

**ACTIVITAT AVALUABLE AC4**
**Mòdul:** MP02- Bases de dades

**UF:** UF3: Llenguatges SQL: DCL i extensió procedimental

**Professor:** Judith López

**Data límit d'entrega:** Consultar al ClickEdu

**Mètode d'entrega:** Per mitjà del Clickedu de l'assignatura. Les activitats entregades més enllà de la data límit només podran obtenir una nota de 5.

**Instruccions:**

**Les tasques s'han d'entregar totes en un únic document PDF.**

**Resultats de l'aprenentatge:**

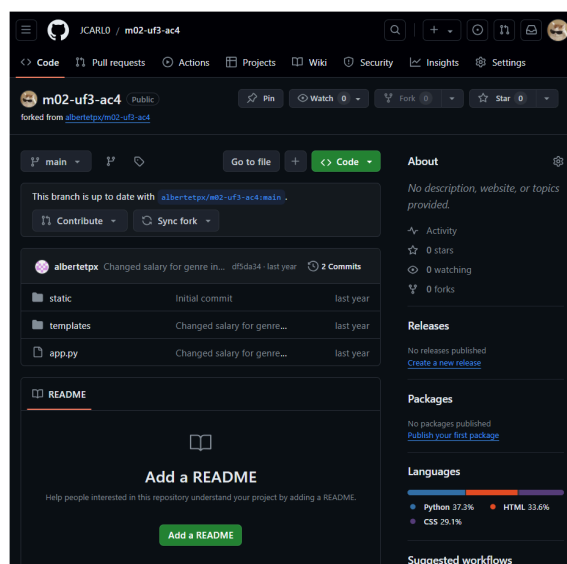
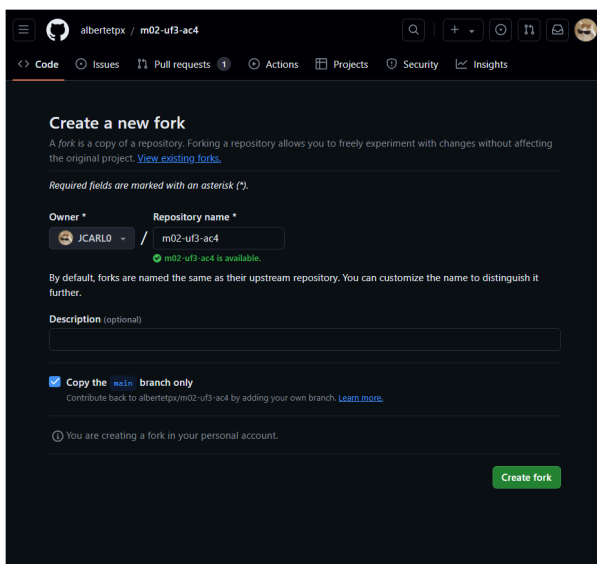
RA1. Implanta mètodes de control d'accés utilitzant assistents, eines gràfiques i comandes del llenguatge del sistema gestor de bases de dades corporatiu.

RA2. Desenvolupa procediments emmagatzemats avaluant i utilitzant les sentències del llenguatge incorporat en el sistema gestor de bases de dades corporatiu.

**És indispensable documentar correctament totes les passes de l'exercici, amb captures de pantalla, segons convingui.**

**Tasca 1 (4 punts) Anàlisi d'una situació d'injecció de codi:**

a) Fes un fork del següent repositori al teu compte de GitHub. A continuació, clona'l a la teva màquina local.

<https://github.com/albertetpx/m02-uf3-ac4.git>


```
C:\Users\Jcarl\Documents\1r Desenvolupament d'aplicacions web\MP2. Bases de dades\M02UF3\AC6>git clone https://github.com/JCARLO/m02-uf3-ac4
Cloning into 'm02-uf3-ac4'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 14 (delta 4), reused 14 (delta 4), pack-reused 0
Receiving objects: 100% (14/14), done.
Resolving deltas: 100% (4/4), done.
```

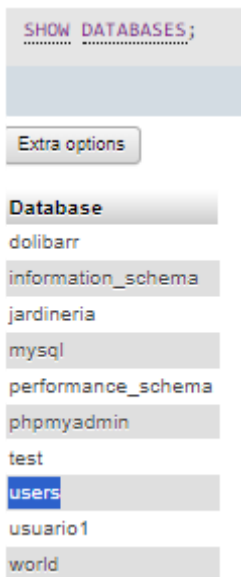
b) Fes les següents operacions per poder desplegar l'aplicació web que s'adjunta (*formulario.rar*).

-A L'APLICACIÓ FLASK: canvia els paràmetres de connexió a la base de dades (en concret, l'usuari i la contrassenya) perquè pugui connectar al teu servidor MySQL.

```
# connectBD: conecta a la base de datos users en MySQL
def connectBD():
    db = mysql.connector.connect(
        host = "localhost",
        user = "root",
        passwd = "",
        database = "users"
    )
    return db
```

-AL SERVIDOR MYSQL: crea la base de dades *users*. No cal que creis cap taula; serà creada per la pròpia aplicació web (observa la funció

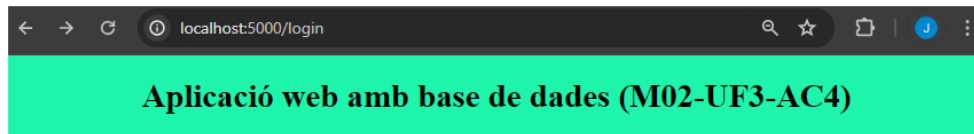
```
CREATE DATABASE IF NOT EXISTS users;
```



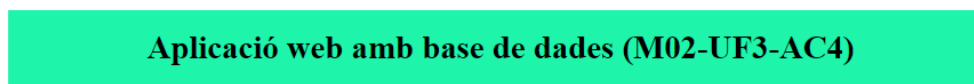
c) Executa l'aplicació (*app.py*) i comprova que arrenca sense errors. Obre el navegador a <http://localhost:5000>, i comprova que:



-L'usuari *user01* amb contrassenya *admin* pot fer **login correcte** i consultar les seves dades.



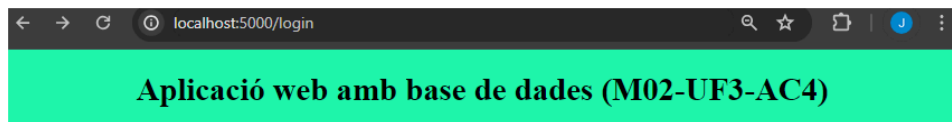
Log in to application

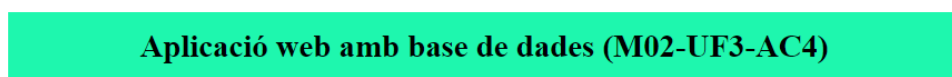
LOGIN CORRECTO

User	Name	Surname 1	Surname 2	Age	Genre
user01	Ramón	Sigüenza	López	35	H

-L'usuari *user01* amb contrassenya *1234* fa un **login incorrecte**.

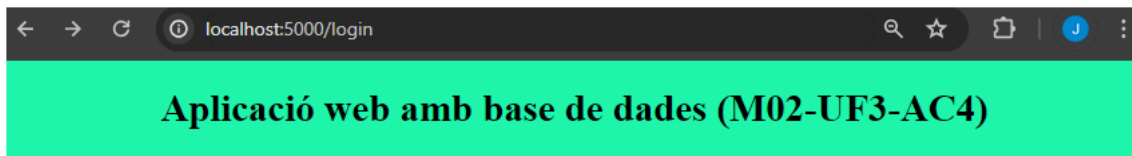


Log in to application

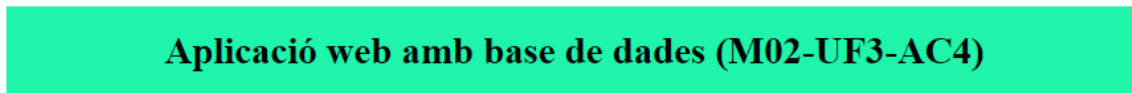
  
  


LOGIN INCORRECTO

d) Prova a autenticar l'usuari *user01* i la contrassenya '*OR 1=1;*' (valor exacte).



### Log in to application

LOGIN CORRECTO

User	Name	Surname 1	Surname 2	Age	Genre
user01	Ramón	Sigüenza	López	35	H

e) Explica què ocorre, i per què estem davant d'una situació d'injecció de codi.

f) Reimplementa la funció *checkUser* perquè faci servir una sentència parametritzada que eviti la situació d'injecció de codi:

Per exemple:

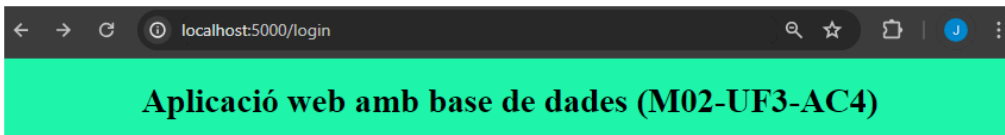
```
query = """Update employee set Salary = %s where id = %s"""  
values = (8000, 5)  
cursor.execute(query, values)
```

```
# checkUser: comprueba si el par usuario-contraseña existe en la BD  
def checkUser(user,password):  
    bd=connectBD()  
    cursor=bd.cursor()  
  
    query = "SELECT user, name, surname1, surname2, age, genre FROM users WHERE user= %s AND password= %s"  
    print(query)  
    cursor.execute(query)  
    userData = cursor.fetchall()  
    bd.close()  
  
    if not userData == []:  
        return False  
    else:  
        return userData[0]
```

Aplicació web amb base de dades (M02-UF3-AC4)

LOGIN INCORRECTO

g) Comprova que, ara, el formulari de login ja no és vulnerable a la injecció de codi.



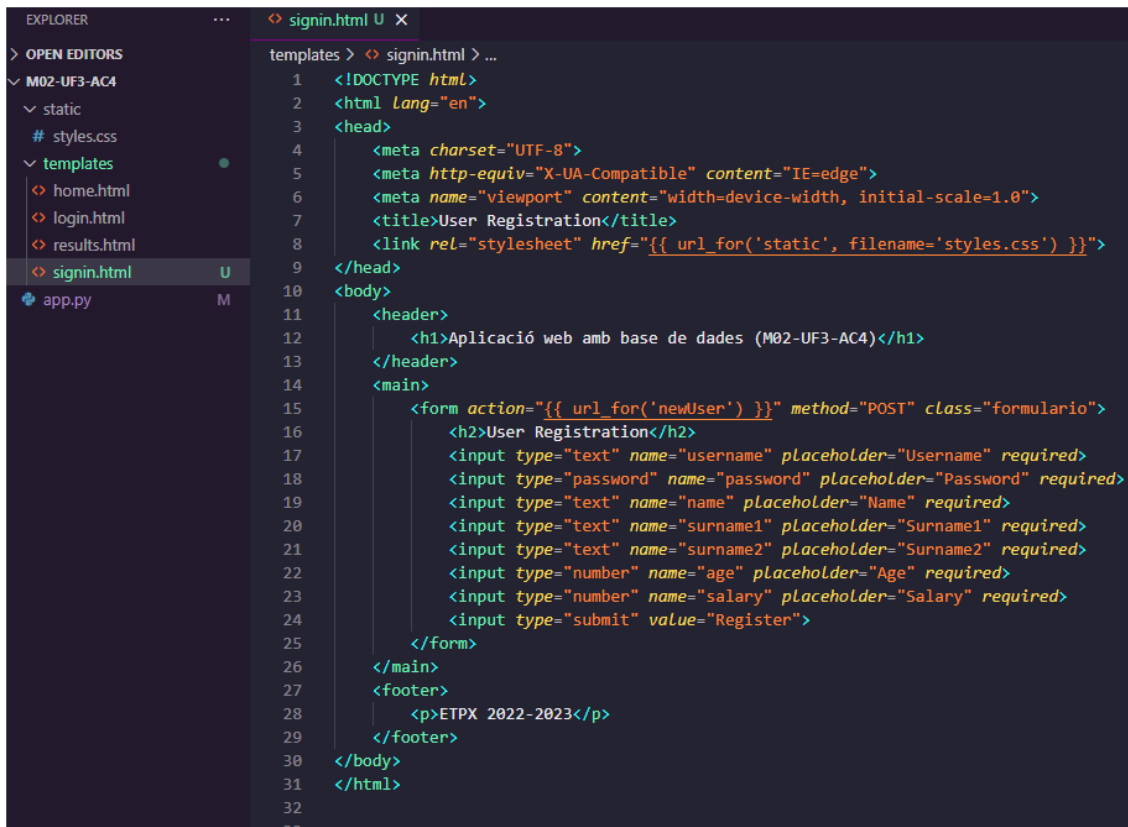
Log in to application

<input type="text" value="user01"/>
<input type="text" value="OR 1=1;"/>
<input type="button" value="Enviar"/>

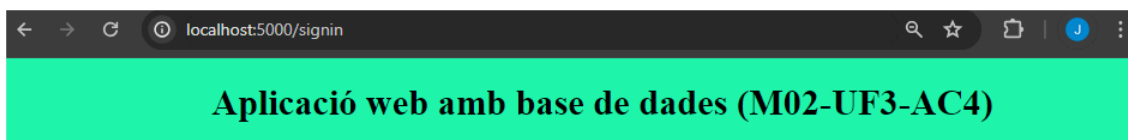
h) Explica per què la instrucció parametritzada resol la vulnerabilitat.

Tasca 2 (6 punts). Completa l'aplicació web amb la funcionalitat de poder crear nous usuaris:

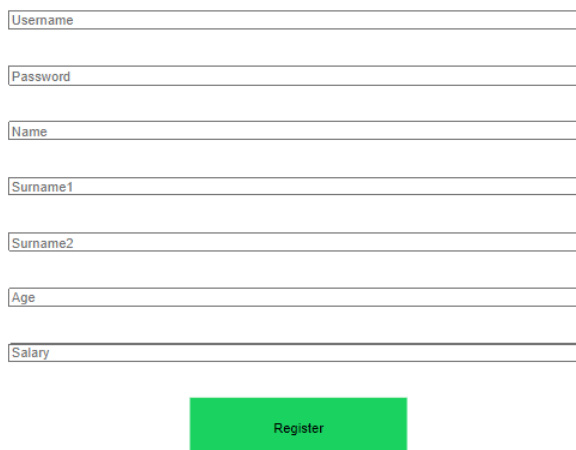
a) Crea una altra plantilla (*signin.html*), seguint l'estructura de *login.html*. Aquesta pàgina haurà de contenir un formulari de registre d'usuari, en que es pugui donar d'alta un usuari amb: nom d'usuari, contrassenya, nom, cognom1, cognom2, edat i salari.



```
1 <!DOCTYPE html>
2 <html Lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>User Registration</title>
8   <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
9 </head>
10 <body>
11   <header>
12     <h1>Aplicació web amb base de dades (M02-UF3-AC4)</h1>
13   </header>
14   <main>
15     <form action="{{ url_for('newUser') }}" method="POST" class="formulario">
16       <h2>User Registration</h2>
17       <input type="text" name="username" placeholder="Username" required>
18       <input type="password" name="password" placeholder="Password" required>
19       <input type="text" name="name" placeholder="Name" required>
20       <input type="text" name="surname1" placeholder="Surname1" required>
21       <input type="text" name="surname2" placeholder="Surname2" required>
22       <input type="number" name="age" placeholder="Age" required>
23       <input type="number" name="salary" placeholder="Salary" required>
24       <input type="submit" value="Register">
25     </form>
26   </main>
27   <footer>
28     <p>ETPX 2022-2023</p>
29   </footer>
30 </body>
31 </html>
```



## User Registration



Username
Password
Name
Surname1
Surname2
Age
Salary
Register

b) Modifica la ruta `"/signin"` a l'aplicació flask per a que mostri la plantilla `signin.html` que acabes de crear.

```
@app.route("/signin")
def signin():
    return render_template("signin.html")
```

c) Crea una nova ruta (`"/newUser"`) a l'aplicació flask per a rebre i processar les dades del formulari de registre. T'hauràs d'inspirar en la ruta `"/results"` ja existent. Des d'aquesta ruta, crida la funció `createUser`.

```
@app.route("/newUser", methods=['POST'])
def newUser():
    if request.method == 'POST':
        formData = request.form
        username = formData['username']
        password = formData['password']
        name = formData['name']
        surname1 = formData['surname1']
        surname2 = formData['surname2']
        age = int(formData['age'])
        salary = float(formData['salary'])
        print(formData)

        createUser(username, password, name, surname1, surname2, age, salary)

    return "User created successfully"
```

d) Associa l'acció del formulari de registre (atribut `action`) a la nova ruta que acabes de crear. Observa com es fa en el formulari de login.

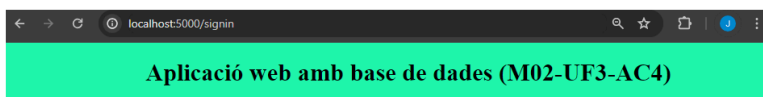
```
<form action="{{ url_for('newUser') }}" method="POST" class="formulario">
```

d) Implementa la funció `createUser` perquè s'escriguin les dades del nou usuari en la base de dades. Utilitza sentències parametritzades.

```
createUser(username, password, name, surname1, surname2, age, salary)
```

e) Comprova el correcte funcionament de l'aplicació

Registrando con nuevos datos-



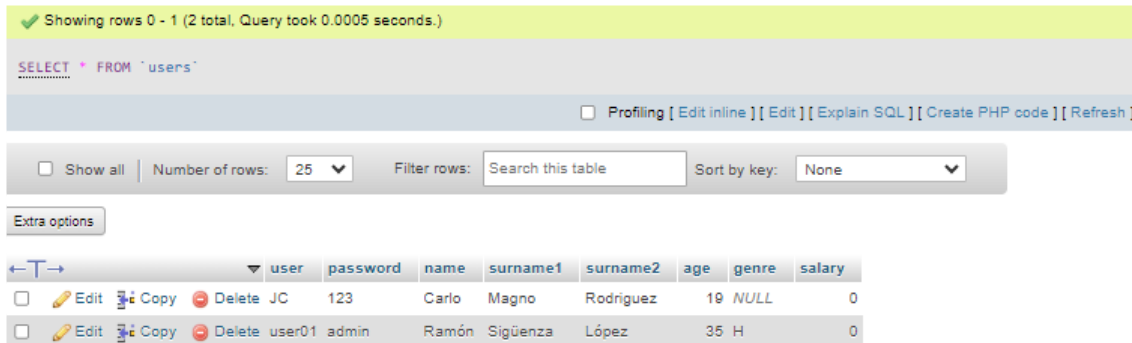
User Registration

<input type="text" value="JC"/>
<input type="password" value="****"/>
<input type="text" value="Carlo"/>
<input type="text" value="Magno"/>
<input type="text" value="Rodriguez"/>
<input type="text" value="19"/>
<input type="text" value="0"/>

Comprobación que se ha creado-



Comprobación en la base de datos-



A screenshot of a database query tool interface. At the top, a green status bar says 'Showing rows 0 - 1 (2 total, Query took 0.0005 seconds.)'. Below it, the SQL query 'SELECT \* FROM `users`' is entered. A toolbar contains options like 'Profiling', 'Edit inline', 'Edit', 'Explain SQL', 'Create PHP code', and 'Refresh'. Below the toolbar, there are controls for 'Show all', 'Number of rows' (set to 25), 'Filter rows' (a search box), and 'Sort by key' (set to None). An 'Extra options' button is also present. The main area displays a table with two rows of user data.

	user	password	name	surname1	surname2	age	genre	salary
<input type="checkbox"/>	JC	123	Carlo	Magno	Rodriguez	19	NULL	0
<input type="checkbox"/>	user01	admin	Ramón	Sigüenza	López	35	H	0

f) Puja el codi complet de l'aplicació a un repositori del teu compte de GitHub i inclou l'enllaç en el PDF que entreguis.

<https://github.com/JCARLO/m02-uf3-ac4>