به نام خدا

مهندسی اینترنت

استاد : مهندس امیرکیوان شفیعی

نویسنده:

عالیه یمنی

موضوع تمرین :

وب سایت با اعتبار سنجی توسط گوگل و تعامل با شبکه های اجتماعی و استفاده از API

مقدمه :

در عصر دیجیتال امروزی، هر وب سایتی باید علاوه بر دسترسی به اطلاعات مفید و گسترده، تجربه‌ی کاربری ساده و بی‌دردسری را نیز برای کاربران خود به ارمغان بیاورد. تنظیم مناسب و اجرای بدون نقص سیستم‌های احراز هویت کاربران بخشی ضروری از هر سایت معتبر محسوب می‌شود که وفاداری کاربران و اعتماد را بالا می‌برد.

استفاده از سیستم‌های احراز هویت مطرحی مانند ورود از طریق حساب کاربری گوگل می‌تواند به عنوان رویکردی مدرن، ایمن، و کم‌دردسر در دسترسی به وب‌سایت‌ها عمل کند.

در این تمرین به تشریح درونمایه فریم‌ورک قدرتمند **Flask** در زبان برنامه‌نویسی پایتون می‌پردازیم که مکانیسم‌های ساده و منعطفی برای ساخت وب‌سایت‌ها و وب‌اپلیکیشن‌های پویا و رسپانسیو ارائه می‌دهد. این فریم‌ورک، با پشتیبانی از چارچوب معماری **MVC** و امکان استفاده از موتور قالب **Jinja۲**، تجربه برنامه‌نویسی بهینه‌ای را فراهم می‌سازد.

علاوه بر تمرکز بر احراز هویت، در این تمرین به بررسی گام‌های اجرایی برای ادغام**API** های شبکه‌های اجتماعی می‌پردازیم. این کار امکانات وسیعی را جهت به اشتراک‌گذاری سریع و آسان محتوا به کاربران در پلتفرم‌های پرکاربردی همچون تلگرام، واتس اپ و اینستاگرام می‌دهد. همزمان، آن را به یک روش موثر برای افزایش دامنه‌ی دید و نفوذ محتوای تولیدی برای توسعه‌دهندگان تبدیل می‌نماید.

در این مستند به زبانی روان و گام به گام، پیاده‌سازی این ویژگی‌ها در قالب یک وب‌سایت تعامل‌گرا با پیاده‌سازی اعتبار سنجی از طریق گوگل و تعامل با شبکه‌های اجتماعی از طریق استفاده از**API** های آن‌ها را شرح می‌دهیم.

برنامه نویسی (توضیحات) :

این برنامه از چند بخش ایجاد شده است که بخش **main.py** مسئول اجرای برنامه می باشد و با دستور **python main.py** در محیط ترمینال اجرا می شود .

```
from __init__ import create_app

app = create_app()

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

بخش __init__.py Flask با اتصال به یک پایگاه داده و مدیریت جلسه کاربران برای احراز هویت است.

```
db = SQLAlchemy()
DB_NAME = "data.db"

def login_is_required(function):
    def wrapper(*args, **kwargs):
        if "google_id" not in session:
            return abort(401)
        else:
            return function()

    return wrapper

def create_app():
    app = Flask(__name__)
    app.config['SECRET_KEY'] = 'fr8551rfesfeve sedfsef'
    app.config['SQLALCHEMY_DATABASE_URI'] = f'sqlite:///{DB_NAME}'
    os.environ["OAUTHLIB_INSECURE_TRANSPORT"] = "1"
    db.init_app(app)

    from Routing import R

    app.register_blueprint(R, url_prefix='/')

    from models import User
```

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

    login_manager = LoginManager()
    login_manager.login_view = 'R.login_page'
    login_manager.init_app(app)

    @login_manager.user_loader
    def load_user(id):
        return User.query.get(int(id))

    return app

def create_database(app):
    if not path.exists('website/' + DB_NAME):
        db.create_all(app=app)
        print('Created Database!')
```

واردات:

کتابخانه‌های مورد نیاز برای مدیریت جلسه‌های کاربری (LoginManager)و ORM پایگاه داده (SQLAlchemy) وارد شده‌اند.

تنظیمات پایگاه داده:

SQLAlchemy به عنوان ORM انتخاب شده تا عملیات پایگاه داده SQL را به صورت شی‌گرایی مدیریت کند.

DB_NAME نام پایگاه داده SQLite است که به عنوان پایگاه داده برنامه استفاده خواهد شد.

تابع:login_is_required

این یک دکوراتور سفارشی است که برای ایجاد احراز هویت اجباری در مسیرهای خاص در Flask استفاده می‌شود. این تابع قبل از اجرای تابع اصلی بررسی می‌کند که آیا کاربر وارد شده است یا خیر.

تابع create_app :

وظیفه ایجاد و پیکربندی نمونه برنامهFlask

کلید محرمانه‌ی وب‌اپ (SECRET_KEY) تنظیم می‌شود تا امنیت جلسات کاربری و فرم‌ها تأمین شود.

SQLALCHEMY_DATABASE_URI مکان پایگاه داده برای اتصال SQLAlchemy تعریف می‌شود.

متغیر محیطی برای فعال‌سازی انتقال امن OAuth در محیط توسعه تنظیم می‌شود به وسیله app.app_context()پایگاه داده‌ها در زمان شروع برنامه ایجاد می‌شود.

مدیریت ورود کاربران:

LoginManager مورد نظر می‌گیرد تا جلسات کاربری را مدیریت کند.

نمای ورود تنظیم می‌شود ,(login_view) که مشخص می‌کند کاربران برای ورود به کجا هدایت شوند.

load_user یک تابع callback است که با استفاده از user_loader, هویت کاربر جاری را بر اساس شناسه‌ی کاربر بارگذاری می‌کند.

تابع create_database

این تابع بررسی می‌کند اگر پایگاه داده‌ای وجود ندارد، آن را ایجاد کرده و یک پیام در خروجی چاپ می‌کند.

Routing.py مربوط به بخش‌های اصلی سیستم ورود و خروج، و احراز هویت کاربران با استفاده از حساب کاربری گوگل در چارچوب Flask است. همچنین روشی برای به اشتراک‌گذاری یک پیام از طریق تلگرام تدارک دیده شده است.

```
app = Flask(__name__)


R = Blueprint('R', __name__)
os.environ["OAUTHLIB_INSECURE_TRANSPORT"] = "1"
app.secret_key = 'tdhthtdy ttydtydty5dr'
migrate = Migrate(app, db)


client_secrets_file = os.path.join(pathlib.Path(__file__).parent,
"client_secret_736096332106-
huf5s52ndbfdl784fqano40ur2p46mp6.apps.googleusercontent.com.json")


GOOGLE_CLIENT_ID = "736096332106-
huf5s52ndbfdl784fqano40ur2p46mp6.apps.googleusercontent.com"
GOOGLE_CLIENT_SECRET = "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"



flow = Flow.from_client_secrets_file(
   client_secrets_file=client_secrets_file,
   scopes=["https://www.googleapis.com/auth/userinfo.profile",
"https://www.googleapis.com/auth/userinfo.email", "openid"],
   redirect_uri="http://127.0.0.1:5000/callback",
)


@R.route('/', methods=['GET', 'POST'])
@login_required
```

```
def home():

  return render_template('home.html',user=current_user)

@R.route('/login')
def login_page():

  return render_template('login.html')

@R.route('/loginbygoogle')
def login_bygoogle():

    authorization_url, state = flow.authorization_url()
    session["state"] = state
    return redirect(authorization_url)


@R.route('/callback')
def callback():

    flow.fetch_token(authorization_response=request.url)

    if not session["state"] == request.args["state"]:
        abort(500)
```

```
    credentials = flow.credentials
    request_session = requests.session()
    cached_session = cachecontrol.CacheControl(request_session)
    token_request = google.auth.transport.requests.Request(session=cached_session)


    id_info = id_token.verify_oauth2_token(
        id_token=credentials._id_token,
        request=token_request,
        audience=GOOGLE_CLIENT_ID

    )
    session["google_id"] = id_info.get("sub")
    session["name"] = id_info.get("name")
    session["email"] = id_info.get("email")
    user = User.query.filter_by(email=session["email"]).first()
    if not user:
        plain_password = ''.join(random.choice(string.ascii_letters) for i in range(10))
        hashed_password = generate_password_hash(plain_password, method='pbkdf2:sha256')


        new_user = User(
            email=session["email"],
```

```
            first_name=session["name"],
            password=hashed_password
        )
        db.session.add(new_user)
        db.session.commit()
        user = new_user
    login_user(user, remember=True)
    return redirect(url_for('R.home'))


@R.route('/share_on_telegram')
@login_required
def sharetotelegram():
    message = f"UserName: {current_user.first_name}\nEMAIL:
{current_user.email}\n visit our Flask APP by Google auth my user id is
:{current_user.id} "
    encoded_message = quote_plus(message)


    url_to_share = url_for('R.home', _external=True)


    telegram_url =
f"https://t.me/share/url?url={quote_plus(url_to_share)}&text={encoded_messag
e}"


    return redirect(telegram_url)


@R.route('/logout')
```

```
@login_required
def logout():
    logout_user()

    return redirect(url_for('R.login_page'))
```

۱. تنظیمات اولیه:

- کتابخانه‌های لازم وارد شده و سپس نمونه‌ای از کلاس Flask ایجاد می‌شود.
- یک راه‌اندازی (Blueprint) با نام ’R‘ ایجاد می‌شود که به تقسیم‌بندی مسیرهای درخواست و بخش‌بندی کد کمک می‌کند.
- متغیرهای محیطی مربوط به OAuth و تنظیم کلید محرمانگی برای جلسات کاربری تعریف می‌شوند.
- شی Flowاز Google OAuth ایجاد می‌شود که تنظیمات مربوط به فرآیند احراز هویت گوگل را مدیریت می‌کند.

۲.مسیرهای ورود و خروج:

- مسیر اصلی (/) تعریف شده که صفحه اصلی قرار است در آن رندر شود و ورود به آن نیازمند احراز هویت است.
- مسیر ورود (/login) صفحه‌ای است که در آن فرم ورود کاربران قرار داده شده است.
- مسیر (/loginbygoogle) راه‌اندازی فرآیند احراز هویت از طریق Google OAuth را انجام می‌دهد.
- مسیر (/callback) به عنوان URL بازگشتی (redirect URI) عمل می‌کند و توکن‌های مربوط به احراز هویت را پردازش می‌کند.
- مسیر خروج (/logout) کاربر را از جلسه خارج می‌کند و به صفحه‌ی ورود منتقل می‌کند.

۳.احراز هویت گوگل:

- در صورت ورود موفقیت‌آمیز از طریق گوگل، اطلاعات کاربر از API گوگل استخراج و در جلسه ذخیره می‌شود.

- اگر کاربر قبلاً در پایگاه داده ثبت نشده باشد، کاربر جدیدی با رمز عبور تصادفی ایجاد و در پایگاه داده ذخیره می‌شود.

۴.به اشتراک‌گذاری در تلگرام:

- مسیر (/share_on_telegram) پیامی را شامل اطلاعات کاربر فعلی ایجاد کرده و آن را برای اشتراک‌گذاری در تلگرام آماده می‌کند.

این کد مبنایی برای ساخت اپلیکیشن‌های امن و به‌روز در وب فراهم می‌کند که به احراز هویت قوی و اتصالات نرم‌افزاری با اپلیکیشن‌های شبکه‌های اجتماعی نیاز دارند .

FLASK در اینجا به عنوان ابزاری با انعطاف پذیری بالا و کمک به توسعه‌دهندگان در مهندسی عملیات مهم و حساس احراز هویت و انتقال داده‌ها، به کار رفته است. این مستندات به صورت دقیق و ساده‌شده روش‌های پیکربندی و عملیاتی کردن هر بخش را توضیح می‌دهد، تا توسعه‌دهندگان بتوانند با استفاده از این نمونه کد، وب‌اپلیکیشن‌های پیشرفته خود را بسازند و به بهترین شکل مدیریت کنند.

فایل Client_secret_***.json دارای مسیرها و توکن های گوگل برای دسترسی به اعتبار سنجی گوگل می باشد.

این فایل در Routing فراخوانی می شود تا در زمان فراخوانی callback استفاده گردد.

در پوشه templates فایل های HTML قرار دارد که با این فایل ها طراحی سمت کاربر شکل می گیرد.

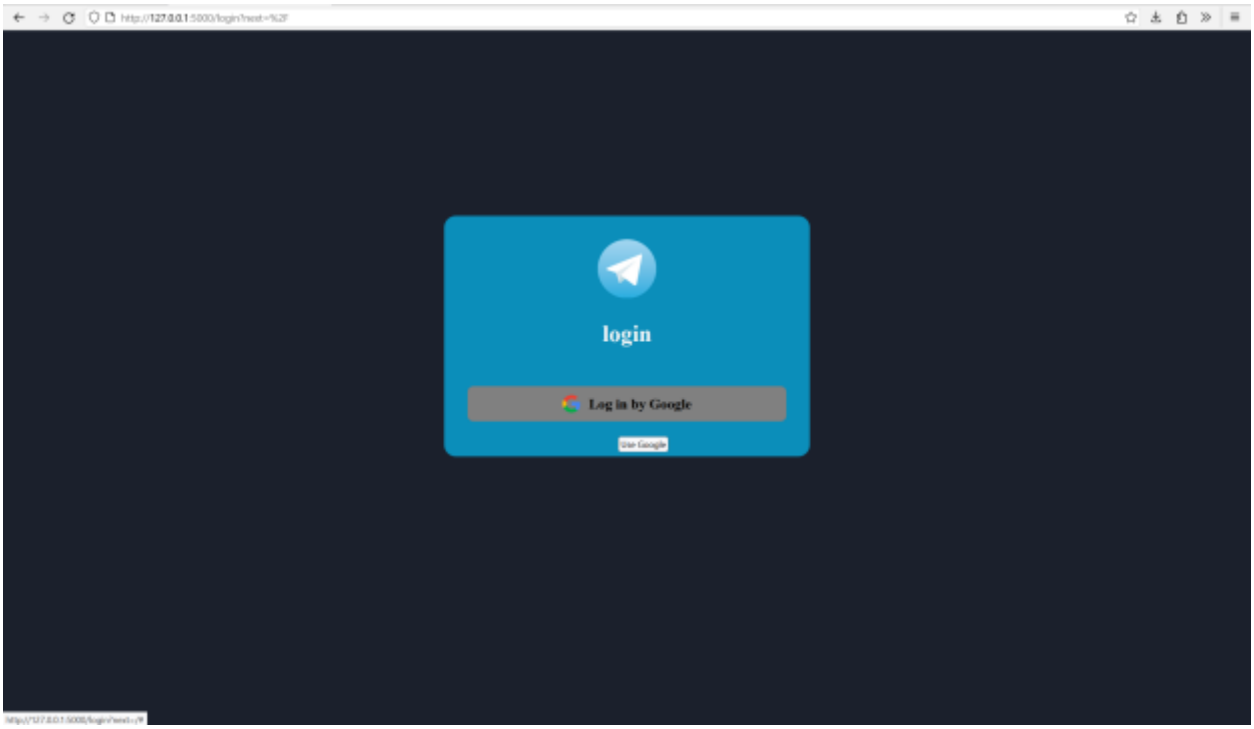
Base.html فایلی است که در آن طرح اولیه بک گراند و منو قرار دارد و توسط jinja2 در بقیه قالب ها گسترش می یابد.

و در Static فایل های طراحی css که مسئول زیبا کردن وب سایت و رسپانسیو سازی آن برای بهتر کردن تجربه کاربر می باشد قرار دارد و همچنین js که در آن کد های برنامه نویسی با زبان javascripts قرار دارد که با استفاده از آن پویا سازی و تعامل با کاربر افزایش می یابد و در رسپانسیو سازی هم کمک می کند.

و در pics تصاویر آیکون ها و... قرار دارد.

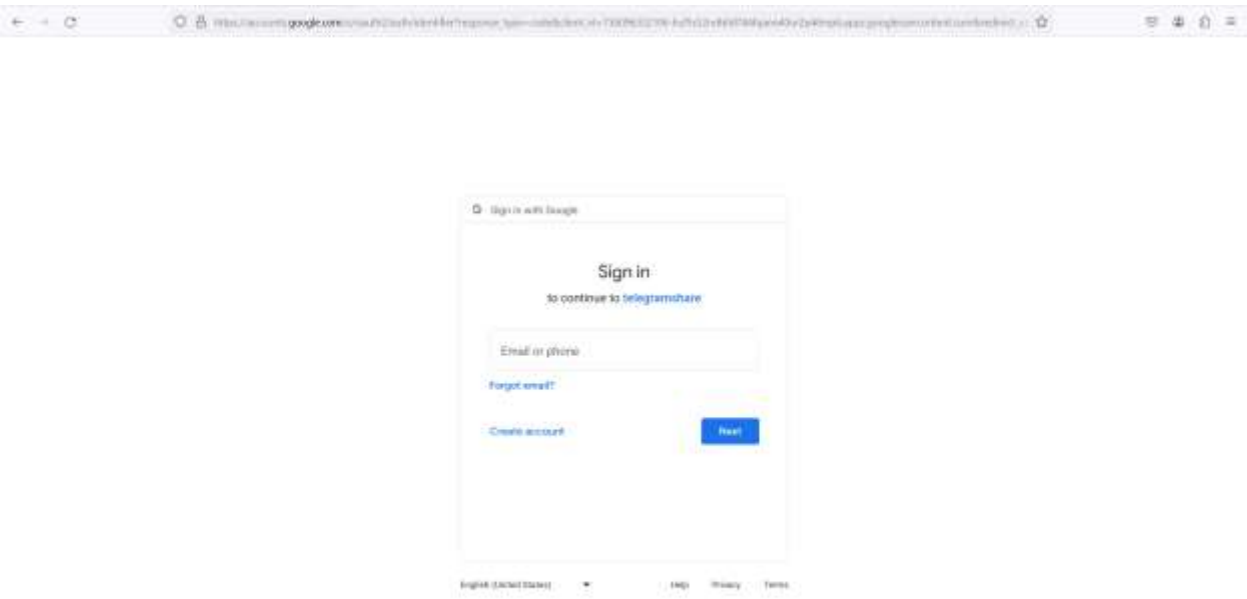
و models.py مسئول ساخت و نگهداری table های دیتابیس می باشد .

اعتبار سنجی گوگل :

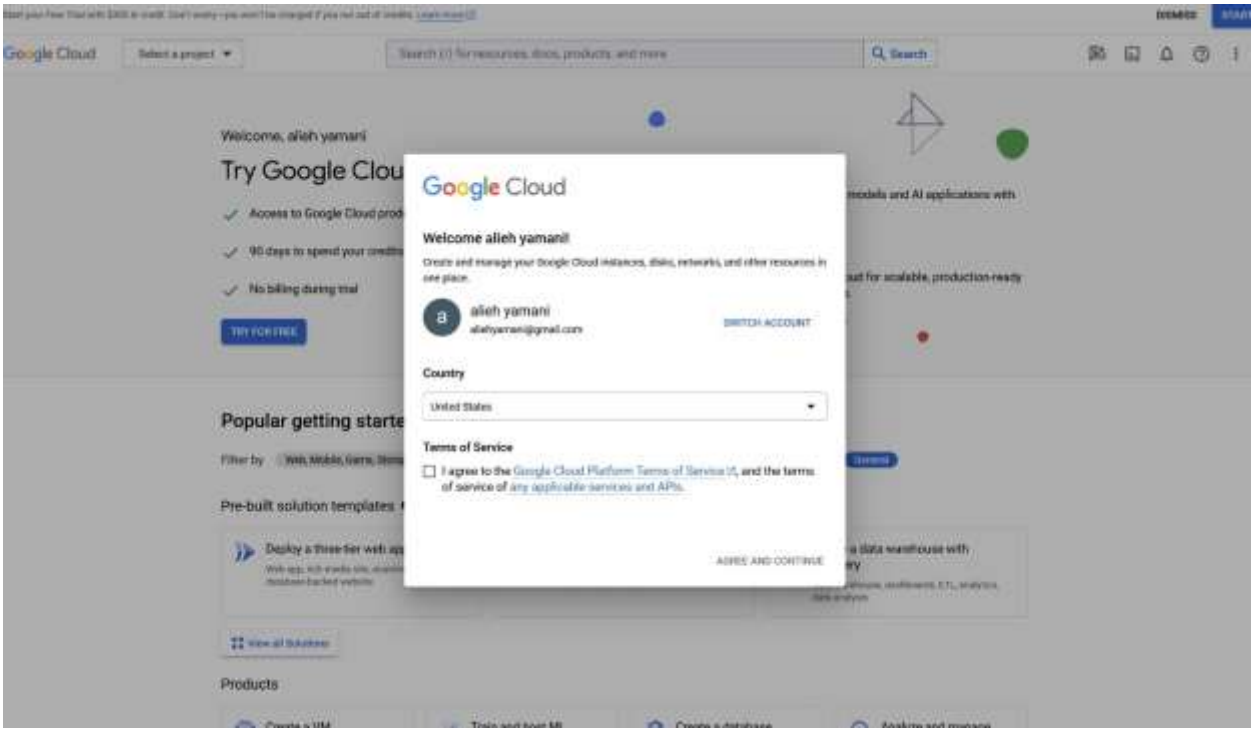


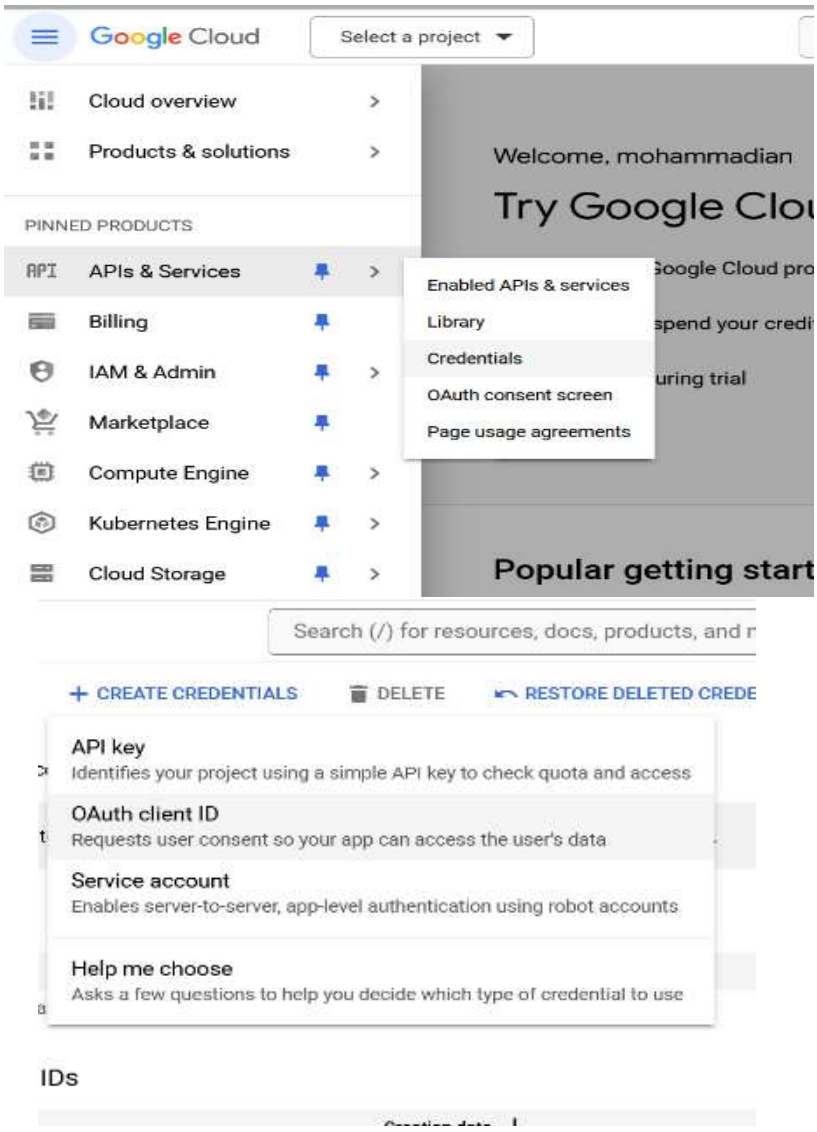
همانطور که در قبل ذکر کردیم در این وب سایت از احراز هویت گوگل برای ورود به بخش کاربری استفاده شده است.

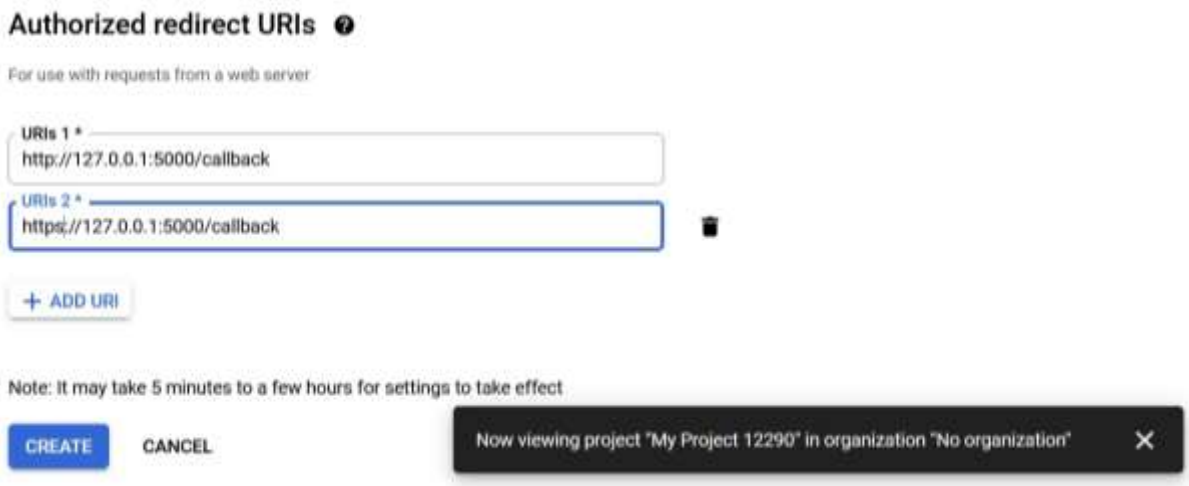
و برای استفاده از آن در صفحه لاگین بعد از انتخاب دکمه login by google تابع loginbygoogle در Routing.py فراخوانی می شود و بعد از گرفتن توکن ها و اعتبار سنجی آن صفحه ورود با گوگل به کاربر نمایش داده می شود و پس از وارد کردن رمز کاربر وارد صفحه خانه می شود.



برای دریافت فایل client_secret.json باید در googlecloud یک credential بسازیم و uri سایت را به آن تنظیم کنیم.




Google Cloud
Select a project
Cloud overview
Products & solutions
PINNED PRODUCTS
APIs & Services
Billing
IAM & Admin
Marketplace
Compute Engine
Kubernetes Engine
Cloud Storage
Enabled APIs & services
Library
Credentials
OAuth consent screen
Page usage agreements
Welcome, mohammadian
Try Google Clou
Popular getting start
Search (/) for resources, docs, products, and r
+ CREATE CREDENTIALS
DELETE
RESTORE DELETED CREDE
API key
Identifies your project using a simple API key to check quota and access
OAuth client ID
Requests user consent so your app can access the user's data
Service account
Enables server-to-server, app-level authentication using robot accounts
Help me choose
Asks a few questions to help you decide which type of credential to use
IDs



ارسال در تلگرام :

این برنامه Flask دارای یک تابع مشخص برای به اشتراک گذاری اطلاعات بر روی شبکه اجتماعی تلگرام است:

تابع:`share_on_telegram`

این تابع مسئولیت آماده‌سازی و به اشتراک گذاری اطلاعات کاربری از طریق تلگرام را بر عهده دارد. ابتدا، یک پیام متنی حاوی نام و ایمیل کاربر جاری تولید می‌شود. سپس این پیام به همراه لینک صفحه اصلی وب‌سایت که به صورت

خارجی (_external) تولید شده است، به فرمتی کدگذاری می‌شود که به خوبی توسط تلگرام تشخیص داده می‌شود.(URL encoding)

پس از آن، یک URL برای اشتراک گذاری در تلگرام ساخته می‌شود که حاوی هر دو قطعه کدگذاری شده‌ی پیام و لینک است. در نهایت، تابع یک هدایت (_redirect)به URL تلگرام را اجرا می‌کند، جایی که کاربر می‌تواند با یک کلیک پیام و لینک را در چت یا کانال خود در تلگرام به اشتراک بگذارد.

کد مربوط به این تابع در پایین آمده است:

```
@routing.route('/share_on_telegram')
@login_required
def share_on_telegram():
```

در این کد تولید پیام متنی با اطلاعات کاربر صورت می گیرد.

کدگذاری متن پیام برای استفاده درURL

```
encoded_message = quote_plus(message)
```

تولید URL کامل وبسایت برای به اشتراک‌گذاری

```
url_to_share = url_for('routing.home', _external=True)
```

تولید URL نهایی تلگرام برای اشتراک‌گذاری پیام و لینک

```
telegram_url=
f"https://t.me/share/url?url={quote_plus(url_to_share)}
&text={encoded_message}"
```

انجام هدایت به URL تلگرام

```
return redirect(telegram_url)
```

فرانت اند :

در ابتدا یک توضیح در مورد وایر فریم ها میدهیم ، وایرفریم‌ها اولین گام در فرایند طراحی صفحات وب هستند و در آنها تمرکز بر روی نحوه‌ی قرار گرفتن و کارکرد عناصر صفحه است تا زمینه‌ساز استایل‌دهی بعدی و نهایی شوند. وایرفریم‌ها اساسی‌ترین و خام‌ترین نقشه‌های طراحی به شمار می‌روند که جزئیاتی مانند رنگ و استایل دقیق را در بر نمی‌گیرند .

به طور خاص، وایرفریم صفحه `login.html` شامل موقعیت‌های تعیین‌شده برای لوگو، فیلدهای ورودی، و دکمه‌ها، از قبیل دکمه ورود با حساب کاربری گوگل خواهد بود بدون آنکه جزئیاتی مثل رنگ یا فونت مشخص شده باشد. هدف وایرفریم ارائه دیدگاهی است که چگونگی آرایش کلی عناصر و چیدمان کلی صفحه را مشخص می‌کند .

در مقایسه، فرانت‌اند به بخشی از وب‌سایت اشاره دارد که کاربران به طور مستقیم با آن تعامل دارند. فرانت‌اند شامل طراحی رابط کاربری، گرافیک، انیمیشن‌ها، و همچنین کدهای HTML, CSS و JavaScript می‌شود که رابط وب را زنده می‌کنند .

به عنوان مثال، فرانت‌اند صفحه `base.html` شامل موارد زیر است:

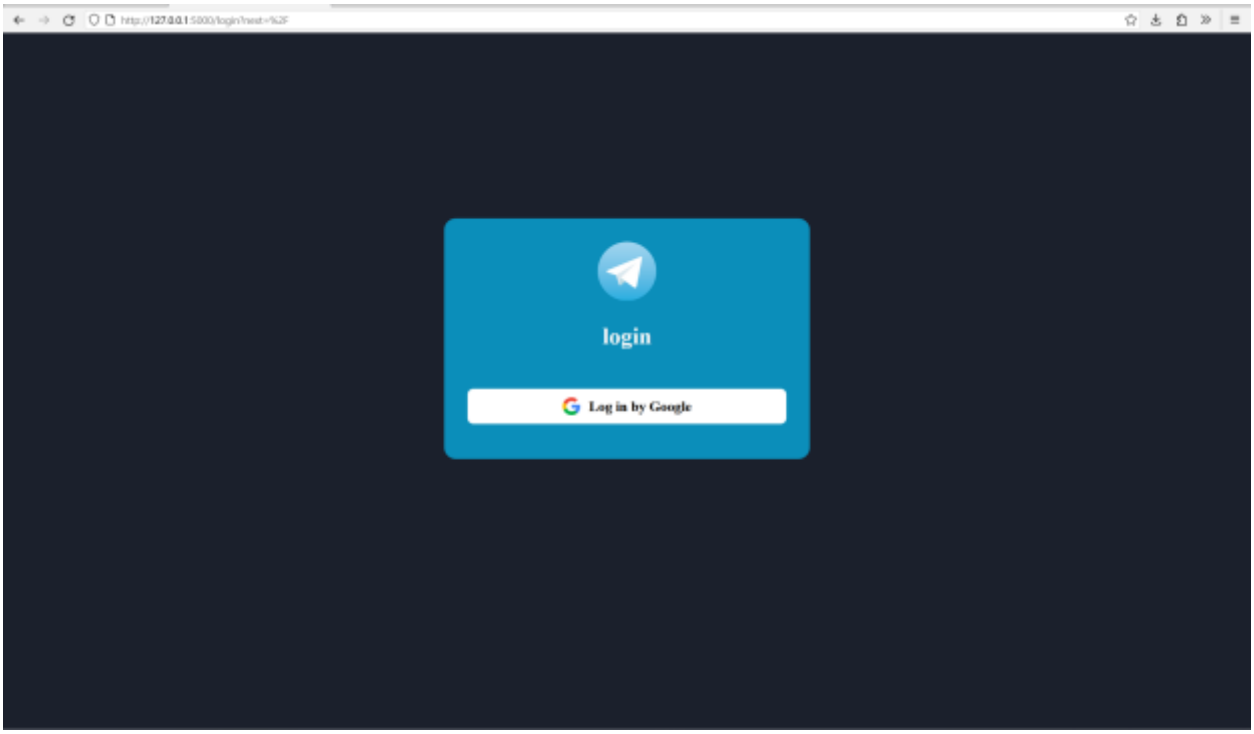
۱ :`HTML` .ساختاری پایه‌ریزی می‌کند برای تمام عناصر موجود در صفحه .

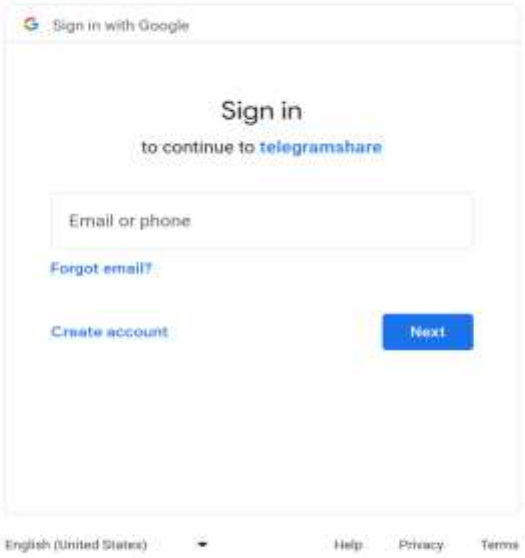
۲ :`CSS` .استایل‌دهی به عناصر HTML را انجام می‌دهد، مانند تنظیم رنگ‌ها، فونت‌ها و دیگر جزئیات طراحی .

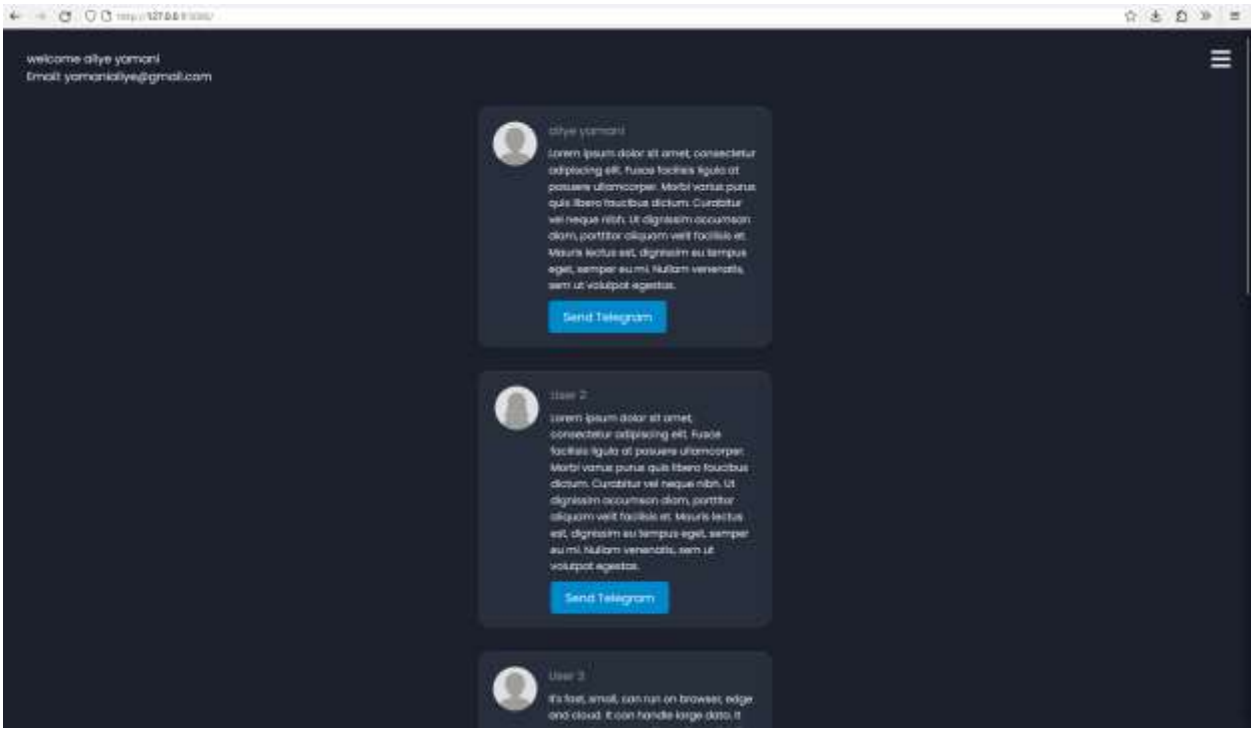
۳ .تصاویر و گرافیک‌ها: این بخش شامل لوگو و هر تصویر یا عنصر گرافیکی دیگر است که می‌تواند به صورت استاتیک یا ایجاد شده با CSS باشد.

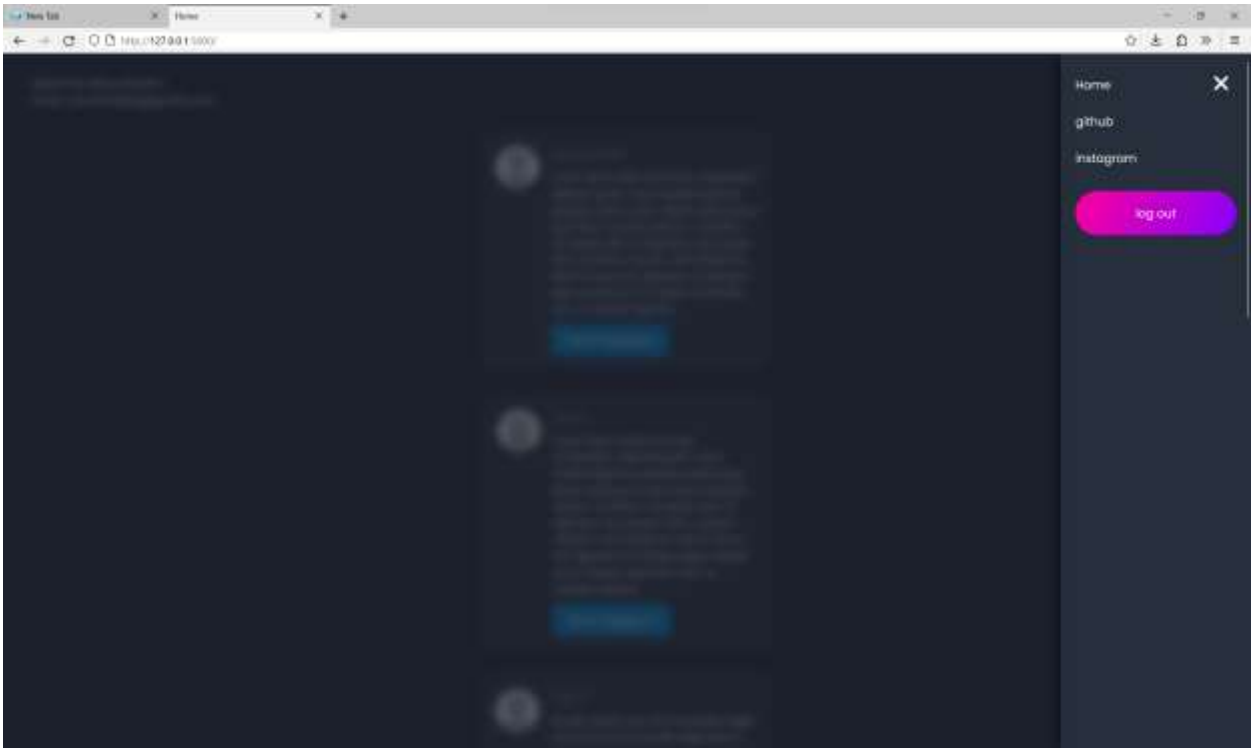
در طراحی فرانت‌اند برای `home.html` و `base.html`، تکنیک‌های توسعه فرانت‌اند مثل به کارگیری قالب‌هایی با قابلیت ارث‌بری (با استفاده از `{% extends "base.html" %}`) برای ایجاد صفحات دینامیک، استفاده از جینجا تمپلیت برای نمایش دادن داده‌ها مانند `{{ user.first_name }}` و پیوندها به صفحات دیگر با `url_for` در فلسک استفاده شده‌اند. این رویکرد انعطاف‌پذیری و قدرت فوق‌العاده‌ای به توسعه‌دهندگان می‌دهد تا بتوانند تجربیات کاربری جذاب و واکنش‌گرا ایجاد کنند.

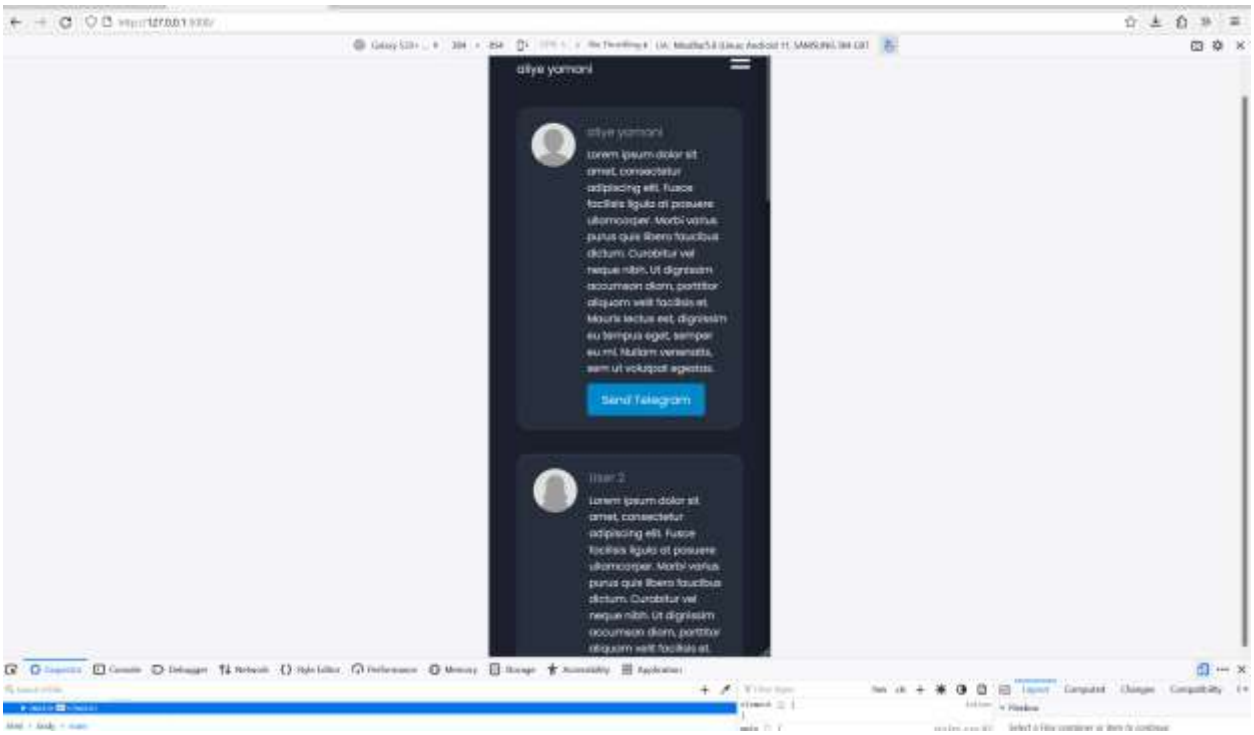
تصاویر تمرین:

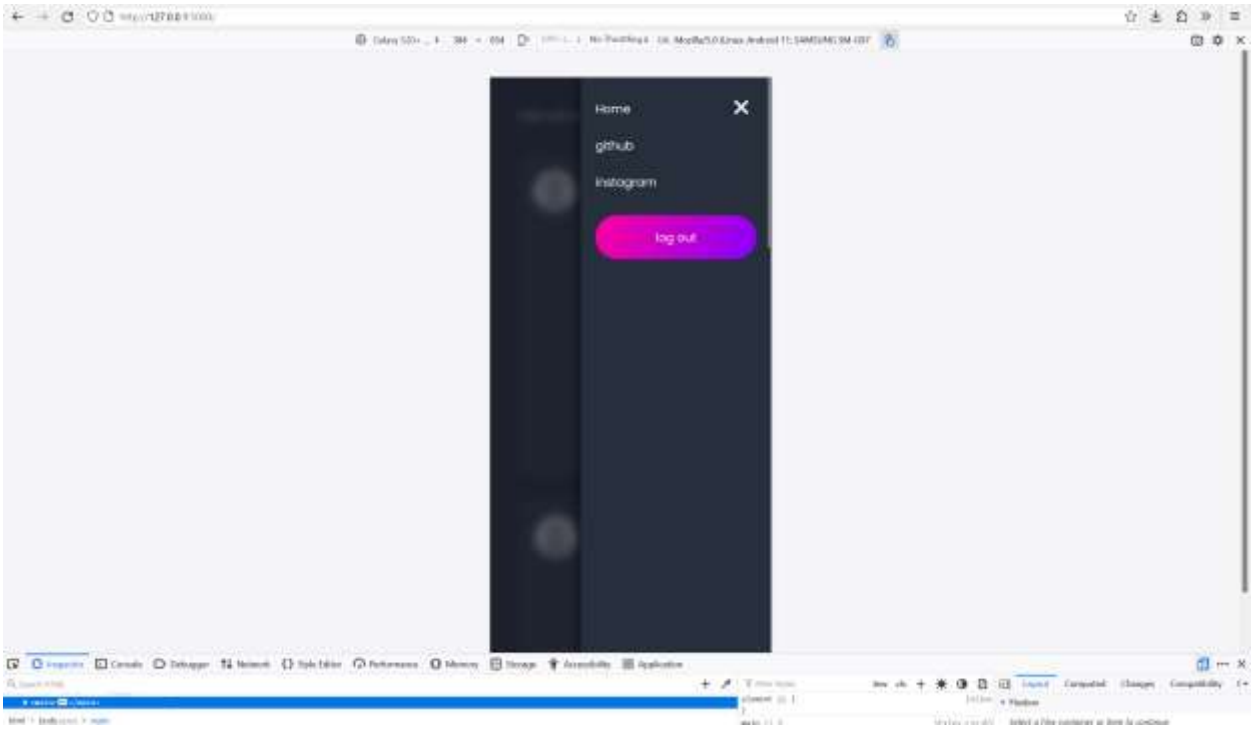





welcome aliye yamani
Email: yamanialiye@gmail.com
aliye yamani
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce facilisis ligula at posuere ullamcorper. Morbi varius purus quis libero faucibus dictum. Curabitur vel neque nibh. Ut dignissim accumsan diam, porttitor aliquam velit facilisis et. Mauris lectus est, dignissim eu tempus eget, semper eu mi. Nullam venenatis, sem ut volutpat egestas.
Send Telegram
User 2
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce facilisis ligula at posuere ullamcorper. Morbi varius purus quis libero faucibus dictum. Curabitur vel neque nibh. Ut dignissim accumsan diam, porttitor aliquam velit facilisis et. Mauris lectus est, dignissim eu tempus eget, semper eu mi. Nullam venenatis, sem ut volutpat egestas.
Send Telegram
User 3
it's fast, small, can run on browser, edge and cloud. it can handle large data. it


Home
github
instagram
log out


aliye yamani
aliye yamani
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce facilisis ligula at posuere ullamcorper. Morbi varius purus quis libero faucibus dictum. Curabitur vel neque nibh. Ut dignissim accumsan diam, porttitor aliquam velit facilisis et. Mauris lectus est, dignissim eu tempus eget, semper eu mi. Nullam venenatis, sem ut volutpat egestas.
Send Telegram
User 2
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce facilisis ligula at posuere ullamcorper. Morbi varius purus quis libero faucibus dictum. Curabitur vel neque nibh. Ut dignissim accumsan diam, porttitor aliquam velit facilisis et.


Home
github
instagram
log out