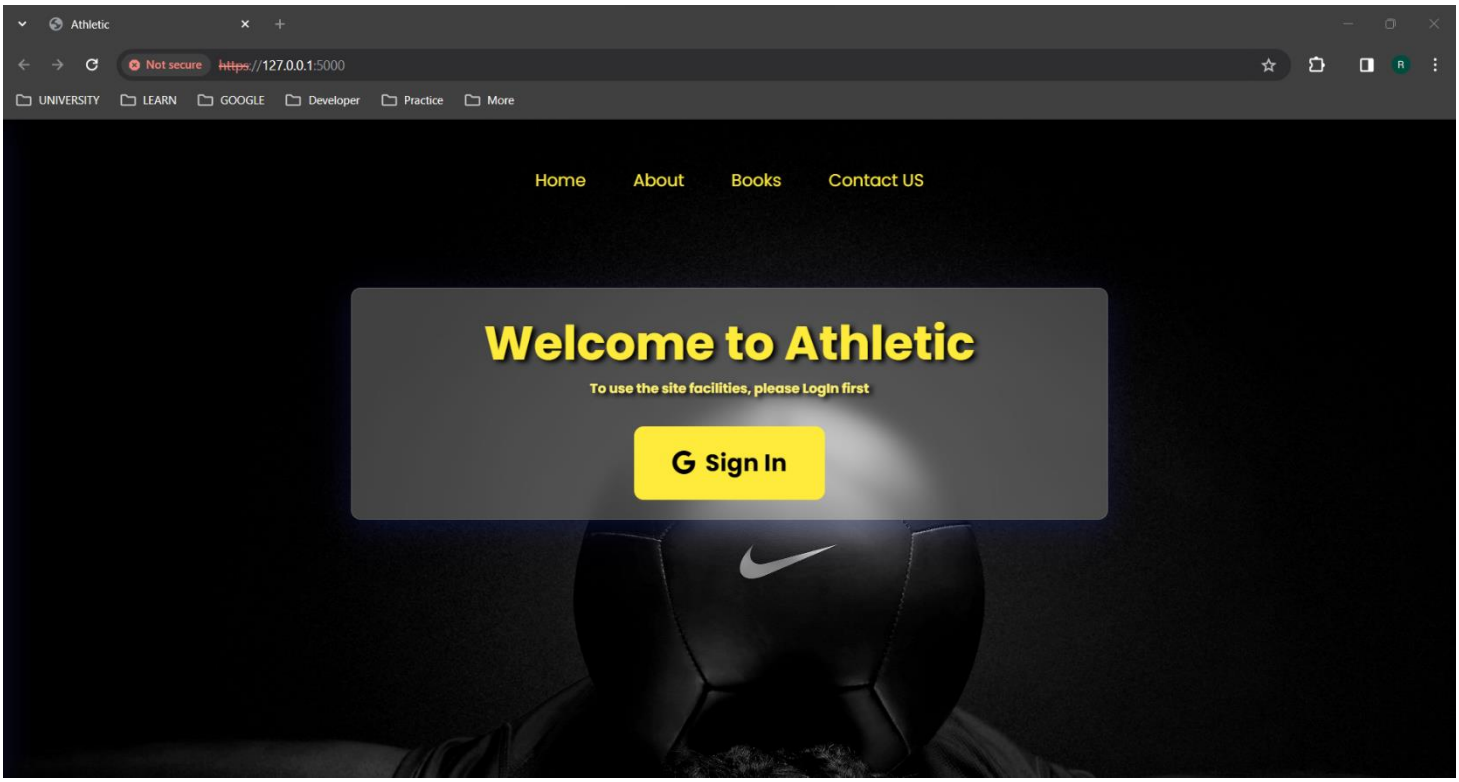
در طراحی این اپلیکیشن تحت وب ما باید از تکنولوژی هایی استفاده کنیم که هم سمت **Front** رو پوشش بدهند و هم سمت **Back**.

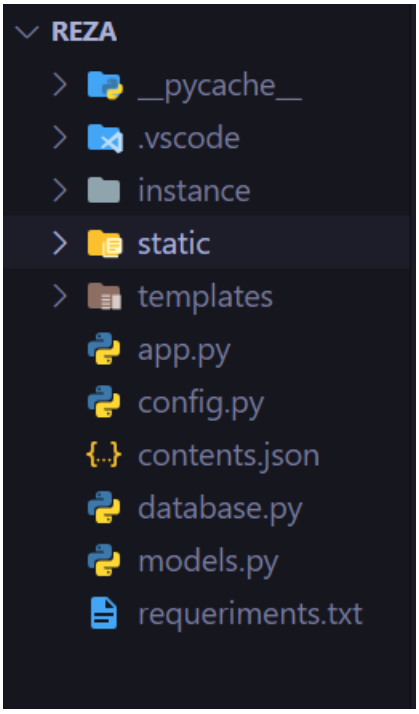
برای قسمت **Front End** از زبان نشانه گذاری **HTML** استفاده شده و برای استایل دهی آن از **CSS** بهره بردیم. بوسیله **Media Query** کار های ریسپانسیو سازی رو انجام دادیم و از فریم ورک **Bootstrap** هم برای برخی از استایل دهی ها استفاده کردیم.

برای ایجاد تعامل بهتر بین کاربر و وب سایت ما از زبان برنامه نویسی **JavaScript** استفاده کردیم؛ این عمل باعث ارتباط خیلی بهتر کاربر با وب سایت میشود.

برای قسمت **BackEnd** هم زبان برنامه نویسی **Python** و ماژول **Flask** استفاده کردیم.

در تصویر پایین نمای کلی از صفحه اصلی سایت را مشاهده میکنید؛ به مرور با بخش مختلف این پروژه آشنا خواهید شد.





بخش اول: پوشه روت

در ایـــن قســـمت پوشـــه روت پـــروژه(**REZA**) را مشـــاهده میکنید.

درون ایـــن پوشـــه تمـــام مـــواردی کـــه بـــرای پـــروژه نیـــاز اســـت قرار داده شده و در **IDE** هم همین فایل باز میشود.

*در پوشـــه **instance** فایـــل **app** قـــرار گرفتـــه کـــه مربـــوط بـــه **Data Base** اســـت و اطلاعـــات کـــاربران اینجـــا ذخیـــره میشود.

*در پوشـــه **static**، فایـــل **app.css** قـــرار گرفتـــه کـــه اســـتایل هـــای ســـایت مـــا آنجـــا قـــرار میگیرنـــد؛ یـــک پوشـــه بـــه نـــام **Script** وجـــود دارد کـــه فایـــل **app.js** آنجـــا قـــرار گرفتـــه و کـــد هـــای **javascript** ما در آن نوشته شده است.

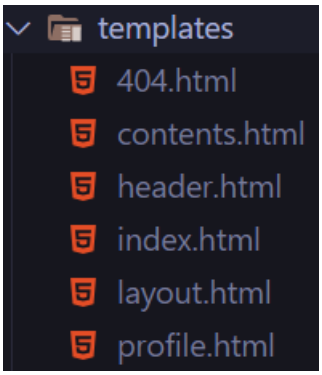
درون پوشـــه **static** یـــک پوشـــه **images** هـــم قـــرار داده شـــده کـــه بـــرای نگهـــداری عکس های پروژهه استفاده میشود.

* در پوشـــه **templates** هـــم ســـندهای **HTML** قـــرار داده شـــده کـــه فایـــل هـــای اصـــلی وبسایت ما رو تشکیل میدهند.

*در فایل **app.py** کدهای مربوط به **BackEnd** قرار داده شده است.

*در فایـــل **config.py** کـــدهای مربـــوط بـــه ارتبـــاط دیتـــابیس بـــا پـــروژه نوشـــته شـــده است.

*در مورد باقی موارد هم سرجای خود به ریز توضیح خواهم داد.

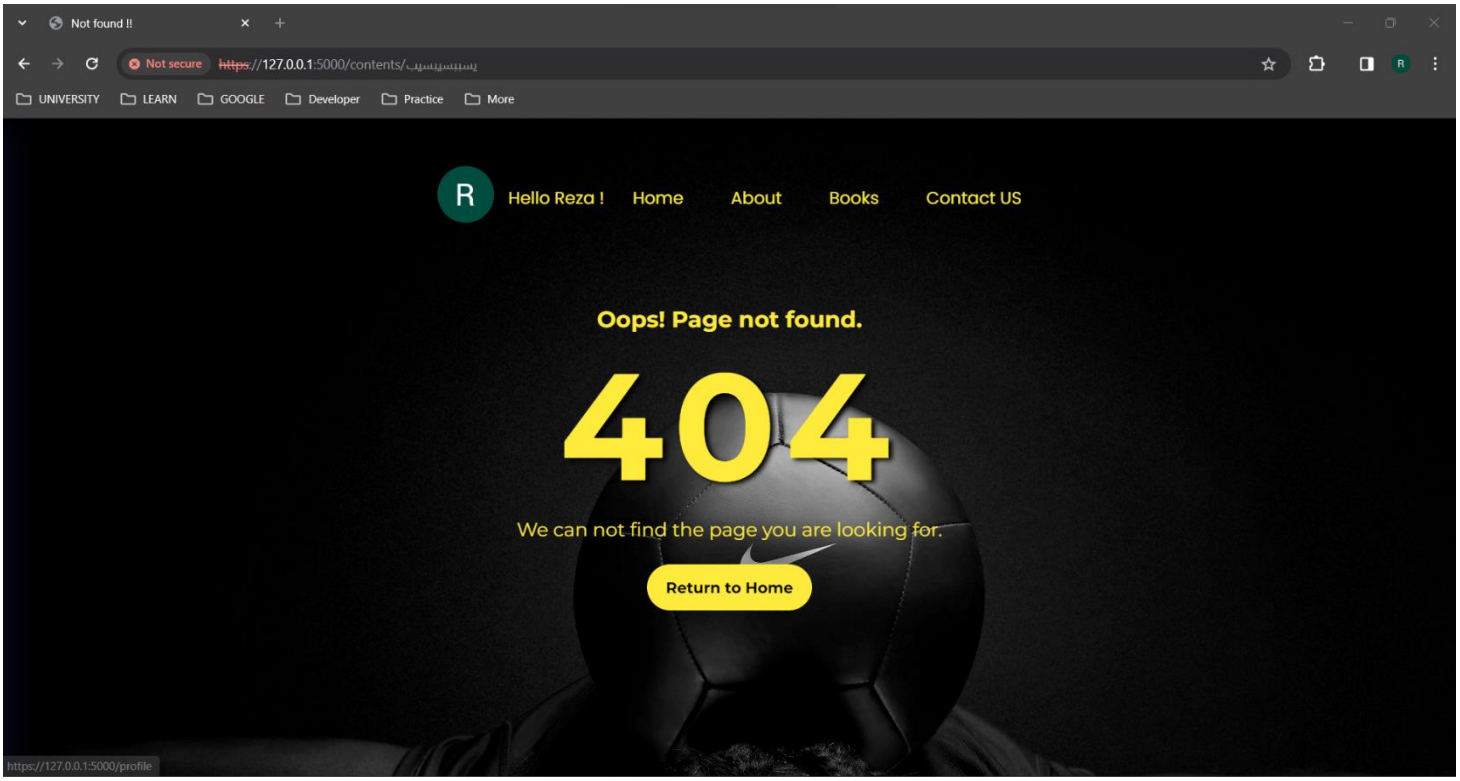


بخش دوم: پوشه templates

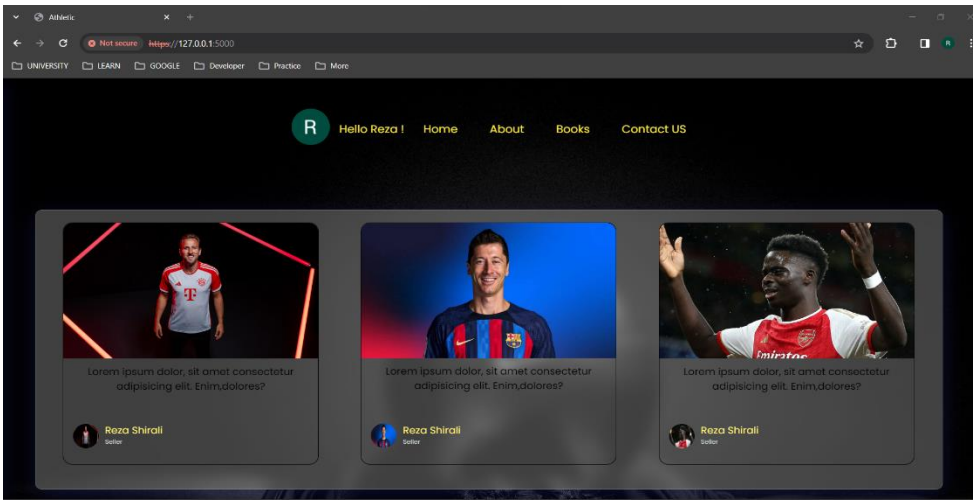
*در این پوشه سندهای **HTML** قرار داده شده است.

هرکـــدام از ایـــن ســـندها بخـــش هـــای مختلفـــی از وب ســـایت مـــا را میسازند. در آنها از **jinja2** استفاده شده است.

*در سند **404.html** کد هایی ارائه شده است تا صفحه **404** برای ما ساخته شود، تا در صورت بروز یک اشتباه از سمت کاربر این صفحه به او نمایش داده شود.



*در سند بعدی یعنی سند **contents.html**، کدهایی نوشته شده است که مربوط به صفحه مقالات است که از طریق آن میتوان به یکی از مقالات دسترسی پیدا کرد؛ در صفحه مقالات که درون همین سند تعبیه شده است شما میتوانید در مورد آن مطلب مطالعه کنید و سپس در صورتی که از آن خوشتان آمد از طریق آیکون های شبکه های اجتماعی که در زیر هر مقاله قرار گرفته اقدام به اشتراک گذاری آن مقاله بکنید.



*در سند بعدی ما به **header.html** میرسیم، در این سند ما به ساختن قسمت هدر سایت میپردازیم که منظور همان **NavBar** وب سایت است.

این سند به دو قسمت **right,left** تقسیم میشود که هر **section** مربوط به کدهای خاص خود است.این دو **section** درون یک تگ والد به نام **nav** قرار گرفته اند؛ ما به این تگ یکسری استایل های خاص دادیم که باعث ایجاد تقارن بین دو **section** میشود.

```
nav {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

در این قطعه کد ما از لایه بندی به وسیله **flex** استفاده میکنیم، پس از اعمال این خط هردو **section** در کنار هم قرار میگیرند ، سپس با اعمال کد **justify-content** هر دو **section** از لحاظ افقی در یک راستا قرار میگیرند که این راستا **center**(وسط) میباشد، در نهایت با اعمال کد **align-items** هردو **section** از لحاظ عمودی هم در یک راستا قرار میگیرند که مانند کد قبل این راستا وسط است.

در این قسمت ما منو های خودمون رو چه در حالت دکستاپ چه در حالت موبایل طراحی میکنیم که برای این امر از لیست ها و لینک ها استفاده میکنیم.

```
<ul class="right">
  <li><a href="/">Home</a></li>
  <li><a href="#">About</a></li>
  <li><a href="#">Books</a></li>
  <li><a href="#">Contact US</a></li>
</ul>/.right
```

```
<div class="nav-menu">
  <h2 class="nav-menu__title">Botanical</h2>
  <ul class="mobile-menu">
    <li class="mobile-menu__item">
      <a href="#" class="mobile-menu__link">Home</a>
    </li>/.mobile-menu__item
    <li class="mobile-menu__item">
      <a href="#" class="mobile-menu__link">About</a>
    </li>/.mobile-menu__item
    <li class="mobile-menu__item">
      <a href="#" class="mobile-menu__link">Books</a>
    </li>/.mobile-menu__item
    <li class="mobile-menu__item">
      <a href="#" class="mobile-menu__link">Contact US</a>
    </li>/.mobile-menu__item
  </ul>/.mobile-menu
  {% if user.is_authenticated %}
  <div class="nav-menu__links-logout">
    <a href="/logout" class="button__logout btn__logout">Logout</a>
  </div>/.nav-menu__links-logout
```

*در سند بعد ما به سند**index.html** میرسیم؛ در این سند هر آنچه که کاربر در بدو ورود به سایت میبیند، کد نویسی میشود.

در این سند ما فرم لاگین خودمون رو طراحی کردیمو همینطور کارت هایی که در صفحه **contents** هم قرار دارند در همین سند طراحی شده اند.

در عکس زیر خط کدهای مربوط به فرم لاگین را مشاهده میکنید.

```html
<!-- ================== LOgin Form ============================== -->
<section class="form__login">
  <div class="wrapper__card--login">
    <h1 class="card__login--title">Welcome to Athletic</h1>
    <p class="card__login--caption">
      To use the site facilities, please LogIn first
    </p>/.card__login--caption
    <div class="wrapper__google--form-login">
      <i class="bi bi-google"></i>
      <a href="/google-login" class="btn__sign-in"> Sign In </a>
    </div>/.wrapper__google--form-login
  </div>/.wrapper__card--login
</section>/.form__login
```

* **layout.html**، سند بعدی است که ما به توضیح آن میپردازیم؛ در این سند ما بیس سند های **HTML** خودمون رو داریم که از طریق موتور **jinja2** به سایر سند های **HTML** لینک شده است.

templates > layout.html > ...

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>{% block title %}{% endblock %}</title>
    <link rel="stylesheet" href="../static/app.css" />
    <link
      rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.2/font/bootstrap-icons.min.css"
    />
  </head>
  <body>
    {% include 'header.html' %}
    {% block content %}{% endblock %}
  </body>
</html>

```

*و در آخر سند **profile.html** را داریم؛ که زمانی به کاربر نشان داده میشود که کاربر از طریق اکانت **Google** خودش در سایت لاگین شده باشد.

templates > profile.html > ...

```html
{% extends 'layout.html' %} {% block title %}Botanical(Welcome){% endblock %} {%
block content %}

<div class="container">
  {% if user.is_authenticated %}
  <h1 class="user__name">Welcome, {{ user.name }}!</h1>
  <p class="email__login">Email: {{ user.email }}</p>
  <div class="profile-picture">
    <img src="{{ user.profile_pic }}" alt="Google Profile Picture" />
  </div>/.profile-picture
  <a href="/" class="link__home">Return to Home</a>
  {% else %}
  <p>You are not logged in</p>
  <a href="/google-login" class="btn__sign-in"> Sign In with Google </a>
  {% endif %}
</div>/.container

{% endblock %}
```

بخش سوم: پوشه static

*در این پوشه ما اول از همه یک فایل **css** داریم به نام **app.css** که کل استایل دهی های ما در این سند قرار گرفته است؛ درنهایت این سند حدود **800** خط کد را به خود اختصاص داده است.

توضیح خط به خط این سند امری غیر ممکن است؛ به همین منظور بخش های مهم این سند را توضیح میدهیم و باقی خط ها را میتوانید در خود سند مشاهده کنید.

در دو خط اول فونت هایی که در این سایت استفاده شده را ایمپورت کردیم.

```
@import url("https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;700&display=swap");
@import url("https://fonts.googleapis.com/css2?family=Montserrat:wght@400;500;700&display=swap");
```

در خط بعدی با استفاده از **:root** ما یکسری تابع ایجاد کردیم و در آنها رنگ های اصلی پروژه قرار داده شده و هرجای دیگه که به این رنگ ها نیاز داشتیم فقط لازمه که با استفاده کد**var()** و اسم اون متغیر رنگ رو اعمال کنیم و از نوشتن خط کد اضافی دوری میکنیم.

```
:root {
  --pr-color: #ffeb3b;
  --sec-color: #fff176;
}
```

سپس یک سری استایل های بیس باید به سند **html** خودمون میدهیم که شامل خط کد های زیر است:

```
html {
  box-sizing: border-box;
  font-size: 62.5%;
  font-family: "Poppins", sans-serif;
}
```

و در نهایت در قسمت **body** هم فونت سایز تعریف شده و هم بک گراند وب سایت روی کل **body** ست میشود.

```
body {
  font-size: 1.6rem;
  margin: 5rem;
  background-image: url("/static/images/sport1.jpg");
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;
  background-attachment: fixed;
}
```

بخش چهارم: Back End

در قسمت **backend** اولین فایلی که با آن سر و کار داریم **app.py** است که تمام کدهای قسمت **backend** در آن قرار دارد که به صورت خط به خط آنها رو توضیح میدهیم:

*در اولین بخش، کتابخانه‌ها و ماژول‌های مورد نیاز برای اجرای برنامه و اجتماعی سازی آن ایمپورت شده‌اند.

```python
# Python standard libraries
import json
from flask import Flask, redirect, request, url_for,render_template
from flask_login import (
    LoginManager,
    current_user,
    login_required,
    login_user,
    logout_user
)
from oauthlib.oauth2 import WebApplicationClient
import requests
import urllib.parse

from config import *
from database import db
from models import *
```

*در بخش بعدی تنظیمات اولیه اپلیکیشن انجام میشود.

```python
# Flask app setup
app = Flask(__name__)
app.config['SECRET_KEY'] = SECRET_KEY
app.config['SQLALCHEMY_DATABASE_URI'] = SQLALCHEMY_DATABASE_URI
```

*خط بعدی مدیریت ورود کاربران به وبسایت را انجام میده.

```python
login_manager = LoginManager()
login_manager.init_app(app)
```

*این خط کد نشان دهنده تنظیم یک تابع به عنوان دستگیره (**handler**) برای مواردی است که کاربر به عنوان "غیرمجاز" (**unauthorized**) شناخته می‌شود. در واقع، این تابع به عنوان یک دستگیره برای وقایعی که کاربر دسترسی مجاز ندارد، فراخوانی می‌شود.

```
@login_manager.unauthorized_handler
def unauthorized():
    return "You must be logged in to access this content.", 403
```

*این کد در حال ایجاد یک نمونه از کلاس `**WebApplicationClient**` از یک کتابخانه مرتبط با **OAuth** است. این کد به نظر می‌رسد که برای اجرای پروتکل **OAuth 2.0** در یک برنامه وب استفاده می‌شود.

به طور کلی، این خط کد نشان‌دهنده آماده‌سازی برنامه برای استفاده از **OAuth 2.0** با استفاده از کتابخانه‌ها یا ابزارهای مرتبط است، به ویژه در ارتباط با **Google OAuth**.

```
# OAuth2 client setup
client = WebApplicationClient(GOOGLE_CLIENT_ID)
```

*این قطعه کد به زبان پایتون با استفاده از چارچوب وب **Flask** نوشته شده است. این کد یک مسیر (**route**) برای اپلیکیشن تعریف کرده که وقتی کاربر به این مسیر دسترسی پیدا کند (به طور پیش‌فرض مسیر اصلی یا ریشه `/` این اپلیکیشن)، یک تابع به نام `**index**` اجرا می‌شود.

```
@app.route("/")
def index():
    if current_user.is_authenticated :
        contents = json.load(open("contents.json", "r", encoding="utf-8"))
    else :
        contents = {}
    return render_template('index.html' , user = current_user ,contents = contents)
```

+تعداد زیادی از باقی کدها دقیقا مانند کد بالا عمل میکنند. پس از توضیح آنها صرف نظر میکنیم.

*این قطعه کد به عنوان یک دستگیره در **Flask** برای مسیر "**/google-login/callback**" تعریف شده است و وظیفه آن اجرای فرآیند احراز هویت کاربر از طریق ورود به سیستم **Google OAuth 2.0** است.

```python
@app.route("/google-login/callback")
def callback():

    code = request.args.get("code")

    google_provider_cfg = get_google_provider_cfg()
    token_endpoint = google_provider_cfg["token_endpoint"]

    token_url, headers, body = client.prepare_token_request(
        token_endpoint,
        authorization_response=request.url,
        redirect_url=request.base_url,
        code=code,
    )
    token_response = requests.post(
        token_url,
        headers=headers,
        data=body,
        auth=(GOOGLE_CLIENT_ID, GOOGLE_CLIENT_SECRET),
    )
    client.parse_request_body_response(json.dumps(token_response.json()))
    userinfo_endpoint = google_provider_cfg["userinfo_endpoint"]
    uri, headers, body = client.add_token(userinfo_endpoint)
    userinfo_response = requests.get(uri, headers=headers, data=body)

    if userinfo_response.json().get("email_verified"):
        unique_id = userinfo_response.json()["sub"]
        users_email = userinfo_response.json()["email"]
        picture = userinfo_response.json()["picture"]
        users_name = userinfo_response.json()["given_name"]
    else:
        return "User email not available or not verified by Google.", 400

    user = User.get(unique_id)
    if not user:
        user = User(id=unique_id, name=users_name, email=users_email, profile_pic=picture)
        db.session.add(user)
    db.session.commit()

    login_user(user)

    return redirect('/profile')
```

*در این کد، تابع `**get_google_provider_cfg**` یک درخواست **GET** به یک **URL** مشخص ارسال می‌کند تا اطلاعات تنظیمات (**configuration**) ارتباط با سرویس **Google OAuth 2.0** را دریافت کند.

- **json**: با استفاده از این متد، محتوای درخواست در قالب **JSON** که به عنوان پاسخ دریافت شده است، باز می‌گردد.

```python
def get_google_provider_cfg():
    return requests.get(GOOGLE_DISCOVERY_URL).json()
```

*و در نهایت قسمت آخر کدهای ما به صورت زیر هستند و توضیحات خط به خط آن را براتون مینویسم:

1. `**db.init_app(app)**`: این دستور از تابع `**init_app**` کتابخانه **SQLAlchemy** استفاده می‌کند تا ارتباط پایگاه داده را با اپلیکیشن **Flask** برقرار کند. به این ترتیب، تنظیمات پایگاه داده مرتبط با اپلیکیشن می‌شوند.

2. `**with app.app_context()**`: این دستور یک محیط (**context**) فراهم می‌کند که متعلق به اپلیکیشن است. این محیط مورد استفاده قرار می‌گیرد تا عملیات‌هایی مانند ایجاد جداول در پایگاه داده انجام شوند.

3. `**db.create_all()**`: این دستور از تابع `**create_all**` کتابخانه **SQLAlchemy** استفاده می‌کند تا تمامی جداول مورد نیاز برای پایگاه داده را بسازد. این عملیات معمولاً در مرحله اولیه اجرای برنامه و پس از تغییرات مهم در ساختار دیتابیس انجام می‌شود.

4. `**app.run(debug=True, ssl_context="adhoc")**`: این دستور برنامه را با استفاده از تابع `**run**` اجرا می‌کند. پارامتر `**debug=True**` نشان‌دهنده اجرای برنامه در حالت اشکال‌زدایی (**debug mode**) است. پارامتر `**ssl_context="adhoc**`" نیز برنامه را با استفاده از یک گواهینامه **SSL** خودآموز (**self-signed**) اجرا می‌کند، که به صورت موقت و برای محیط توسعه استفاده می‌شود. این گزینه معمولاً برای اجرای برنامه بر روی **HTTPS** در حالت توسعه استفاده می‌شود.

```
if __name__ == "__main__":
    db.init_app(app)
    with app.app_context():
        db.create_all()
    app.run(debug=True,ssl_context="adhoc")
```

امیدوارم رضایت کامل از این وب اپلیکیشن داشته باشید.

رضا شیرالی-**9912814050**