

MEDIA STREAMING WITH IBM CLOUD STREAMING

PHASE – 3

DEVELOPMENT PHASE – 1

Date	18-october-2023
Team ID	5562
Team Name	Proj_227250_Team_1
Project Name	Media Streaming With IBM Cloud Streaming

1.INTRODUCION:

Streaming files like audio, video and others are stored on a server somewhere on the world wide web(WWW). When a user request file ,It gets transmitted over the web as sequential packets of data that are streamed instantly . Since streaming data is broken down into data packets ,its transmission is similar to that of other types of data sent over the internet.

An audio or video player hosted by the browser accepts the flow of data packets from the streaming service's remote server and interprets them as video or audio, then plays the media for the user.

STREAMING REQUIREMENTS:

Streaming usually requires a reliable, high-speed internet connection because the media files must be retrieved from a remote location and then delivered to a user's local system with minimal lag or latency (delay). A slow connection decreases the speed at which the content is delivered, affecting the user's streaming experience.

INSTALL PYTHON IN OS:

1. **Download PyCharm:**

Visit the JetBrains website (<https://www.jetbrains.com/pycharm/download/>) and download the community (free) or professional version of PyCharm, depending on your needs. Make sure to download the Windows version.

2. **Run the Installer:**

Locate the downloaded installer (usually an .exe file) and double-click it to run the installation.

3. **Choose Installation Type:**

During the installation, you'll be prompted to select the installation type. You can choose the default settings or customize them according to your preferences. You can also select the option to create associations for .py files, which allows you to open Python scripts with PyCharm by default.

4. **Select the Destination Folder:**

Choose the folder where you want to install PyCharm, or leave it at the default location.

5. **Create Shortcuts:**

You can choose whether you want to create desktop shortcuts or add PyCharm to the Start menu.

6. **Complete the Installation:**

Click the "Install" button to start the installation process. PyCharm will be installed on your system.

7. **Launch PyCharm:**

Once the installation is complete, you can launch PyCharm by checking the "Run PyCharm" option in the installer or by finding it in your Start menu or desktop shortcuts.

INSTALLING PACKAGES :

1.PACKAGE NAME: Flask

USE : to use flask framework in python

COMMAND TO INSTALL: pip install flask

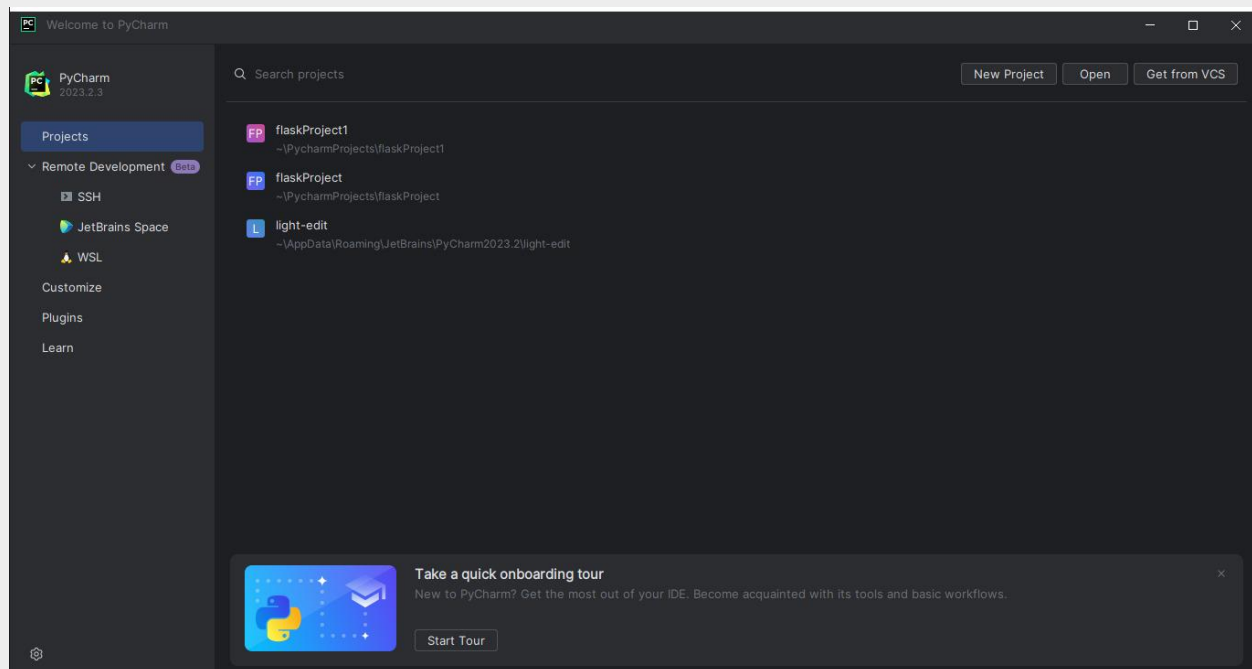
```
Command Prompt
Collecting flask
  Downloading flask-3.0.0-py3-none-any.whl (99 kB)
----- 99.7/99.7 kB 5.6 MB/s eta 0:00:00
Collecting Werkzeug>=3.0.0
  Downloading werkzeug-3.0.0-py3-none-any.whl (226 kB)
----- 226.6/226.6 kB 13.5 MB/s eta 0:00:00
Collecting Jinja2>=3.1.2
  Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
----- 133.1/133.1 kB ? eta 0:00:00
Collecting itsdangerous>=2.1.2
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting click>=8.1.3
  Downloading click-8.1.7-py3-none-any.whl (97 kB)
----- 97.9/97.9 kB 5.5 MB/s eta 0:00:00
Collecting blinker>=1.6.2
  Downloading blinker-1.6.3-py3-none-any.whl (13 kB)
Collecting colorama
  Downloading colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-2.1.3-cp311-cp311-win_amd64.whl (17 kB)
Installing collected packages: MarkupSafe, itsdangerous, colorama, blinker, Werkzeug, Jinja2, click, flask
Successfully installed Jinja2-3.1.2 MarkupSafe-2.1.3 Werkzeug-3.0.0 blinker-1.6.3 click-8.1.7 colorama-0.4.6 flask-3.0.0 itsdangerous-2.1.2

[notice] A new release of pip available: 22.3 -> 23.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip

C:\Users\Elcot>
```

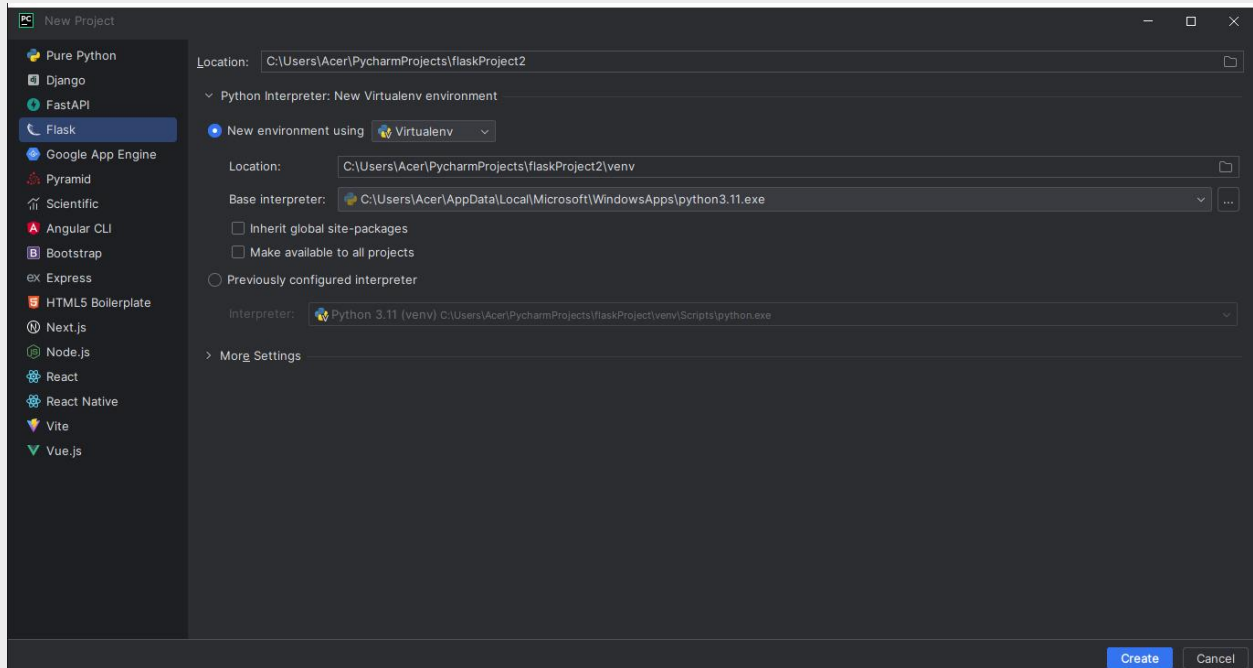
GETTING STARTED WITH PYCHARM:

1. To create a new project click on new project:

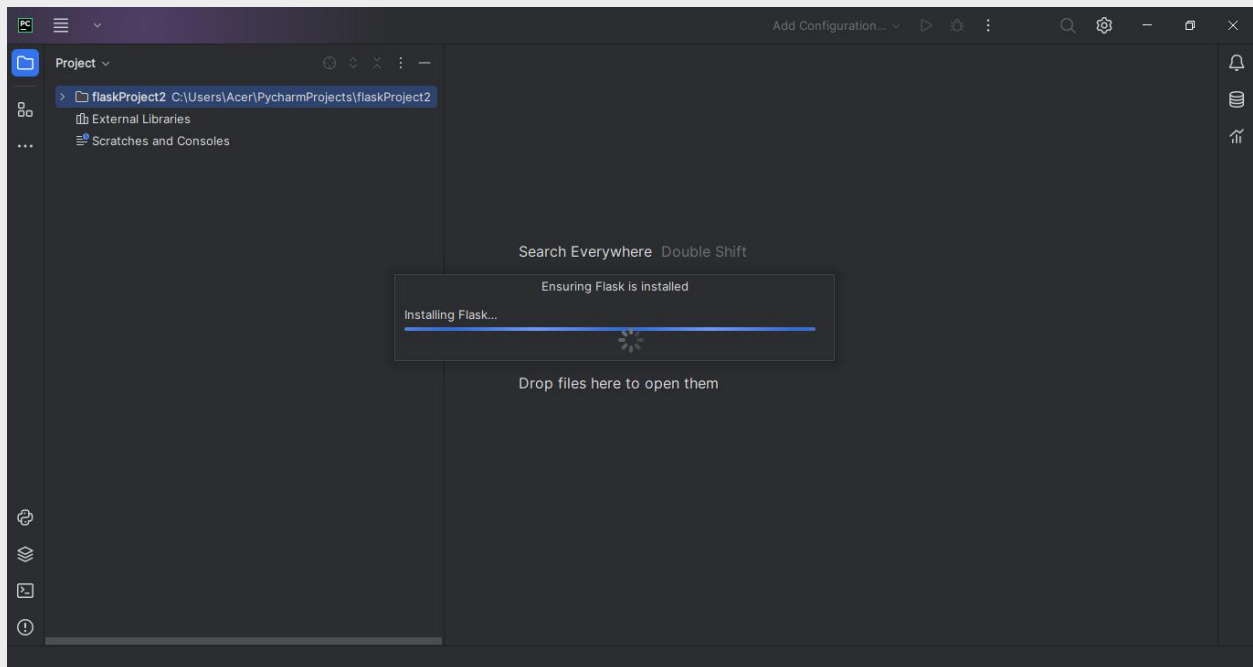


2. Select flask to create a project in flask program

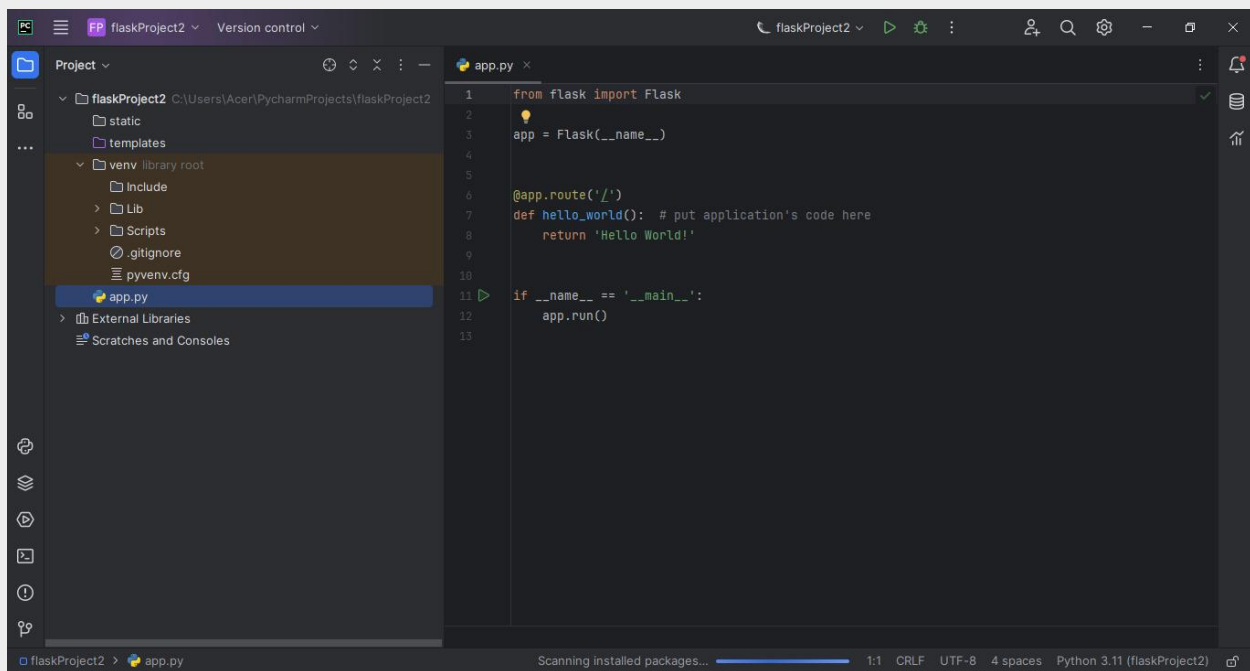
- Select your environment as you need.
- Select the required path you need the projects to be stored.



3. Pycharm will automatically installs required packages for flask automatically.



4. Required folders will be installed automatically. Then we can ready to code successfully.

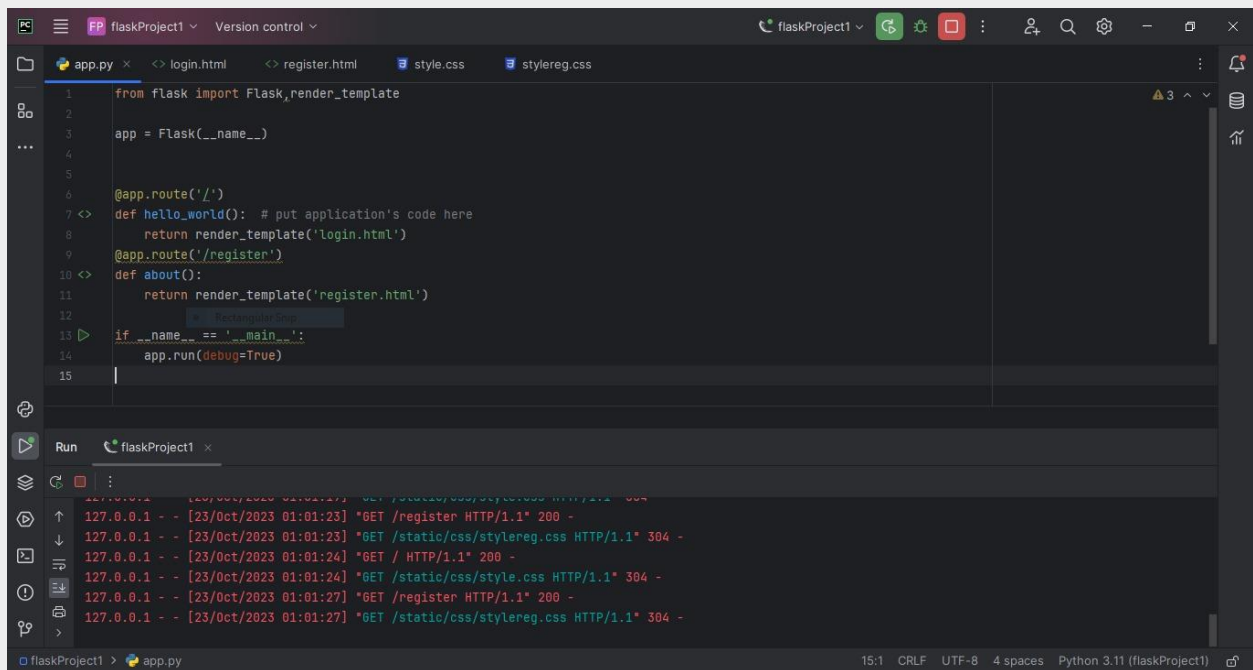


STEPS TO CREATE LOGIN AND REGISTRATION PAGE:

1. Code this to render a html code in flask

```
from flask import Flask,render_template

app = Flask(__name_)
@app.route('/')
def hello_world(): # put application's code here
    return render_template('login.html')
@app.route('/register')
def about():
    return render_template('register.html')
if __name__ == '__main__':
    app.run(debug=True)
```



```
1 from flask import Flask, render_template
2
3 app = Flask(__name__)
4
5
6 @app.route('/')
7 def hello_world(): # put application's code here
8     return render_template('login.html')
9 @app.route('/register')
10 def about():
11     return render_template('register.html')
12
13 if __name__ == '__main__':
14     app.run(debug=True)
15
```

Run flaskProject1

```
127.0.0.1 - - [23/Oct/2023 01:01:23] "GET /static/css/stylereg.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Oct/2023 01:01:23] "GET /register HTTP/1.1" 200 -
127.0.0.1 - - [23/Oct/2023 01:01:24] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [23/Oct/2023 01:01:24] "GET /static/css/style.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Oct/2023 01:01:27] "GET /register HTTP/1.1" 200 -
127.0.0.1 - - [23/Oct/2023 01:01:27] "GET /static/css/stylereg.css HTTP/1.1" 304 -
```

2. Code for login page html:

```
<!doctype html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<title>CodePen - Animated Login Form using Html & CSS Only</title>
```

```
<link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='css/style.css') }}">
```

</head>

```
<body> <!-- partial:index.partial.html -->
```

[illegible]

[illegible]

<div class="signin">


```
<div class="content">
```

```
<h2>Sign In</h2>
```

```
<div class="form">
```

```
<div class="inputBox">
```

```
<input type="text" required> <i>Email</i>
```

```
</div>
```

```
<div class="inputBox">
```

```
<input type="password" required> <i>Password</i>
```

```
</div>
```

```
<div class="links"> <a href="#">Forgot Password</a> <a  
href="/register">Signup</a>
```

```
</div>
```

```
<div class="inputBox">
```

`<input type="submit" value="Login">`

`</div>`

`</div>`

`</div>`

`</div>`

`</section> <!-- partial -->`

`</body>`

`</html>`

```
1 <!doctype html>
2
3 <html lang="en">
4
5 <head>
6
7   <meta charset="UTF-8">
8
9   <title>CodePen - Animated Login Form using Html & CSS Only</title>
10
11   <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/style.css') }}">
12 </head>
13
14 <body> <!-- partial:index.partial.html -->
15
16 <section> <span></span> <span></span> <span></span> <span></span> <span></span> <span></span> <span></span> <span></span> <span></span> <span></span> <span></span> <span></span>
17
18 <div class="signin">
19
20   <div class="content">
21
22     <h2>Sign In</h2>
23
24   <div class="form">
25
26     <div class="inputBox">
```

3. Code for registration page html

[illegible]

[illegible]

<div class="signup">

<div class="content">

<h2>Sign Up</h2>

<div class="form">

<div class="inputBox">

<input type="text" required> <i>Username</i>

</div>

<div class="inputBox">

<input type="number" required> <i>Mobile No</i>

</div>

<div class="inputBox">

<input type="text" required> <i>Email</i>

</div>

<div class="inputBox">

<input type="password" required> <i>Password</i>

</div>

<div class="links"> already a member login

</div>

<div class="inputBox">

<input type="submit" value="Create">

</div>

</div>

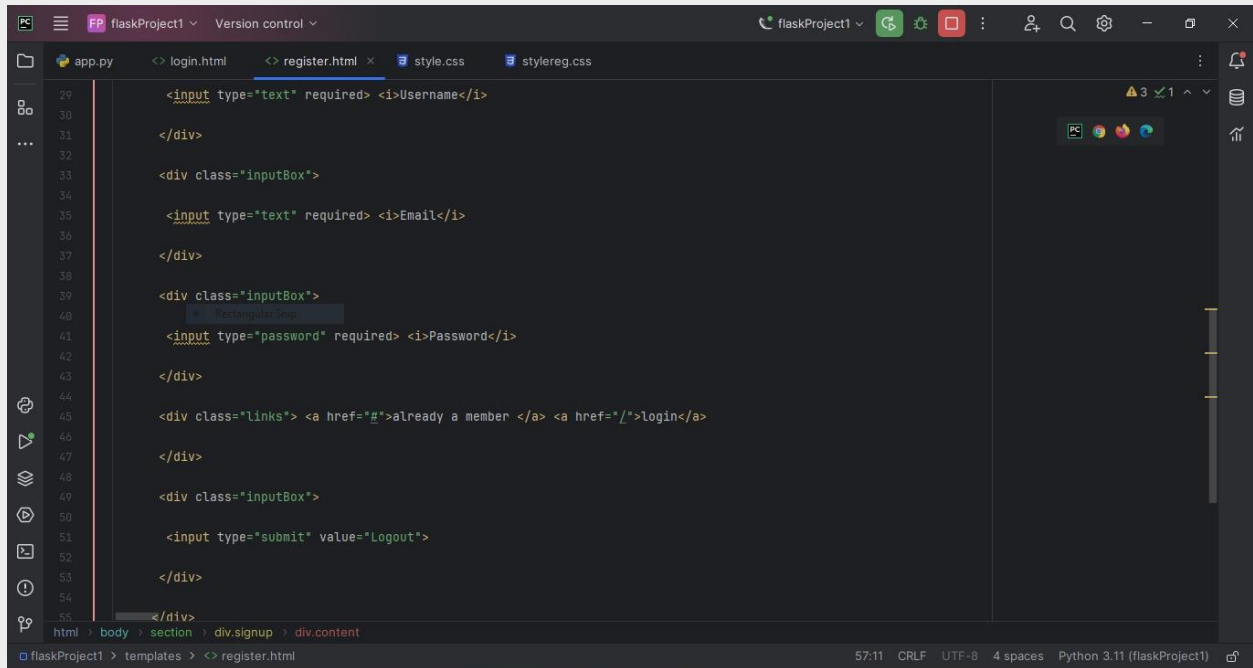
</div>

</div>

</section> <!-- partial -->

</body>

</html>

A screenshot of a code editor window titled 'flaskProject1'. The editor shows a file named 'register.html' with HTML code for a registration form. The code includes input fields for 'Username', 'Email', and 'Password', a 'Logout' button, and links for 'already a member' and 'login'. The editor has a dark theme and a sidebar on the left showing the file structure. The status bar at the bottom indicates '57:11 CRLF UTF-8 4 spaces Python 3.11 (flaskProject1)'.

```
29 <input type="text" required> <i>Username</i>
30
31 </div>
32
33 <div class="inputBox">
34
35 <input type="text" required> <i>Email</i>
36
37 </div>
38
39 <div class="inputBox">
40 <input type="password" required> <i>Password</i>
41
42 </div>
43
44 <div class="links"> <a href="#">already a member </a> <a href="/">login</a>
45
46 </div>
47
48 <div class="inputBox">
49
50 <input type="submit" value="Logout">
51
52 </div>
53
54 </div>
```

4. SYNCHRONIZING A CSS FILE WITH LOGIN HTML FOR DESIGNING PURPOSE:

```
@import
url('https://fonts.googleapis.com/css2?family=Quicksand:wght@300;400;500;
600;700&display=swap');

*

{

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Quicksand', sans-serif;

}

body

{
```

```
display: flex;
justify-content: center;
align-items: center;
min-height: 100vh;
background: #000;
}
```

section

```
{
position: absolute;
width: 100vw;
height: 100vh;
display: flex;
justify-content: center;
align-items: center;
gap: 2px;
flex-wrap: wrap;
overflow: hidden;
}
```

section::before

```
{
content: "";
position: absolute;
width: 100%;
```



```
height: 100%;  
background: linear-gradient(#000,#0f0,#000);  
animation: animate 5s linear infinite;  
}
```

```
@keyframes animate
```

```
{  
  0%  
  {  
    transform: translateY(-100%);  
  }  
  100%  
  {  
    transform: translateY(100%);  
  }  
}
```

```
section span
```

```
{  
  position: relative;  
  display: block;  
  width: calc(6.25vw - 2px);  
  height: calc(6.25vw - 2px);  
  background: #181818;  
  z-index: 2;
```

```
    transition: 1.5s;
}
section span:hover
{
    background: #0f0;
    transition: 0s;
}
```

```
section .signin
{
    position: absolute;
    width: 400px;
    background: #222;
    z-index: 1000;
    display: flex;
    justify-content: center;
    align-items: center;
    padding: 40px;
    border-radius: 4px;
    box-shadow: 0 15px 35px rgba(0,0,0,9);
}
```

```
section .signin .content
```

```
{  
  position: relative;  
  width: 100%;  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  flex-direction: column;  
  gap: 40px;  
}
```

section .signin .content h2

```
{  
  font-size: 2em;  
  color: #0f0;  
  text-transform: uppercase;  
}
```

section .signin .content .form

```
{  
  width: 100%;  
  display: flex;  
  flex-direction: column;  
  gap: 25px;  
}
```

section .signin .content .form .inputBox

```
{
  position: relative;
  width: 100%;
}
section .signin .content .form .inputBox input
{
  position: relative;
  width: 100%;
  background: #333;
  border: none;
  outline: none;
  padding: 25px 10px 7.5px;
  border-radius: 4px;
  color: #fff;
  font-weight: 500;
  font-size: 1em;
}
section .signin .content .form .inputBox i
{
  position: absolute;
  left: 0;
  padding: 15px 10px;
  font-style: normal;
```

```
color: #aaa;

transition: 0.5s;

pointer-events: none;
}

.signin .content .form .inputBox input:focus ~ i,
.signin .content .form .inputBox input:valid ~ i
{
  transform: translateY(-7.5px);
  font-size: 0.8em;
  color: #fff;
}

.signin .content .form .links
{
  position: relative;
  width: 100%;
  display: flex;
  justify-content: space-between;
}

.signin .content .form .links a
{
  color: #fff;
  text-decoration: none;
}
```

```
.signin .content .form .links a:nth-child(2)
{
  color: #0f0;
  font-weight: 600;
}

.signin .content .form .inputBox input[type="submit"]
{
  padding: 10px;
  background: #0f0;
  color: #000;
  font-weight: 600;
  font-size: 1.35em;
  letter-spacing: 0.05em;
  cursor: pointer;
}

input[type="submit"]:active
{
  opacity: 0.6;
}

@media (max-width: 900px)
{
  section span
  {
```

```
width: calc(10vw - 2px);

height: calc(10vw - 2px);

}

}

@media (max-width: 600px)

{

section span

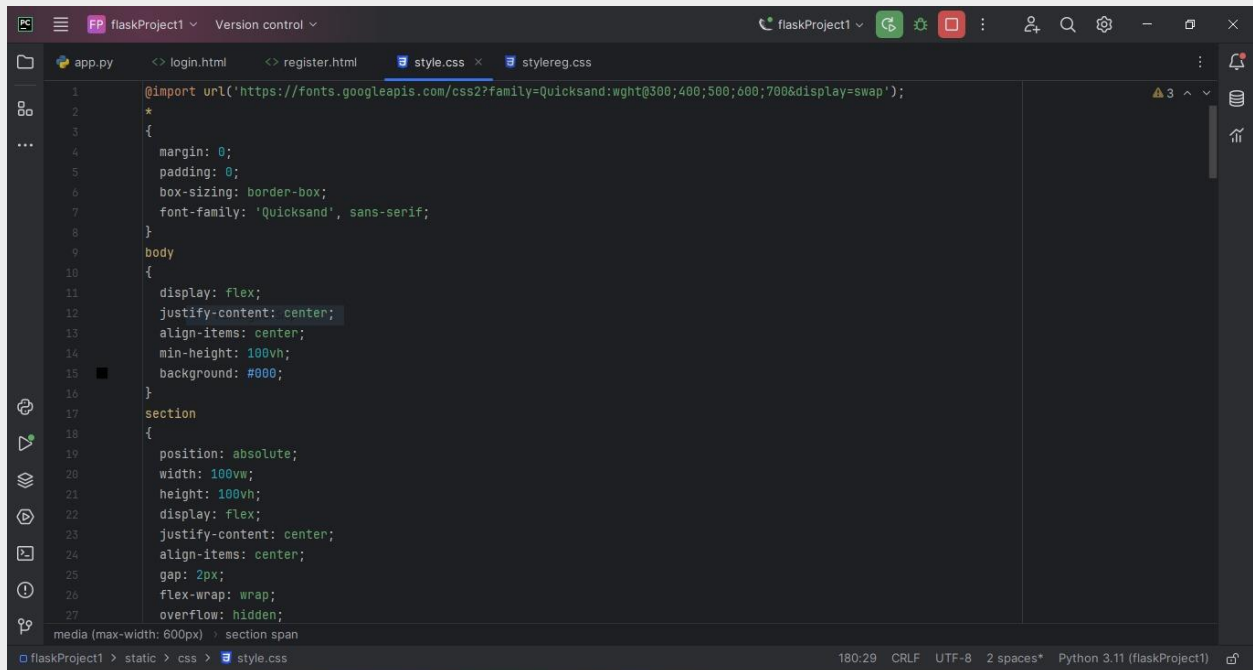
{

width: calc(20vw - 2px);

height: calc(20vw - 2px);

}

}
```



The screenshot shows a VS Code editor window with the following details:

- File Explorer:** Shows files `app.py`, `login.html`, `register.html`, `style.css` (selected), and `stylereg.css`.
- Code Editor:** Displays the content of `style.css`. The code includes an `@import` statement for Google Fonts, a `body` selector with flexbox and background settings, and a `section` selector with absolute positioning and flexbox. A media query is partially visible at the bottom.
- Code Content:**

```
1  @import url('https://fonts.googleapis.com/css2?family=Quicksand:wght@300;400;500;600;700&display=swap');
2  *
3  {
4      margin: 0;
5      padding: 0;
6      box-sizing: border-box;
7      font-family: 'Quicksand', sans-serif;
8  }
9  body
10 {
11     display: flex;
12     justify-content: center;
13     align-items: center;
14     min-height: 100vh;
15     background: #000;
16 }
17 section
18 {
19     position: absolute;
20     width: 100vw;
21     height: 100vh;
22     display: flex;
23     justify-content: center;
24     align-items: center;
25     gap: 2px;
26     flex-wrap: wrap;
27     overflow: hidden;
```
- Media Query:** The bottom of the image shows the start of a media query: `media (max-width: 600px) {` followed by `section span`.
- Status Bar:** Shows the file path `flaskProject1 > static > css > style.css`, line 180:29, and encoding/formatting settings (CRLF, UTF-8, 2 spaces, Python 3.11).

5. SYNCHRONIZING A CSS FILE WITH REGISTRATION HTML FOR DESIGNING PURPOSE:

```
@import
url('https://fonts.googleapis.com/css2?family=Quicksand:wght@300;400;500;
600;700&display=swap');
*
{
margin: 0;
padding: 0;
box-sizing: border-box;
font-family: 'Quicksand', sans-serif;
}
body
{
display: flex;
justify-content: center;
align-items: center;
min-height: 100vh;
background: #000;
}
section
{
position: absolute;
width: 100vw;
```



```
height: 100vh;
display: flex;
justify-content: center;
align-items: center;
gap: 2px;
flex-wrap: wrap;
overflow: hidden;
}
section::before
{
content: "";
position: absolute;
width: 100%;
height: 100%;
background: linear-gradient(#000,#0f0,#000);
animation: animate 5s linear infinite;
}
@keyframes animate
{
0%
{
transform: translateY(-100%);
}
}
```

100%

```
{  
  transform: translateY(100%);  
}  
}
```

section span

```
{  
  position: relative;  
  display: block;  
  width: calc(6.25vw - 2px);  
  height: calc(6.25vw - 2px);  
  background: #181818;  
  z-index: 2;  
  transition: 1.5s;  
}
```

section span:hover

```
{  
  background: #0f0;  
  transition: 0s;  
}
```

section .signup

```
{
```

```
position: absolute;
width: 400px;
background: #222;
z-index: 1000;
display: flex;
justify-content: center;
align-items: center;
padding: 40px;
border-radius: 4px;
box-shadow: 0 15px 35px rgba(0,0,0,9);
}
```

```
section .signup .content
```

```
{
position: relative;
width: 100%;
display: flex;
justify-content: center;
align-items: center;
flex-direction: column;
gap: 40px;
}
```

```
section .signup .content h2
```

```
{
  font-size: 2em;
  color: #0f0;
  text-transform: uppercase;
}
section .signup .content .form
{
  width: 100%;
  display: flex;
  flex-direction: column;
  gap: 25px;
}
section .signup .content .form .inputBox
{
  position: relative;
  width: 100%;
}
section .signup .content .form .inputBox input
{
  position: relative;
  width: 100%;
  background: #333;
  border: none;
```

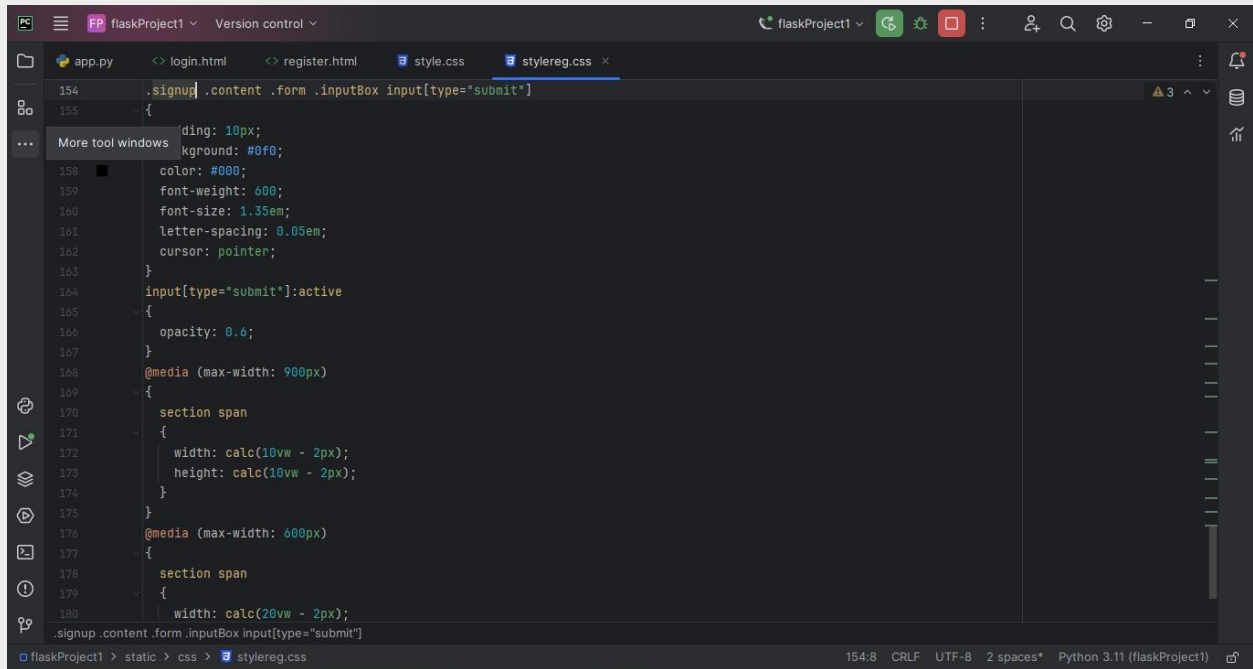
```
outline: none;
padding: 25px 10px 7.5px;
border-radius: 4px;
color: #fff;
font-weight: 500;
font-size: 1em;
}
section .signup .content .form .inputBox i
{
position: absolute;
left: 0;
padding: 15px 10px;
font-style: normal;
color: #aaa;
transition: 0.5s;
pointer-events: none;
}
.signup .content .form .inputBox input:focus ~ i,
.signup .content .form .inputBox input:valid ~ i
{
transform: translateY(-7.5px);
font-size: 0.8em;
color: #fff;
```

```
}  
  
.signup .content .form .links  
{  
  position: relative;  
  width: 100%;  
  display: flex;  
  justify-content: space-between;  
}  
  
.signup .content .form .links a  
{  
  color: #fff;  
  text-decoration: none;  
}  
  
.signup .content .form .links a:nth-child(2)  
{  
  color: #0f0;  
  font-weight: 600;  
}  
  
.signup .content .form .inputBox input[type="submit"]  
{  
  padding: 10px;  
  background: #0f0;  
  color: #000;
```

```
font-weight: 600;
font-size: 1.35em;
letter-spacing: 0.05em;
cursor: pointer;
}
input[type="submit"]:active
{
  opacity: 0.6;
}
@media (max-width: 900px)
{
  section span
  {
    width: calc(10vw - 2px);
    height: calc(10vw - 2px);
  }
}
@media (max-width: 600px)
{
  section span
  {
    width: calc(20vw - 2px);
    height: calc(20vw - 2px);
```

}

}



The screenshot shows a code editor with a dark theme. The file explorer on the left shows a project named 'flaskProject1' with files 'app.py', 'login.html', 'register.html', 'style.css', and 'stylereg.css'. The 'stylereg.css' file is open in the editor. The code is CSS for a submit button, with line numbers 154 to 180 visible. The code includes a selector for the submit button, a list of styles (background, color, font, cursor), an active state, and two media queries for responsive design.

```
154 .signup .content .form .inputBox input[type="submit"]
155 {
156     background: #0f0;
157     color: #000;
158     font-weight: 600;
159     font-size: 1.35em;
160     letter-spacing: 0.05em;
161     cursor: pointer;
162 }
163
164 input[type="submit"]:active
165 {
166     opacity: 0.6;
167 }
168
169 @media (max-width: 900px)
170 {
171     section span
172     {
173         width: calc(10vw - 2px);
174         height: calc(10vw - 2px);
175     }
176 }
177
178 @media (max-width: 600px)
179 {
180     section span
181     {
182         width: calc(20vw - 2px);
183     }
184 }
```

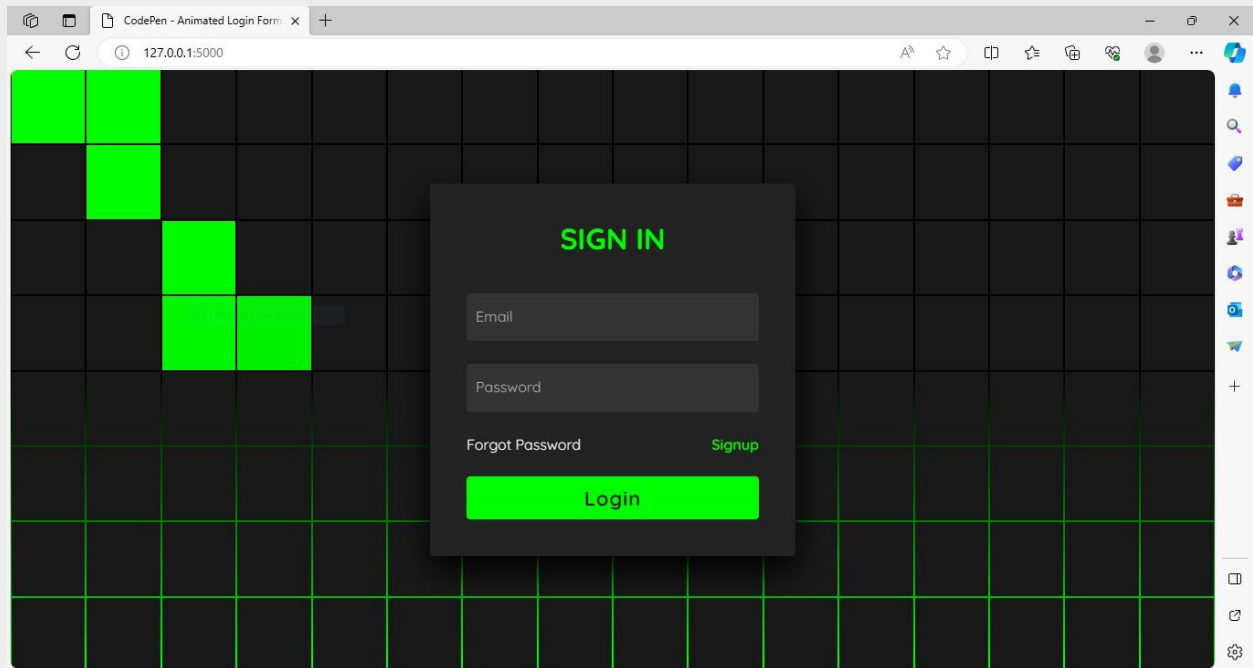
6. CREATING A SQL TO STORE THE DATA'S IN THE DATABASE:

CREATE DATABASE IF NOT EXISTS UserDatabase;

USE UserDatabase;

CREATE TABLE IF NOT EXISTS Users (
 UserID INT AUTO_INCREMENT PRIMARY KEY,
 Username VARCHAR(50) NOT NULL,
 MobileNo VARCHAR(15) NOT NULL,
 EmailID VARCHAR(100) NOT NULL,
 Password VARCHAR(255) NOT NULL
);

7. FINAL OUTPUT FOR LOGIN PAGE:



8. FINAL OUTPUT FOR REGISTRATION PAGE:

