



UANL

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN



FCFM

FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

Desarrollo de Software Seguro

Actividad práctica (asíncrona): Aplicación con login de Google (AuthN)

Estudiante: Osmar Abelardo Bustos Vázquez

Carrera: Lic. en Seguridad de Tecnologías de Información

Matrícula: 1912361

Grupo: 064

Docente: M.C. Romeo Alfonso Sánchez López

6 de noviembre 2023

AGOSTO-DICIEMBRE-2023

Login seguro con Google

Enlace de página en Github: <https://github.com/osm4r/DDSCross-Site-Request-Forgery>

Aprendizajes:

- Uso de OAuth2
- Uso de OpenID Connect (OIDC)
- Creación de un login basado en Google
- Reforzamiento de conocimiento de sqlite
- Manera de realizar un login más seguro sin la necesidad de administrar contraseñas

Dificultades:

Tuve varias dificultades, dos en específico. La primera fue la manera de instalar los módulos ya que debido a las versiones se generaban problemas de compatibilidad, y por esto mismo, no funcionaba la aplicación. Opté por quitar las versiones en el archivo del requirements para que se instalara la versión más nueva.

También tuve otra dificultad al finalizar la actividad mediante la realización del código expuesto en el documento pdf. Sin embargo, revisando los archivos de código que nos proporcionó el maestro, logré identificar que faltaban algunas líneas, y lo complementé y ya funcionó.

Pruebas y capturas de pantalla:

Start your Free Trial with \$300 in credit. Don't worry—you won't be charged if you run out of credits. [Learn more](#)

DISMISS [START FREE](#)

Google Cloud

DDS Aplicacion login Google

Search (/) for resources, docs, products, an...

Search

0

API APIs & Services

Enabled APIs & services

Library

Credentials

OAuth consent screen

Page usage agreements

Edit app registration

OAuth consent screen — Scopes — Test users — **4 Summary**

OAuth consent screen

EDIT

User type

External


App name

OsmaLabDSS

Support email

osmarfishy@gmail.com

App logo



Application homepage link

Not provided

Google Cloud

DDS Aplicacion login Google

Search (/) for resources, docs, produ...

Search

0

API APIs & Services

Enabled APIs & services

Library

Credentials

OAuth consent screen

Page usage agreements

Credentials

+ CREATE CREDENTIALS

DELETE




RESTORE DELETED CREDENTIALS

Create credentials to access your enabled APIs. [Learn more](#)

API Keys

<input type="checkbox"/>	Name	Creation date ↓	Restrictions	Actions
No API keys to display				

OAuth 2.0 Client IDs

<input type="checkbox"/>	Name	Creation date ↓	Type	Client ID	Actions
<input type="checkbox"/>	LabDSS	Nov 6, 2023	Web application	1824847144826-cgp...	  

Service Accounts

[Manage service accounts](#)

<input type="checkbox"/>	Email	Name ↑	Actions
No service accounts to display			

OABV

3

The screenshot displays the Visual Studio Code interface for a project named "DDS-APLICACION-LOGIN-...". The Explorer sidebar on the left shows the project structure with files: `env`, `.gitattributes`, `app.py` (4, U), `db.py` (3, U), `README.md`, `requirements.txt` (U), `schema.sql` (U), and `user.py` (1, U). The main editor area contains four open files:

- requirements.txt**:

```
1 requests==2.21.0
2 flask==1.0.2
3 gauthlib==3.0.1
4 pyOpenSSL==19.0.0
5 Flask-Login==0.4.1
```
- db.py**:

```
1 # http://flask.pocoo.org
2
3 import sqlite3
4 import click
5 from flask import current_app
6 from flask.cli import with_appcontext
7
8
9 def get_db():
10     if "db" not in g:
11         g.db = sqlite3.connect(
12             "sqlite_db",
13         )
14     g.db.row_factory = sqlite3.Row
15     return g.db
16
17 def close_db(e=None):
18     db = g.pop("db", None)
19     if db is not None:
20         db.close()
21
22 def init_db():
23     db = get_db()
```
- schema.sql**:

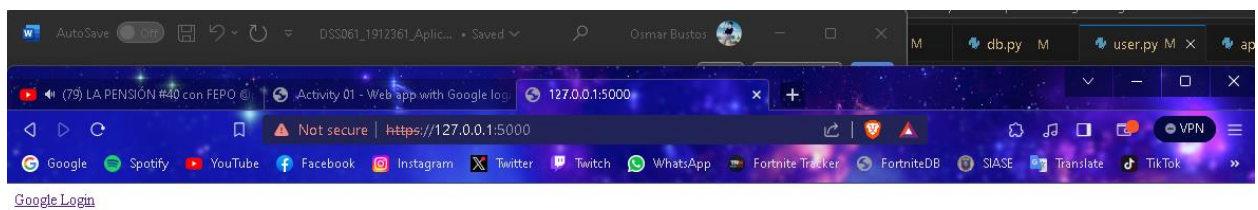
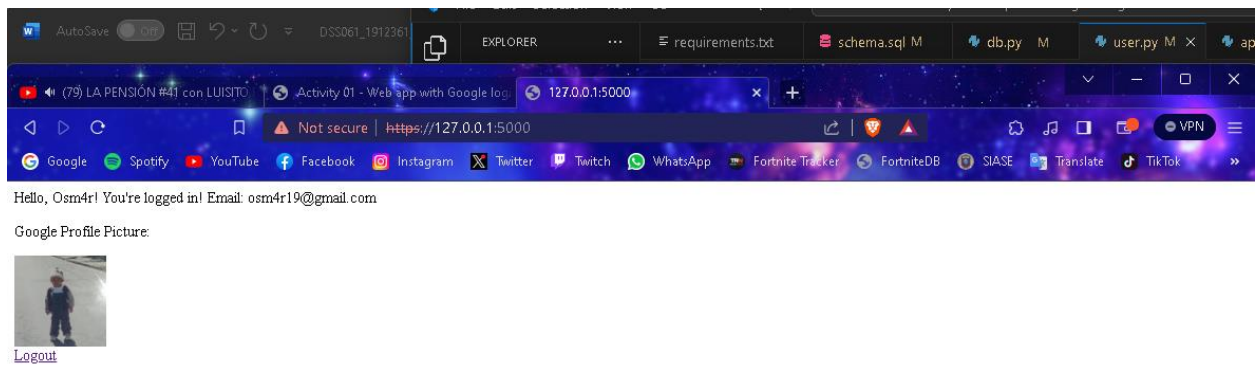
```
1 CREATE TABLE user (
2     id TEXT PRIMARY KEY,
3     name TEXT NOT NULL,
4     email TEXT UNIQUE NOT NULL,
5     profile_pic TEXT NOT NULL,
6 );
```
- user.py**:

```
1 from flask_login import UserMixin
2 from db import get_db
3
4 class User(UserMixin):
5     def __init__(self, id_, name, email, profile_pic):
6         self.id = id_
7         self.name = name
8         self.email = email
9         self.profile_pic = profile_pic
10
11 @staticmethod
12 def get(user_id):
13     db = get_db()
14     user = db.execute(
15         "SELECT * FROM user WHERE id = ?"
16     ).fetchone()
17     if not user:
18         return None
19     user = User(
20         id=user[0], name=user[1], email=user[2], profile_pic=user[3]
21     )
22     return user
```
- app.py**:

```
1 # Python standard libraries
2 import json
3 import os
4 import sqlite3
5 # Third-party libraries
6 from flask import Flask, redirect, request
7 from flask_login import (
8     LoginManager,
9     current_user,
10     login_required,
11     login_user,
12     logout_user,
13 )
14 from gauthlib.oauth2 import WebApplication
15 import requests
16 # Internal imports
17 from db import init_db_command
18 from user import User
19
20 # Configuration
21 GOOGLE_CLIENT_ID = os.environ.get("GOOGLE_CLIENT_ID")
22 GOOGLE_CLIENT_SECRET = os.environ.get("GOOGLE_CLIENT_SECRET")
```

The status bar at the bottom indicates the active file is `main.py` at line 5, column 19, with a UTF-8 encoding and CRLF line endings.

```
114 @app.route("/login/callback")
115 def callback():
116     # Get authorization code Google sent back to you
117     code = request.args.get("code")
118
119     # Find out what URL to hit to get tokens that allow you to ask for
120     # things on behalf of a user
121     google_provider_cfg = get_google_provider_cfg()
122     token_endpoint = google_provider_cfg["token_endpoint"]
123
124     # Prepare and send a request to get tokens! Yay tokens!
125     token_url, headers, body = client.prepare_token_request(
126         token_endpoint,
127         authorization_response=request.url,
128         redirect_url=request.base_url,
129         code=code
130     )
131     token_response = requests.post(
132         token_url,
133         headers=headers,
134         data=body,
135         auth=(GOOGLE_CLIENT_ID, GOOGLE_CLIENT_SECRET),
136     )
137
138     # Parse the tokens!
139     client.parse_request_body_response(json.dumps(token_response.json()))
140
141     # Now that you have tokens (yay) let's find and hit the URL
142     # from Google that gives you the user's profile information,
143     # including their Google profile image and email
144     userinfo_endpoint = google_provider_cfg["userinfo_endpoint"]
145     url, headers, body = client.add_token(userinfo_endpoint)
146     userinfo_response = requests.get(url, headers=headers, data=body)
147
148     # You want to make sure their email is verified.
149     # The user authenticated with Google, authorized your
150     # app, and now you've verified their email through Google!
151     if userinfo_response.json().get("email_verified"):
152         unique_id = userinfo_response.json()["sub"]
153         users_email = userinfo_response.json()["email"]
154         picture = userinfo_response.json()["picture"]
155         users_name = userinfo_response.json()["given_name"]
156     else:
157         return "User email not available or not verified by Google.", 400
158
159     # Create a user in your db with the information provided by Google
160     user = User(
161         id=unique_id, name=users_name, email=users_email, profile_pic=picture
162     )
163
164     # Doesn't exist? Add it to the database.
165     if not User.get(unique_id):
166         User.create(unique_id, users_name, users_email, picture)
167
168     # Begin user session by logging the user in
169     login_user(user)
170
171     # Send user back to homepage
172     return redirect(url_for("index"))
173
174
175 @app.route("/logout")
176 @login_required
177 def logout():
178     logout_user()
179     return redirect(url_for("index"))
180
181
182 if __name__ == "__main__":
183     app.run(debug=True)
```





Hello, Osmar! You're logged in! Email: osmarfishy@gmail.com

Google Profile Picture:



[Logout](#)

```
Command Prompt - python <
&prompt=consent HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:08:02] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:08:04] "GET /logout HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:08:04] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:08:07] "GET /login HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:08:22] "GET /login/callback?code=4/0AfJohXl0-QCTr5AgoYl_GFYku-zYNmjCA6yCUl5fhopHqJtXq19F8zwwluxAU-AkD0ZEBg&scope=email+profile+openid+https://www.googleapis.com/auth/userinfo.email+https://www.googleapis.com/auth/userinfo.profile&authuser=3&prompt=consent HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:08:22] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:08:59] "GET /logout HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:08:59] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:09:00] "GET /login HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:09:03] "GET /login/callback?code=4/0AfJohXn5gZlWM8Dtkt88YHytzAUlqEwdZma5QqkKJhex-4sBj97LyG0bv7K1Csha4m7_Gg&scope=email+profile+openid+https://www.googleapis.com/auth/userinfo.email+https://www.googleapis.com/auth/userinfo.profile&authuser=3&prompt=consent HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:09:03] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:10:31] "GET /logout HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:10:31] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:10:32] "GET /login HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:10:40] "GET /login/callback?code=4/0AfJohXn0Kq81V_wJSZSZjRNN3r3fo4AXZ_eHtNn5locz_M7WYa9Rp8H4fLY6zwBdRNLNQ&scope=email+profile+openid+https://www.googleapis.com/auth/userinfo.email+https://www.googleapis.com/auth/userinfo.profile&authuser=1&prompt=consent HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:10:40] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:10:43] "GET /logout HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:10:43] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:10:45] "GET /login HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:10:59] "GET /login/callback?code=4/0AfJohXk4MIgs9qRd03uQWMDpNhxzIi4lcNzHEZM261dTu_LwTrtiahV9DkiX7bIWV_5jw&scope=email+profile+openid+https://www.googleapis.com/auth/userinfo.email+https://www.googleapis.com/auth/userinfo.profile&authuser=0&prompt=consent HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:10:59] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Nov/2023 21:11:00] "GET /logout HTTP/1.1" 302 -
127.0.0.1 - - [06/Nov/2023 21:11:00] "GET / HTTP/1.1" 200 -
```

Reflexión final (conclusión):

En conclusión, pienso que esta actividad práctica es fue muy importante para mejorar mi conocimiento en cuanto a la creación de un login más seguro con Google que, así se pueden evitar muchas de las complicaciones y responsabilidades asociadas con la gestión de contraseñas y la seguridad de las cuentas de usuario en una aplicación web.

Esto porque en lugar de pedirle a los usuarios que recuerden otras contraseñas y se estresen reteniendo más información, mejor con esta solución delega la autenticación y la seguridad a Google completamente.

Además, pienso que el conocimiento de OAuth 2 y OpenID Connect es importante para lograr una comunicación con Google para así permitir que los usuarios se autenticuen de manera segura.

Durante la realización de la actividad, tuve varias dificultades que afectaron el desarrollo eficaz de la tarea. Estas fueron problemas de compatibilidad de versiones al instalar módulos, lo que resultó en la falta de funcionalidad de la aplicación y la falta de código en un archivo.

En cuanto a las dificultades que tuve, me estresé un poco sin embargo me gustó ya que hay que adaptarse a los desafíos que surgen durante la programación en general. La resolución de problemas, la investigación que realicé y la ayuda del profesor mediante los archivos proporcionados, fueron clave para completar la actividad con éxito de manera eficiente.

Bibliografía:

Define and access the database — flask documentation (3.0.X). (n.d.).

Palletsprojects.com. Retrieved November 7, 2023, from

<https://flask.palletsprojects.com/en/3.0.x/tutorial/database/>