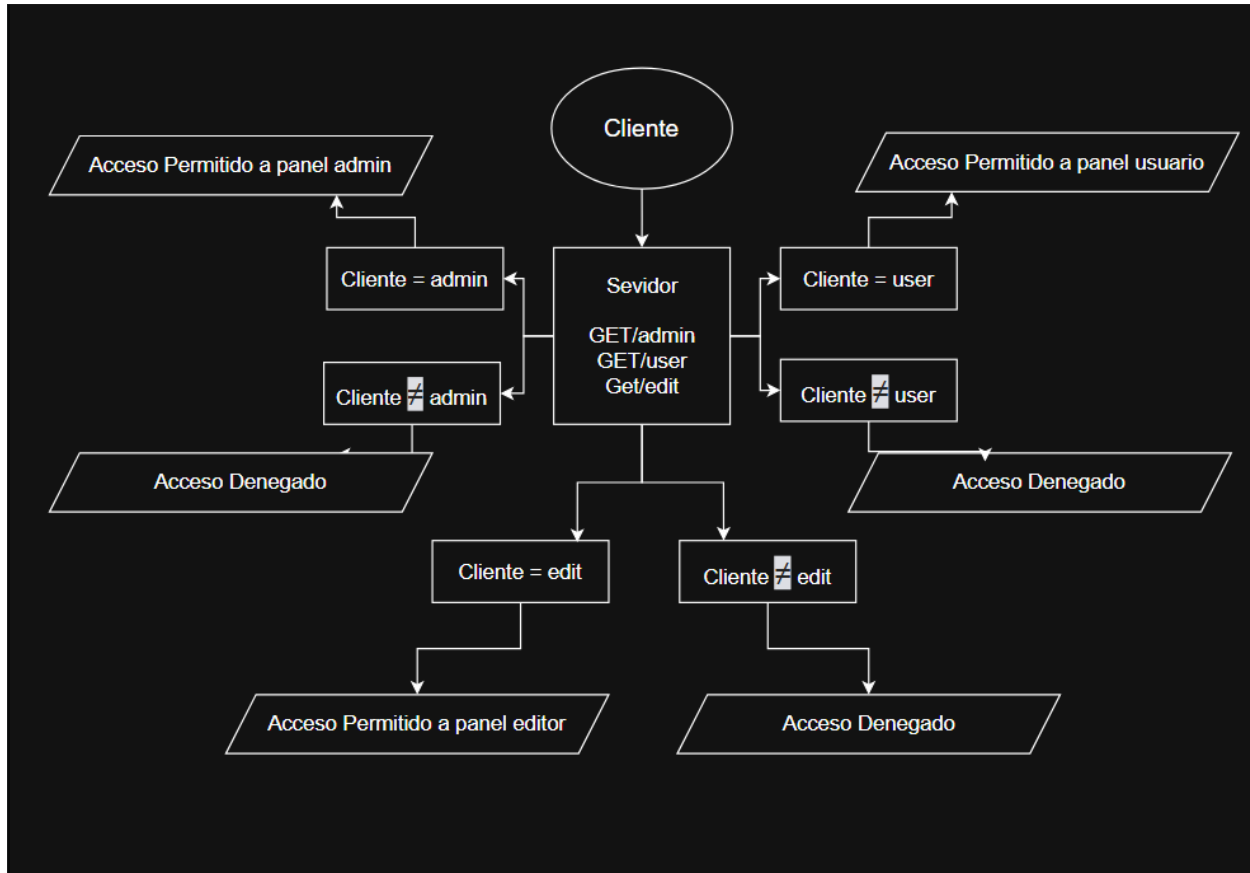


Actividad – Modulo3-Lección2:

Diagrama:



Cuando el usuario es un administrador este se le permite acceso al solamente al panel de administrador, pero si intenta entrar al panel de usuario o editor este no tiene acceso:

```

app.py
Actividad-Modulo3-leccion2 > app.py > ...
6 app.config["SECRET_KEY"] = "mi_clave_secreta"
7
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(roleNeed("admin"))
13 user_permission = Permission(roleNeed("user"))
14 edit_permission = Permission(roleNeed("edit"))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "admin"} # Cambiar el r
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Iden
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Iden
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Iden
  
```

```

request.http
Actividad-Modulo3-leccion2 > request.http > GET /edit
5
6 # Probar el acceso al panel de administración
7 Send Request
8 GET http://127.0.0.1:5000/admin
9
10 ###
11 # Probar el acceso al panel de usuario
12 Send Request
13 GET http://127.0.0.1:5000/user
14
15 ###
16 # Probar el acceso a la página de edición
17 Send Request
18 GET http://127.0.0.1:5000/edit
  
```

```

Response(12ms)
1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:12:19 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 61
6 Vary: Cookie
7 Set-Cookie: session=eyJpZGVudGloeS5hdXRoX3R5cGUjOm51bG
wsImlkZW50aXR5ImkiJoiYmRTaM4iFq.aAMpaw.mXscV0596V_D6I
Vd-CP8011hg-A; HttpOnly; Path=/
8 Connection: close
9
10 Panel de Administrador: Acceso permitido para administ
radores
  
```

```
app.py x ... request.http x ... Response(9ms) x
Actividad-Modulo3-leccion2 > app.py > ...
6 app.config["SECRET_KEY"] = "mi_clave_secreta"
7
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(RoleNeed('admin'))
13 user_permission = Permission(RoleNeed('user'))
14 edit_permission = Permission(RoleNeed('edit'))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "admin"} # Cambiar el r
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Iden
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Iden
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Iden
28     else:
29         identity_changed.send(app, identity=Anor

```

```
request.http x ...
Actividad-Modulo3-leccion2 > request.http > GET /edit
5
6 # Probar el acceso al panel de administración
7 GET http://127.0.0.1:5000/admin
8
9 ###
10
11 # Probar el acceso al panel de usuario
12 GET http://127.0.0.1:5000/user
13
14 ###
15
16 # Probar el acceso a la página de edición
17 GET http://127.0.0.1:5000/edit

```

```
Response(9ms) x
1 HTTP/1.1 403 FORBIDDEN
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:10:37 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 50
6 Vary: Cookie
7 Set-Cookie: session=eyJpZGVudGloeShdXRoX3R5cGU0w51bG
wsImlkZm50aXR5LmlkIjo1YmRTaW1fQ.aaWpHq.22GaGZB3r5IOCr
WkFKBmrIDm1rw; HttpOnly; Path=/
8 Connection: close
9
10 Acceso Denegado: No tienes los permisos necesarios

```

```
app.py x ... request.http x ... Response(9ms) x
Actividad-Modulo3-leccion2 > app.py > ...
6 app.config["SECRET_KEY"] = "mi_clave_secreta"
7
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(RoleNeed('admin'))
13 user_permission = Permission(RoleNeed('user'))
14 edit_permission = Permission(RoleNeed('edit'))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "admin"} # Cambiar el r
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Iden
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Iden
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Iden

```

```
request.http x ...
Actividad-Modulo3-leccion2 > request.http > GET /edit
5
6 # Probar el acceso al panel de administración
7 GET http://127.0.0.1:5000/admin
8
9 ###
10
11 # Probar el acceso al panel de usuario
12 GET http://127.0.0.1:5000/user
13
14 ###
15
16 # Probar el acceso a la página de edición
17 GET http://127.0.0.1:5000/edit

```

```
Response(9ms) x
1 HTTP/1.1 403 FORBIDDEN
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:11:47 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 50
6 Vary: Cookie
7 Set-Cookie: session=eyJpZGVudGloeShdXRoX3R5cGU0w51bG
wsImlkZm50aXR5LmlkIjo1YmRTaW1fQ.aaWpYw.s8fMrbQW81vsY
vPcw0cai_ABRy; HttpOnly; Path=/
8 Connection: close
9
10 Acceso Denegado: No tienes los permisos necesarios

```

Cuando el usuario es un “user” este tan solo puede entrar al panel de usuarios, pero este tiene restringido los paneles de administrador y editor:

```
app.py x ... request.http x ... Response(6ms) x
Actividad-Modulo3-leccion2 > app.py > ...
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(RoleNeed('admin'))
13 user_permission = Permission(RoleNeed('user'))
14 edit_permission = Permission(RoleNeed('edit'))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "user"} # Cambiar el rc
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Iden
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Iden
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Iden
28     else:
29         identity_changed.send(app, identity=Anor
30
31 # Ruta principal accesible para todos

```

```
request.http x ...
Actividad-Modulo3-leccion2 > request.http > GET /edit
6 # Probar el acceso al panel de administración
7 GET http://127.0.0.1:5000/admin
8
9 ###
10
11 # Probar el acceso al panel de usuario
12 GET http://127.0.0.1:5000/user
13
14 ###
15
16 # Probar el acceso a la página de edición
17 GET http://127.0.0.1:5000/edit

```

```
Response(6ms) x
1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:15:33 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 58
6 Vary: Cookie
7 Set-Cookie: session=eyJpZGVudGloeShdXRoX3R5cGU0w51bG
wsImlkZm50aXR5LmlkIjo1YmRTaW1fQ.aaWpYw.RGK-kq589x3dy3Bs
Bmml2EQZAc; HttpOnly; Path=/
8 Connection: close
9
10 Panel de Usuario: Acceso permitido para usuarios regul
ares

```

```
app.py x request.http x Response(6ms) x
Actividad-Modulo3-leccion2 > app.py > ...
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(RoleNeed('admin'))
13 user_permission = Permission(RoleNeed('user'))
14 edit_permission = Permission(RoleNeed('edit'))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "user"} # Cambiar el rol
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Id
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Id
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Id
28     else:
29         identity_changed.send(app, identity=Anon
30
31 # Ruta principal accesible para todos
32 @app.route("/")
33 def index():
34     return render_template("index.html")
35
36 request.http x
6 # Probar el acceso al panel de administración
7 Send Request
8 GET http://127.0.0.1:5000/admin
9
10 ###
11 # Probar el acceso al panel de usuario
12 Send Request
13 GET http://127.0.0.1:5000/user
14
15 ###
16 # Probar el acceso a la página de edición
17 Send Request
18 GET http://127.0.0.1:5000/edit
19
20 Response(6ms) x
1 HTTP/1.1 403 FORBIDDEN
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:13:57 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 50
6 Vary: Cookie
7 Set-Cookie: session=eyJpZGVudG10eS5hdXRoX3R5cGU0w51bG
wsImlkZ50aXR5ImkiJoi2XNlci99.aWp5Q0.RCtGhCpEXBZx1v7P
yOsoX98Za1w; HttpOnly; Path=/
8 Connection: close
9
10 Acceso Denegado: No tienes los permisos necesarios
```

```
app.py x request.http x Response(12ms) x
Actividad-Modulo3-leccion2 > app.py > ...
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(RoleNeed('admin'))
13 user_permission = Permission(RoleNeed('user'))
14 edit_permission = Permission(RoleNeed('edit'))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "user"} # Cambiar el rol
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Id
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Id
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Id
28     else:
29         identity_changed.send(app, identity=Id
30
31 # Ruta principal accesible para todos
32 @app.route("/")
33 def index():
34     return render_template("index.html")
35
36 request.http x
6 # Probar el acceso al panel de administración
7 Send Request
8 GET http://127.0.0.1:5000/admin
9
10 ###
11 # Probar el acceso al panel de usuario
12 Send Request
13 GET http://127.0.0.1:5000/user
14
15 ###
16 # Probar el acceso a la página de edición
17 Send Request
18 GET http://127.0.0.1:5000/edit
19
20 Response(12ms) x
1 HTTP/1.1 403 FORBIDDEN
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:16:47 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 50
6 Vary: Cookie
7 Set-Cookie: session=eyJpZGVudG10eS5hdXRoX3R5cGU0w51bG
wsImlkZ50aXR5ImkiJoi2XNlci99.aWp5Q0.RCtGhCpEXBZx1v7P
yOsoX98Za1w; HttpOnly; Path=/
8 Connection: close
9
10 Acceso Denegado: No tienes los permisos necesarios
```

Cuando un usuario es un editor este tiene permitido el panel de editor solamente, pero esta restringido el panel de administrador y usuario:

```
app.py x request.http x Response(9ms) x
Actividad-Modulo3-leccion2 > app.py > ...
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(RoleNeed('admin'))
13 user_permission = Permission(RoleNeed('user'))
14 edit_permission = Permission(RoleNeed('edit'))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "edit"} # Cambiar el rol
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Id
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Id
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Id
28     else:
29         identity_changed.send(app, identity=Anon
30
31 # Ruta principal accesible para todos
32 @app.route("/")
33 def index():
34     return render_template("index.html")
35
36 request.http x
6 # Probar el acceso al panel de administración
7 Send Request
8 GET http://127.0.0.1:5000/admin
9
10 ###
11 # Probar el acceso al panel de usuario
12 Send Request
13 GET http://127.0.0.1:5000/user
14
15 ###
16 # Probar el acceso a la página de edición
17 Send Request
18 GET http://127.0.0.1:5000/edit
19
20 Response(9ms) x
1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:18:42 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 37
6 Vary: Cookie
7 Set-Cookie: session=eyJpZGVudG10eS5hdXRoX3R5cGU0w51bG
wsImlkZ50aXR5ImkiJoi2XNlci99.aWp5Q0.RCtGhCpEXBZx1v7P
yOsoX98Za1w; HttpOnly; Path=/
8 Connection: close
9
10 Página de Edición: Acceso permitido
```

```
app.py x request.http x Response(6ms) x
Actividad-Modulo3-leccion2 > app.py > ...
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(RoleNeed('admin'))
13 user_permission = Permission(RoleNeed('user'))
14 edit_permission = Permission(RoleNeed('edit'))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "edit"} # Cambiar el rc
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Iden
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Iden
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Iden
28     else:
29         identity_changed.send(app, identity=Anon
30
31 # Ruta principal accesible para todos

Actividad-Modulo3-leccion2 > request.http > GET /edit
6 # Probar el acceso al panel de administración
7 Send Request
8 GET http://127.0.0.1:5000/admin
9 ###
10 # Probar el acceso al panel de usuario
11 Send Request
12 GET http://127.0.0.1:5000/user
13 ###
14 # Probar el acceso a la página de edición
15 Send Request
16 GET http://127.0.0.1:5000/edit
17

1 HTTP/1.1 403 FORBIDDEN
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:19:07 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 50
6 Vary: Cookie
7 Set-Cookie: session-eyJpZGVudG0eS5hdXR5X3R5cGU0eS1lbg
wsImlkZy50aXR5LmlkIjo1ZWRpdCJ9.aWwRw.Lx3Fzly2Sic05wKS
ZyEK5gsV1Y; HttpOnly; Path=/
8 Connection: close
9
10 Acceso Denegado: No tienes los permisos necesarios
```

```
app.py x request.http x Response(9ms) x
Actividad-Modulo3-leccion2 > app.py > ...
8 # Configuración de Flask-Principal
9 Principal(app)
10
11 # Crear permisos basados en roles
12 admin_permission = Permission(RoleNeed('admin'))
13 user_permission = Permission(RoleNeed('user'))
14 edit_permission = Permission(RoleNeed('edit'))
15
16 # Simular un usuario con roles (modificar según
17 current_user = {"role": "edit"} # Cambiar el rc
18
19 # Configuración dinámica de identidad basada en
20 @app.before_request
21 def set_identity():
22     if current_user["role"] == "admin":
23         identity_changed.send(app, identity=Iden
24     elif current_user["role"] == "user":
25         identity_changed.send(app, identity=Iden
26     elif current_user["role"] == "edit":
27         identity_changed.send(app, identity=Iden
28     else:
29         identity_changed.send(app, identity=Anon
30
31 # Ruta principal accesible para todos
32 @app.route("/")

Actividad-Modulo3-leccion2 > request.http > GET /edit
6 # Probar el acceso al panel de administración
7 Send Request
8 GET http://127.0.0.1:5000/admin
9 ###
10 # Probar el acceso al panel de usuario
11 Send Request
12 GET http://127.0.0.1:5000/user
13 ###
14 # Probar el acceso a la página de edición
15 Send Request
16 GET http://127.0.0.1:5000/edit
17

1 HTTP/1.1 403 FORBIDDEN
2 Server: Werkzeug/3.1.3 Python/3.13.3
3 Date: Mon, 21 Apr 2025 02:19:29 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 50
6 Vary: Cookie
7 Set-Cookie: session-eyJpZGVudG0eS5hdXR5X3R5cGU0eS1lbg
wsImlkZy50aXR5LmlkIjo1ZWRpdCJ9.aWwRw.D3d2A_BB7H0DBP2s
Ym1hQWCFn0; HttpOnly; Path=/
8 Connection: close
9
10 Acceso Denegado: No tienes los permisos necesarios
```

Enlace al repositorio de Github:

https://github.com/BenyahirMartinez2004/Comp_2052/tree/main/Actividad-Modulo3-leccion2