

LAPORAN

PEMROGRAMAN WEB PRAKTIK

Dosen Pengampu:
Aditya Dimas Dewanto, S.T., M.T.



Dibuat oleh:

Nama : Muhammad Zaki As Shidiqi
NPM : 5240411230

PROGRAM STUDI INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS TEKNOLOGI YOGYAKARTA
YOGYAKARTA
2025

DAFTAR ISI

DAFTAR ISI.....	2
BAB I MENJALANKAN MYSQL DAN INTEGRASI DENGAN FLASK.....	3
1.1 MySQL.....	3
1.2 Integrasi MySQL dengan Flask.....	5
BAB II PENERAPAN SESSION PADA FLASK	9
2.1 Sistem Login pada Flask.....	9
2.2 Tugas	13
KESIMPULAN.....	20

BAB I

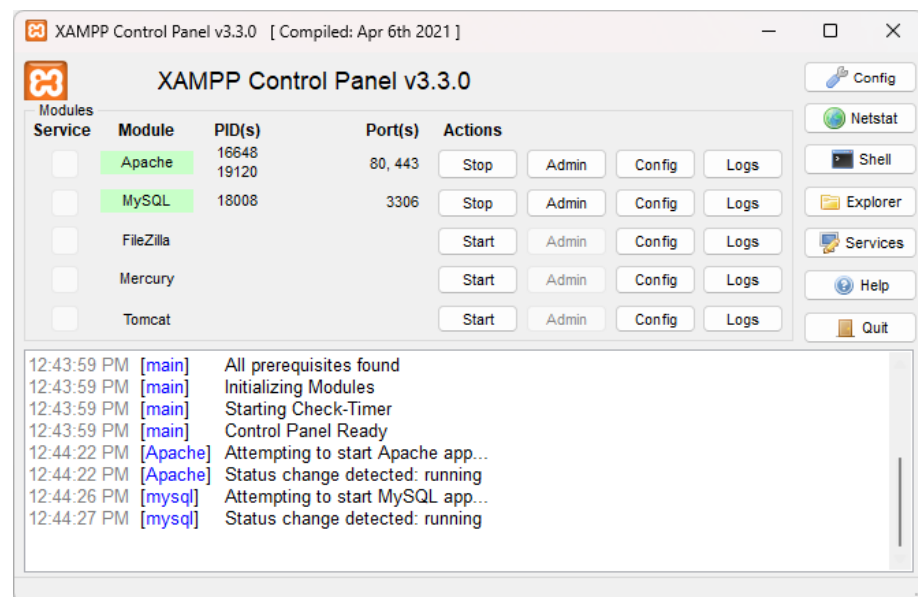
MENJALANKAN MYSQL DAN INTEGRASI DENGAN FLASK

1.1 MySQL

MySQL adalah sebuah sistem manajemen *database* relasional (RDBMS) *opensource* yang berfungsi untuk menyimpan, mengelola, dan memproses data. Untuk menjalankan MySQL pada komputer, kita membutuhkan satu tools yaitu XAMPP. XAMPP adalah sebuah software yang mengintegrasikan server web Apache dan MySQL. Fungsinya untuk menjalankan MySQL di server lokal pada komputer.

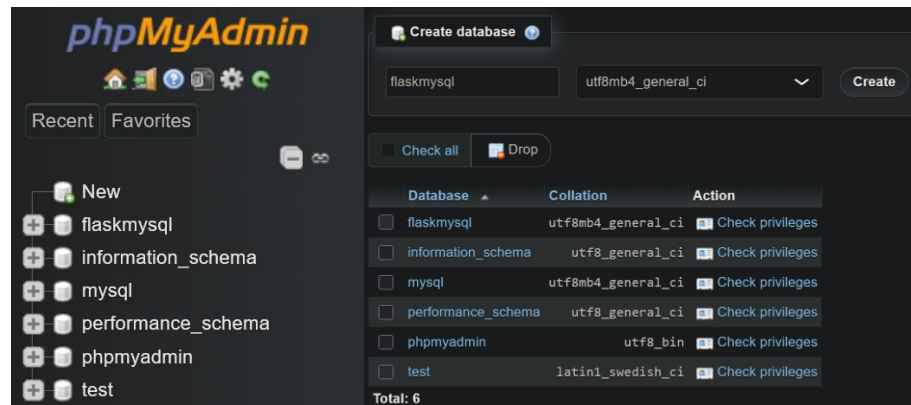
a. Menjalankan MySQL.

Jalankan *software* XAMPP pada komputer. Setelah terbuka, klik tombol start pada modul Apache. Setelah Apache sudah berjalan, kemudian klik tombol start pada modul MySQL. Kemudian klik tombol admin pada modul MySQL untuk membuka panel phpMyAdmin.



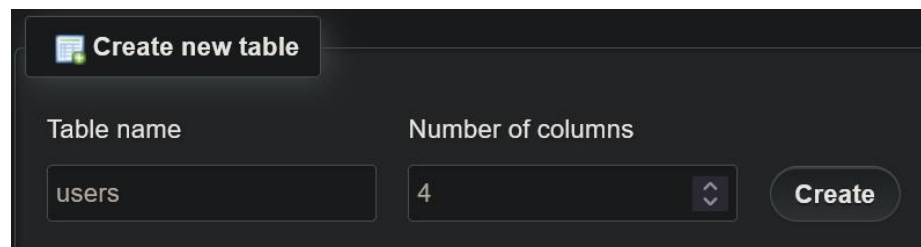
b. Membuat database

Setelah phpMyAdmin terbuka, kemudian klik tombol *New* untuk membuat *database* baru. Setelah di klik pada bagian kanan akan muncul menu untuk menambahkan *database*. Karena pada materi kali ini tentang integrasi Flask dengan MySQL nama untuk *database*-nya "flaskmysql". Setelah memberikan nama klik tombol *create* untuk membuat *database*-nya.

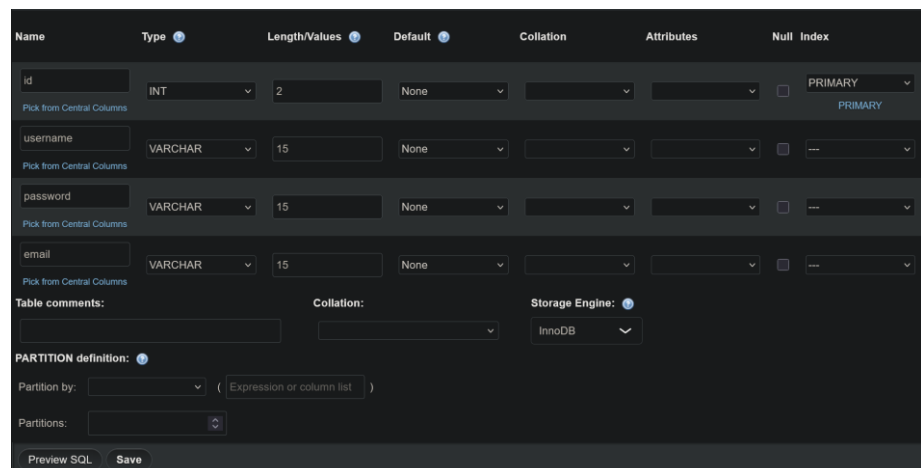


c. Membuat tabel users.

Setelah *database* berhasil dibuat, akan muncul menu untuk menambahkan tabel pada *database*. Pada kolom *Table name* masukan “users” karena kita akan membuat tabel yang bernama “users” yang akan berisi banyak data pengguna nantinya. Pada kolom *Number of columns* masukan 4 karena tabel ini akan memiliki 4 kolom. Kemudian klik tombol *create* untuk membuat tabel tersebut.



d. Membuat kolom.



e. Mengisi tabel.

Column	Type	Function	Null	Value
id	int(15)			0
username	varchar(15)			admin
password	varchar(15)			admin123
email	varchar(20)			admin@xyz.com

Go

1.2 Integrasi MySQL dengan Flask

Flask adalah sebuah micro framework yang memungkinkan kita memungkinkannya kita membuat webserver menggunakan python. Prinsip kerjanya dengan me-*return* html pada setiap function yang kita berikan route diatasnya.

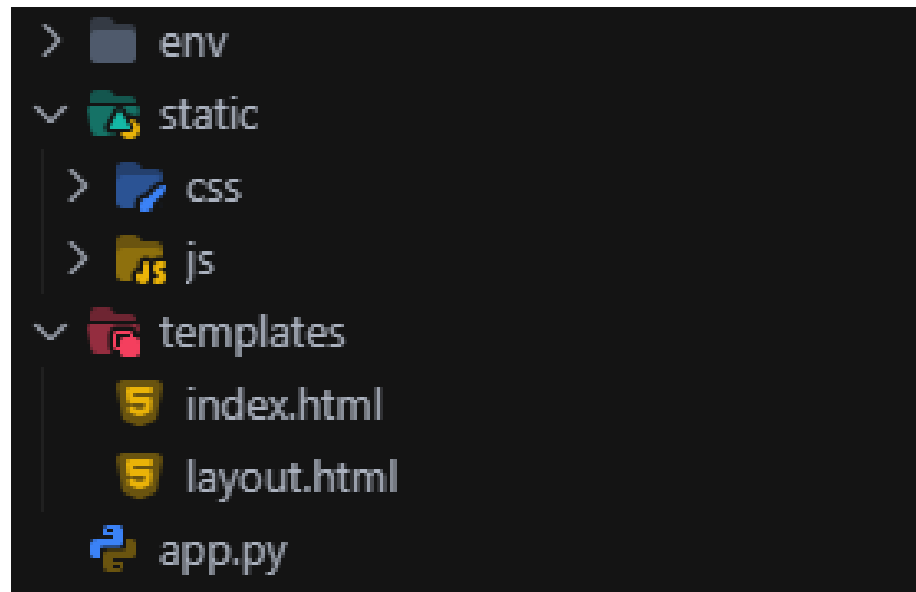
a. Perintah pada CLI.

Sebelum menginstal *flask-mysqldb* pastikan sudah membuat environment menggunakan *virtualenv* dan sudah mengaktifkannya. Selanjutnya untuk menginstall *flask-mysqldb* kita dapat mengetikan command berikut pada terminal dan tunggu proses instalasi sampai selesai.

```
pip install flask-mysqldb
```

b. Membuat sturktur folder.

Jika proses penginstalan sudah selesai, maka kita bisa memulai proses setup proyek kita. Kita bisa memulai dengan membuat struktur folder terlebih dahulu. Berikut adalah struktur folder pada proyek flask ini.



c. Menghubungkan Flask dengan MySQL

Selanjutnya kita mulai menuliskan kode pada file `app.py`. Pada awal baris kita harus meng-*import* beberapa *library* yaitu Flask, `render_template`, dan MySQL. Untuk membuat aplikasi web menggunakan Flask kita perlu menuliskan sintaks “`app = Flask(__name__)`”. Kemudian menuliskan sintaks untuk menghubungkan MySQL dengan aplikasi Flask. Untuk mendapatkan data *users*, kita lakukan query ke database. Selain itu, pada *return* dari *route home* selain “`index.html`” ada *users* yang diambil dari hasil *query* ke database.

```
1 from flask import Flask, render_template
2 from flask_mysql import MySQL
3
4 app = Flask(__name__)
5
6 app.config['MYSQL_HOST'] = 'localhost'
7 app.config['MYSQL_USER'] = 'root'
8 app.config['MYSQL_PASSWORD'] = ''
9 app.config['MYSQL_DB'] = 'flaskmysql'
10
11 mysql = MySQL(app)
12
13 @app.route('/')
14 def home():
15     cur = mysql.connection.cursor()
16     cur.execute('SELECT * FROM users')
17     data = cur.fetchall()
18     cur.close()
19     return render_template('index.html', users=data)
20
21 if __name__ == '__main__':
22     app.run(debug=True)
```

d. Membuat layout untuk template atau acuan halaman lainnya.

Setelah selesai menuliskan kode pada file `app.py` kita lanjutkan menuliskan kode html pada file `layout.html` yang berada di folder `templates`. File ini akan menjadi layout utama untuk halaman lainnya. Kode file ini berupa tag paling luar dari html yaitu `<html>`, `<head>`, dan `<body>`. Selain ada ketiga tak tersebut, terdapat 2 block yaitu head dan body, kedua block ini yang nantinya adalah bagian isi dari website kita.

```

1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6     <link
7       rel="stylesheet"
8       href="{{ url_for('static', filename='css/bootstrap.min.css') }}"
9     />
10    {% block head%} {% endblock %}
11  </head>
12  <body>
13    {% block body%} {% endblock %}
14    <script src="{{ url_for('static', filename='js/bootstrap.bundle.min.js') }}"></script>
15  </body>
16 </html>

```

- e. Membuat isi untuk halaman yang menampilkan tabel *users*.

Pada halaman ini kita menggunakan tag `<table>` untuk membuat sebuah tabel, `<thead>` untuk membuat header dari tabel, `<tbody>` untuk membuat isi dari tabel, `<tr>` untuk membuat baris. Untuk menampilkan data dari tabel *users* kita menggunakan *for loop* supaya setiap data *users* akan membuat baris baru pada tabel.

```

1 {% extends 'layout.html' %} {% block head %}
2 <title>Koneksi Flask dan MySQL</title>
3 {% endblock %} {% block body %}
4 <div class="container mt-3">
5   <h1 class="mb-3">Flask dan MySQL</h1>
6   <div class="mb-3">
7     <table class="table table-striped">
8       <thead>
9         <tr>
10          <th>ID</th>
11          <th>Username</th>
12          <th>Password</th>
13          <th>Email</th>
14        </tr>
15      </thead>
16      <tbody>
17        {% for row in users %}
18          <tr>
19            <th scope="row">{{ row.0 }}</th>
20            <td>{{ row.1 }}</td>
21            <td>{{ row.2 }}</td>
22            <td>{{ row.3 }}</td>
23          </tr>
24        {% endfor %}
25      </tbody>
26    </table>
27  </div>
28 {% endblock %}
29 </div>

```

f. Hasil

Flask dan MySQL

ID	Username	Password	Email
1	adhitsa	adhitsa123	adhitsa@xyz.com
2	ali	ali123	ali@xyz.com
3	zaki	zaki123	zaki@xyz.com

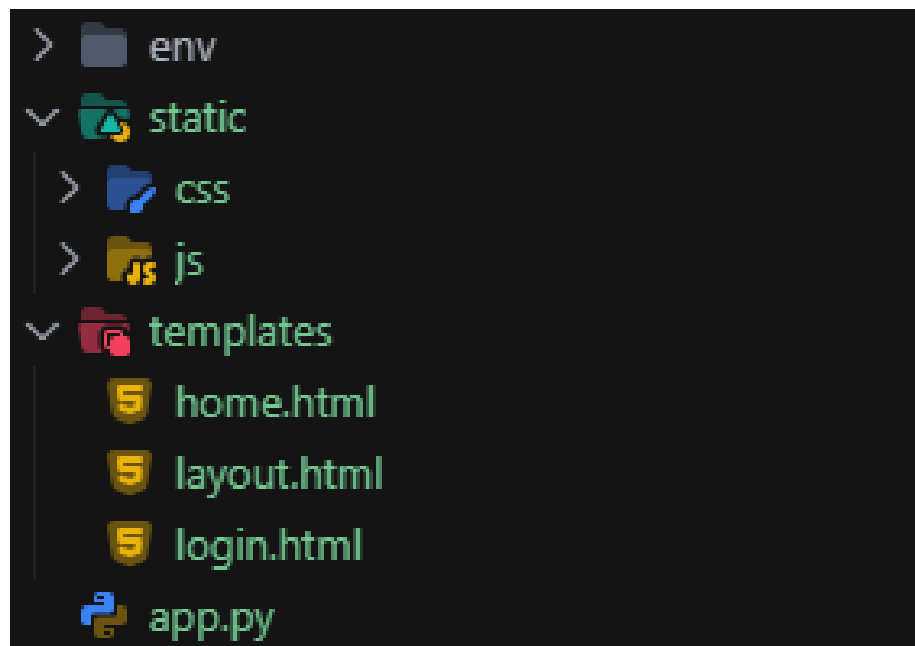
BAB II

PENERAPAN SESSION PADA FLASK

2.1 Sistem Login pada Flask

Sistem login adalah sebuah sistem yang mengharuskan *client* untuk melakukan validasi data sebelum mengakses halaman atau fitur lain pada sebuah aplikasi. Dengan sistem ini akan membuat aplikasi yang kita buat lebih aman dari *spam request* yang biasanya dilakukan oleh bot.

a. Struktur folder



b. Import library yang dibutuhkan

Pada awal baris di file `app.py` *import library* yang dibutuhkan, berbeda dengan sebelumnya untuk membuat sistem login pada aplikasi Flask membutuhkan beberapa tambahan library seperti `session`, `request`, `redirect`, dan `url_for`.



c. Membuat *secret key* dan menghubungkan Flask dengan MySQL

Untuk menerapkan *session* pada aplikasi Flask, kita perlu membuat *secret key*. Ini nantinya akan tersimpan pada browser pengguna dan akan hilang dalam rentang tertentu.

```

1 app = Flask(__name__)
2
3 app.secret_key = 'HSAIJWEHRIUHF92165009DIUFGIFSG989234R440837047IDSAFF892'
4
5 app.config['MYSQL_HOST'] = 'localhost'
6 app.config['MYSQL_USER'] = 'root'
7 app.config['MYSQL_PASSWORD'] = ''
8 app.config['MYSQL_DB'] = 'flaskmysql'
9
10 mysql = MySQL(app)

```

d. Route login

Login *route* ini akan terletak pada *path* root, supaya dapat menangkap isi form yang dikirimkan perlu memberikan izin penggunaan *method* “POST”. Untuk menangkap apa yang dikirimkan dari form kita menggunakan *library request* yang sudah kita *import* sebelumnya. Sebelum melakukan query ke *database* kita lakukan validasi apakah form kosong atau tidak. Jika tidak selanjutnya akan menampung isi dari field email ke dalam variabel email dan *password* ke dalam variabel passwd. Kemudian dilakukan query ke *database* untuk mengambil data pengguna dengan email dan password yang cocok, jika ada maka akan diberikan session dan diarahkan ke halaman home. tetapi jika tidak ada akan di kembalikan lagi ke halaman login.

```

1 @app.route('/', methods=['GET', 'POST'])
2 def login():
3     if request.method == 'POST' and 'inpEmail' in request.form and 'inpPass' in request.form:
4         email = request.form['inpEmail']
5         passwd = request.form['inpPass']
6
7         cur = mysql.connection.cursor()
8         cur.execute('SELECT * FROM users WHERE email = %s AND password = %s', (email, passwd))
9
10        data = cur.fetchone()
11
12        if data:
13            session['is_logged_in'] = True
14            session['username'] = data[1]
15
16            return redirect(url_for('home'))
17
18        else:
19            return render_template('login.html', error='Invalid email or password')
20
21    else:
22        return render_template('login.html')

```

e. Route home

Home route ini akan terletak pada *path* “/home”. Ketika pengguna mengakses halaman home, akan dilakukan validasi apakah memiliki session atau tidak. Jika *client* memiliki session maka akan dilakukan query untuk mengambil semua data *user* ke database lalu menampung hasilnya ke variabel data, baru kemudian me-return home.html dan variabel users yang mengambil dari data. Jika *client* tidak memiliki *session* maka akan diarahkan ke halaman login.

```

1 @app.route('/home')
2 def home():
3     if 'is_logged_in' in session:
4         cur = mysql.connection.cursor()
5         cur.execute(''SELECT * FROM users'')
6
7         data = cur.fetchall()
8
9         return render_template('home.html', users=data)
10    else:
11        return redirect(url_for('login'))

```

f. Route logout

Logout route ini akan terletak pada *path* “/logout”. Pada *route* ini akan dilakukan penghapusan *session* dan akan mengarahkan *client* ke halaman login.

```

1 @app.route('/logout')
2 def logout():
3     session.pop('is_logged_in', None)
4     session.pop('username', None)
5
6     return redirect(url_for('login'))

```

g. Halaman login

Ini adalah isi dari login.html. Pada halaman ini terdapat form login dengan method “POST” yang memiliki dua field input yaitu email dan password. Selain itu, ada juga tombol untuk submit form.

```

1 {% extends 'layout.html' %}
2
3 {% block head %}
4 <title>Login Page</title>
5 {% endblock %}
6
7 {% block body %}
8 <div class="mt-5">
9     <div class="row justify-content-center">
10         <div class="col-4">
11             <div class="card">
12                 <div class="card-header">
13                     <h2 class="text-center">Login</h2>
14                 </div>
15                 <div class="card-body">
16                     <form method="POST">
17                         <div>
18                             <label class="form-label">Email Address</label>
19                             <input type="email" name="inpEmail" class="form-control" required>
20                         </div>
21                         <div>
22                             <label class="form-label">Password</label>
23                             <input type="password" name="inpPass" class="form-control" required>
24                         </div>
25                         <div class="d-grid">
26                             <button type="submit" class="btn btn-primary mt-4">Login</button>
27                         </div>
28                     </form>
29                 </div>
30             </div>
31         </div>
32     </div>
33 </div>
34 {% endblock %}

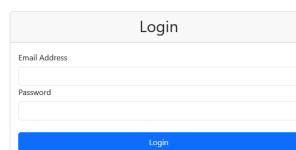
```

h. Halaman home

Pada halaman ini kita menggunakan tag `<table>` untuk membuat sebuah tabel, `<thead>` untuk membuat header dari tabel, `<tbody>` untuk membuat isi dari tabel, `<tr>` untuk membuat baris. Untuk menampilkan data dari tabel *users* kita menggunakan *for loop* supaya setiap data *users* akan membuat baris baru pada tabel.

```
1 {% extends 'layout.html' %} {% block head %}
2 <title>Home</title>
3 {% endblock %} {% block body %}
4 <div class="mt-5 container">
5 <h1 class="mb-3">Home</h1>
6 <div class="mb-3"></div>
7 <a href="/logout" class="btn btn-danger">Logout</a>
8 <table class="table table-striped">
9   <thead>
10     <tr>
11       <th>ID</th>
12       <th>Username</th>
13       <th>Password</th>
14       <th>Email</th>
15     </tr>
16   </thead>
17   <tbody>
18     {% for row in users %}
19     <tr>
20       <th scope="row">{{ row.0 }}</th>
21       <td>{{ row.1 }}</td>
22       <td>{{ row.2 }}</td>
23       <td>{{ row.3 }}</td>
24     </tr>
25     {% endfor %}
26   </tbody>
27 </table>
28 </div>
29 {% endblock %}
```

i. Hasil



Login
Email Address
Password
Login

Home

Logout			
ID	Username	Password	Email
1	adhitia	adhitia123	adhitia@xyz.com
2	ali	ali123	ali@xyz.com
3	zaki	zaki123	zaki@xyz.com

2.2 Tugas

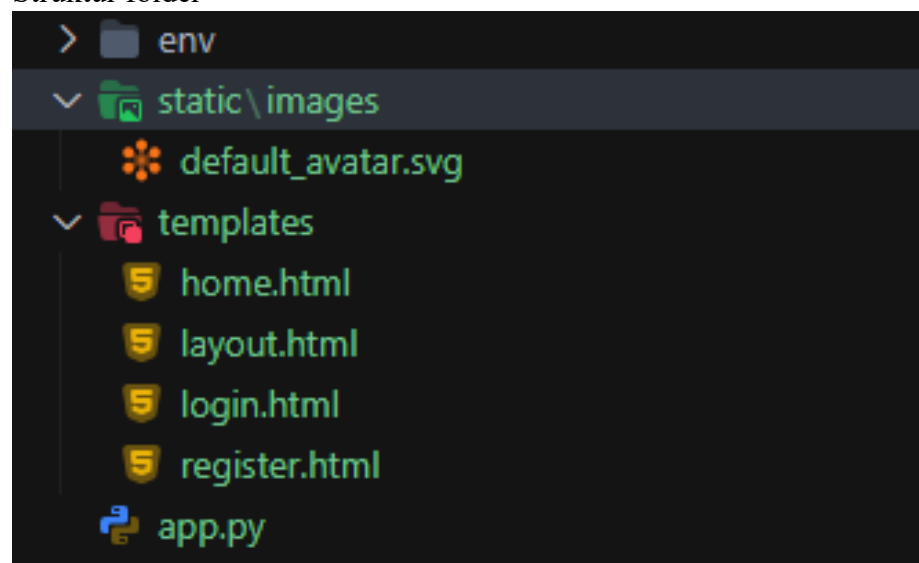
Saya tidak lagi menggunakan Bootstrap untuk mengerjakan tugas ini, saya menggunakan TailwindCSS karena lebih *customizable*.

- a. Menambahkan rule *auto increment* pada id

Sebelum menambahkan fitur *register* perlu untuk mengubah kolom id menjadi *auto increment* supaya id ter-generate otomatis saat ada data baru yang masuk. Selain itu, untuk kolom username dan email diatur *unique* supaya tidak terdapat duplikasi data.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
<input type="checkbox"/>	1 id 🔑	int(2)			No	None		AUTO_INCREMENT
<input type="checkbox"/>	2 username 🔑	varchar(15)	utf8mb4_general_ci		No	None		
<input type="checkbox"/>	3 password	varchar(15)	utf8mb4_general_ci		No	None		
<input type="checkbox"/>	4 email 🔑	varchar(20)	utf8mb4_general_ci		No	None		

- b. Struktur folder



c. Route login

Saya mengubah path dari awalnya “/” menjadi “/login” supaya path lebih sesuai dengan isi halaman. Saya juga menambahkan validasi yang berfungsi untuk mengarahkan *client* yang sudah memiliki session, ketika mengakses path /login maka akan di arahkan ke path “/” atau halaman home. Saya juga menambahkan informasi email dan password tidak valid ke *client* jika email dan password yang diinputkan tidak ada pada database. Saya juga menambahkan username dan email ke dalam session supaya bisa menampilkan data *user* yang sedang digunakan.

```
1 @app.route('/login', methods=['GET', 'POST'])
2 def login():
3     if request.method == 'POST' and 'inpEmail' in request.form and 'inpPass' in request.form:
4         email = request.form['inpEmail']
5         passwd = request.form['inpPass']
6
7         cur = mysql.connection.cursor()
8         cur.execute('SELECT * FROM users WHERE email = %s AND password = %s', (email, passwd))
9
10        data = cur.fetchone()
11
12        if data:
13            session['is_logged_in'] = True
14            session['username'] = data[1]
15            session['email'] = data[3]
16
17            return redirect(url_for('home'))
18
19        else:
20            return render_template('login.html', error='Invalid email or password')
21
22    elif 'is_logged_in' in session:
23        return redirect(url_for('home'))
24
25    else:
26        return render_template('login.html')
```

d. Route root

Saya juga mengubah path yang awalnya “/home” menjadi “/”. Saya juga membuat variabel user yang akan memuat username dan email yang didapatkan dari session ke dalam sebuah list.

```
1 @app.route('/')
2 def home():
3     if 'is_logged_in' in session:
4         user = [session['username'], session['email']]
5         cur = mysql.connection.cursor()
6         cur.execute('SELECT * FROM users')
7
8         data = cur.fetchall()
9
10        return render_template('home.html', users=data, user_data=user)
11    else:
12        return redirect(url_for('login'))
```

e. Route logout

Saya menambahkan validasi supaya dilakukan penghapusan session jika *client* sudah login saja.

```

1 @app.route('/logout')
2 def logout():
3     if 'is_logged_in' in session:
4         session.pop('is_logged_in', None)
5         session.pop('username', None)
6         session.pop('email', None)
7
8     return redirect(url_for('login'))

```

f. Route Register

Route ini berfungsi untuk meng-*handle* request client untuk melakukan registrasi pada *path* “/register”. Karena pada route ini mengambil data dari form, maka perlu untuk mengizinkan *method* “POST”. Pada route ini juga terdapat validasi untuk memastikan semua kolom input tidak kosong. Jika semua kolom input sudah di validasi maka semua isi dari input *client* akan ditampung ke masing-masing variabel yaitu username, email, passwd. Kemudian akan dilakukan *query* ke *database* untuk memasukan data ke *database*, jika berhasil maka *client* akan diarahkan ke halaman login. Tidak lupa, di sini juga terdapat validasi supaya mengarahkan client yang sudah login ke path “/” jika mengakses *path* ini.

```

1 @app.route('/register', methods=['GET', 'POST'])
2 def register():
3     if request.method == 'POST' and 'inpUsername' in request.form and 'inpEmail' in request.form
4       and 'inpPass' in request.form:
5         username = request.form['inpUsername']
6         email = request.form['inpEmail']
7         passwd = request.form['inpPass']
8
9         cur = mysql.connection.cursor()
10        cur.execute('INSERT INTO users (username, email, password) VALUES (%s, %s, %s)',
11                    (username, email, passwd))
12        mysql.connection.commit()
13        cur.close()
14
15        return redirect(url_for('login'))
16
17    elif 'is_logged_in' in session:
18        return redirect(url_for('home'))
19
20    else:
21        return render_template('register.html')

```

g. Halaman login

Saya menabahkan informasi jika email dan password tidak valid, ini akan muncul setelah *client* melakukan submit.

```
1 {% extends 'layout.html' %} {% block head %}
2 <title>Login Page</title>
3 {% endblock %} {% block body %}
4 <div class="flex container mx-auto justify-center items-center h-screen">
5   <div class="bg-white p-8 rounded-md shadow-md space-y-5">
6     <h1 class="text-3xl text-zinc-700 font-bold text-center">Login</h1>
7     <form method="POST" class="space-y-5">
8       <div class="relative">
9         <input
10           type="email"
11           id="email"
12           class="block px-2.5 pb-2.5 pt-4 w-full text-sm text-gray-900 bg-transparent rounded-lg
border-1 border-gray-300 appearance-none focus:outline-none focus:ring-0 focus:border-zinc-700
peer"
13           placeholder=" "
14           name="inpEmail"
15         />
16         <label
17           for="email"
18           class="absolute text-sm text-gray-500 transition-all duration-300 transform -translate-
y-4 scale-75 top-2 z-10 origin-[0] bg-white px-2 peer-focus:px-2 peer-focus:text-zinc-700 peer-
placeholder-shown:scale-100 peer-placeholder-shown:-translate-y-1/2 peer-placeholder-shown:top-1/2
peer-focus:top-2 peer-focus:scale-75 peer-focus:-translate-y-4 rtl:peer-focus:translate-x-1/4
rtl:peer-focus:left-auto start-1 hover:cursor-text"
19         >Email</label>
20       >
21     </div>
22     <div class="relative">
23       <input
24         type="password"
25         id="password"
26         class="block px-2.5 pb-2.5 pt-4 w-full text-sm text-gray-900 bg-transparent rounded-lg
border-1 border-gray-300 appearance-none focus:outline-none focus:ring-0 focus:border-zinc-700
peer"
27         placeholder=" "
28         name="inpPass"
29       />
30       <label
31         for="password"
32         class="absolute text-sm text-gray-500 transition-all duration-300 transform -translate-
y-4 scale-75 top-2 z-10 origin-[0] bg-white px-2 peer-focus:px-2 peer-focus:text-zinc-700 peer-
placeholder-shown:scale-100 peer-placeholder-shown:-translate-y-1/2 peer-placeholder-shown:top-1/2
peer-focus:top-2 peer-focus:scale-75 peer-focus:-translate-y-4 rtl:peer-focus:translate-x-1/4
rtl:peer-focus:left-auto start-1 hover:cursor-text"
33       >Password</label>
34     >
35   </div>
36   <div class="text-center">
37     {% if error %}
38     <span class="text-red-500 text-sm">{{ error }}</span>
39     {% endif %}
40   </div>
41   <div class="text-center">
42     <button
43       type="submit"
44       class="w-full bg-zinc-700 text-white p-2 rounded-md hover:bg-zinc-600 hover:cursor-
pointer transition-all duration-300"
45     >
46       Login
47     </button>
48     <span class="text-gray-400 text-sm">
49       >Don't have an account?
50       <a href="/register" class="text-zinc-700">Register</a></span>
51   >
52 </div>
53 </form>
54 </div>
55 </div>
56 {% endblock %}
```


h. Halaman register

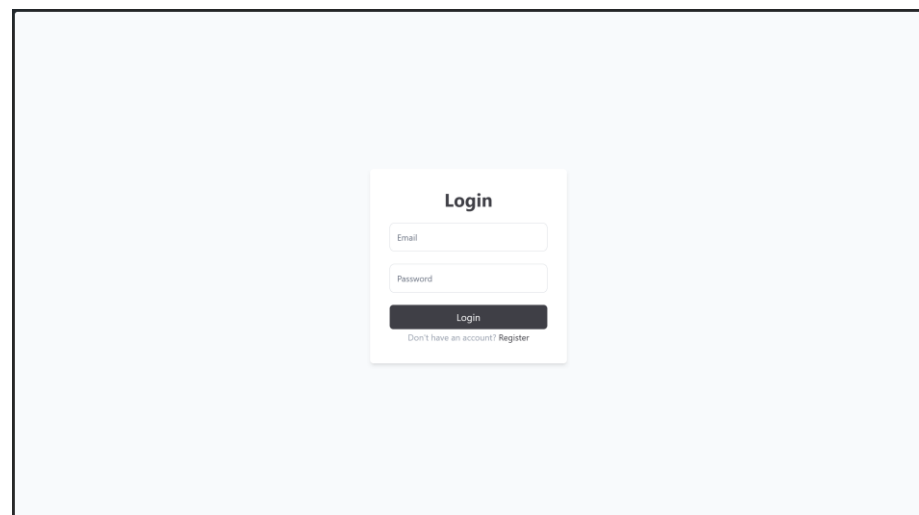
```
1 {% extends 'layout.html' %} {% block head %}
2 <title>Register Page</title>
3 {% endblock %} {% block body %}
4 <div class="flex container mx-auto justify-center items-center h-screen">
5   <div class="bg-white p-8 rounded-md shadow-md space-y-5">
6     <h1 class="text-3xl text-zinc-700 font-bold text-center">Register</h1>
7     <form method="POST" class="space-y-5">
8       <div class="relative">
9         <input
10           type="text"
11           id="username"
12           class="block px-2.5 pb-2.5 pt-4 w-full text-sm text-gray-900 bg-transparent rounded-lg
border-1 border-gray-300 appearance-none focus:outline-none focus:ring-0 focus:border-zinc-700
peer"
13           placeholder=" "
14           name="inpUsername"
15         />
16         <label
17           for="username"
18           class="absolute text-sm text-gray-500 transition-all duration-300 transform -translate-
y-4 scale-75 top-2 z-10 origin-[0] bg-white px-2 peer-focus:px-2 peer-focus:text-zinc-700 peer-
placeholder-shown:scale-100 peer-placeholder-shown:-translate-y-1/2 peer-placeholder-shown:top-1/2
peer-focus:top-2 peer-focus:scale-75 peer-focus:-translate-y-4 rtl:peer-focus:translate-x-1/4
rtl:peer-focus:left-auto start-1 hover:cursor-text"
19         >Username</label>
20       </div>
21       <div class="relative">
22         <input
23           type="email"
24           id="email"
25           class="block px-2.5 pb-2.5 pt-4 w-full text-sm text-gray-900 bg-transparent rounded-lg
border-1 border-gray-300 appearance-none focus:outline-none focus:ring-0 focus:border-zinc-700
peer"
26           placeholder=" "
27           name="inpEmail"
28         />
29         <label
30           for="email"
31           class="absolute text-sm text-gray-500 transition-all duration-300 transform -translate-
y-4 scale-75 top-2 z-10 origin-[0] bg-white px-2 peer-focus:px-2 peer-focus:text-zinc-700 peer-
placeholder-shown:scale-100 peer-placeholder-shown:-translate-y-1/2 peer-placeholder-shown:top-1/2
peer-focus:top-2 peer-focus:scale-75 peer-focus:-translate-y-4 rtl:peer-focus:translate-x-1/4
rtl:peer-focus:left-auto start-1 hover:cursor-text"
32         >Email</label>
33       </div>
34       <div class="relative">
35         <input
36           type="password"
37           id="password"
38           class="block px-2.5 pb-2.5 pt-4 w-full text-sm text-gray-900 bg-transparent rounded-lg
border-1 border-gray-300 appearance-none focus:outline-none focus:ring-0 focus:border-zinc-700
peer"
39           placeholder=" "
40           name="inpPass"
41         />
42         <label
43           for="password"
44           class="absolute text-sm text-gray-500 transition-all duration-300 transform -translate-
y-4 scale-75 top-2 z-10 origin-[0] bg-white px-2 peer-focus:px-2 peer-focus:text-zinc-700 peer-
placeholder-shown:scale-100 peer-placeholder-shown:-translate-y-1/2 peer-placeholder-shown:top-1/2
peer-focus:top-2 peer-focus:scale-75 peer-focus:-translate-y-4 rtl:peer-focus:translate-x-1/4
rtl:peer-focus:left-auto start-1 hover:cursor-text"
45         >Password</label>
46       </div>
47       <div class="text-center">
48         <button
49           type="submit"
50           class="w-full bg-zinc-700 text-white p-2 rounded-md hover:bg-zinc-600 hover:cursor-
pointer transition-all duration-300"
51         >Register
52       </button>
53       <span class="text-gray-400 text-sm">
54         >Already have an account?
55         <a href="/" class="text-zinc-700">Login</a></span>
56       </div>
57     </form>
58   </div>
59 </div>
60 </div>
61 </div>
62 </div>
63 </div>
64 </div>
65 {% endblock %}
```

i. Halaman home

Saya menambahkan sedikit detail pada halaman home yaitu menampilkan *username* dan email *user* yang sedang login, dan memberikan text tebal pada tabel untuk *user* yang sedang login.

```
1 {% extends 'layout.html' %} {% block head %}
2 <title>Home</title>
3 {% endblock %} {% block body %}
4 <div class="mt-5 container max-w-3xl mx-auto">
5   <div class="relative overflow-x-auto shadow-md sm:rounded-lg">
6     <table
7       class="w-full text-sm text-left rtl:text-right text-gray-500"
8     >
9       <thead
10        class="text-xs text-gray-100 uppercase bg-zinc-700"
11      >
12        <tr>
13          <th scope="col" class="px-6 py-3">No</th>
14          <th scope="col" class="px-6 py-3">Username</th>
15          <th scope="col" class="px-6 py-3">Password</th>
16          <th scope="col" class="px-6 py-3">Email</th>
17        </tr>
18      </thead>
19      <tbody>
20        {% for user in users %}
21        <tr class="bg-white border-b border-gray-200 {% if user[1] == user_data[0] %} font-
semibold {% endif %}">
22          <td class="px-6 py-4">{{ loop.index }}</td>
23          <td class="px-6 py-4">{{ user[1] }}</td>
24          <td class="px-6 py-4">{{ user[2] }}</td>
25          <td scope="row"
26            class="px-6 py-4 whitespace-nowrap">{{ user[3] }}</td>
27        </tr>
28        {% endfor %}
29      </tbody>
30    </table>
31  </div>
32 </div>
33 {% endblock %}
```

j. Hasil



zaki

zaki@xyz.com

Logout

NO	USERNAME	PASSWORD	EMAIL
1	adhitsa	adhitsa123	adhitsa@xyz.com
2	ali	ali123	ali@xyz.com
3	zaki	zaki123	zaki@xyz.com

Register

Username

Email

Password

Register

Already have an account? Login

Login

Email

Password

Invalid email or password

Login

Don't have an account? Register

KESIMPULAN

Dengan demikian dapat disimpulkan bahwa sistem berhasil mengimplementasikan fitur login, register, dan logout menggunakan Flask, MySQL, dan session. Proses autentikasi pengguna dilakukan dengan memanfaatkan mekanisme session untuk menyimpan data pengguna yang sedang aktif. Saat pengguna melakukan login, sistem akan memvalidasi data yang dikirim melalui form menggunakan method “POST”, kemudian memeriksa kecocokan antara email dan password di database. Jika data sesuai, maka session akan dibuat dan pengguna diarahkan ke halaman home, sedangkan jika tidak sesuai, pengguna akan dikembalikan ke halaman login.

Dengan adanya session ini, aplikasi dapat mengenali status pengguna tanpa harus melakukan login berulang kali. Selain itu, penggunaan Flask memudahkan dalam pengelolaan rute, koneksi ke database MySQL, serta pengaturan alur autentikasi. Secara keseluruhan, program ini telah berhasil mewujudkan sistem login yang aman, terstruktur, dan mudah dikembangkan untuk kebutuhan web berbasis Python.

Repository Github: <https://github.com/muzaaqi/pemrograman-web-praktik/tree/main/meet-4>