

# Flask Session

The concept of a session is very much similar to that of a cookie. However, the session data is stored on the server.

The session can be defined as the duration for which a user logs into the server and logs out. The data which is used to track this session is stored into the temporary directory on the server.

The session data is stored on the top of cookies and signed by the server cryptographically.

In the flask, a session object is used to track the session data which is a dictionary object that contains a key-value pair of the session variables and their associated values.

The following syntax is used to set the session variable to a specific value on the server.

```
Session[<variable-name>] = <value>
```

To remove a session variable, use the pop() method on the session object and mention the variable to be removed.

```
session.pop(<variable-name>, none)
```

Let's see a simple example to understand how can we set and get the session variable.

## Example

```
from flask import *
app = Flask(__name__)
app.secret_key = "abc"

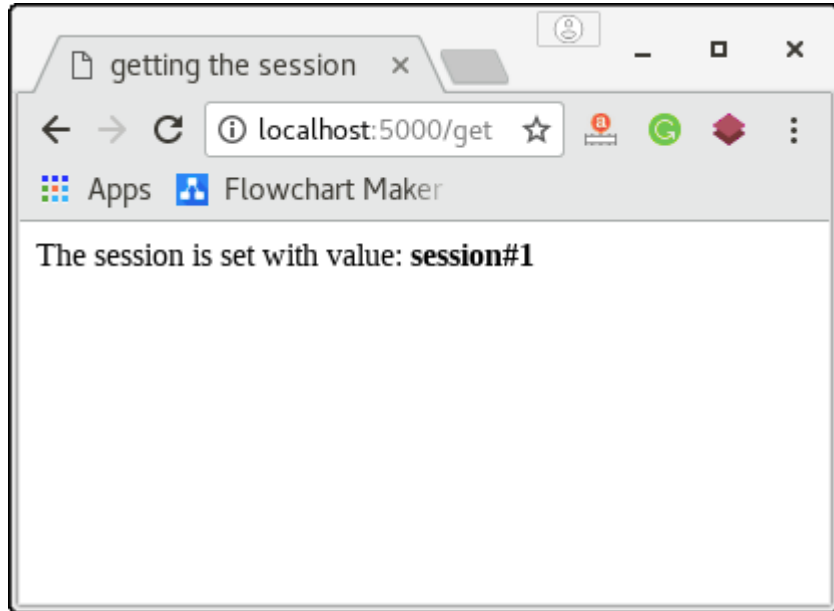
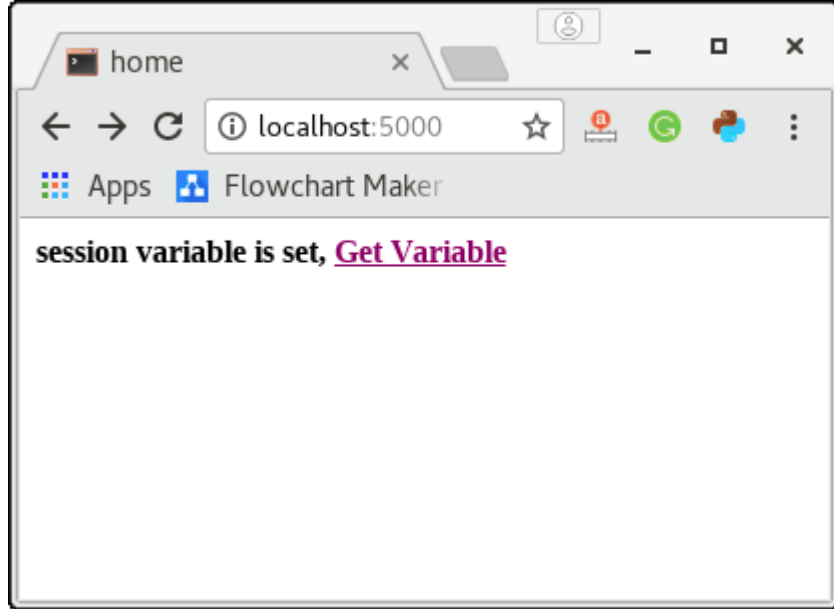
@app.route('/')
def home():
    res = make_response("<h4>session variable is set, <a href='/get'>Get Variable</a></h4>")
    session['response']='session#1'
    return res;

@app.route('/get')
def getVariable():
    if 'response' in session:
        s = session['response'];
        return render_template('getsession.html',name = s)

if __name__ == '__main__':
    app.run(debug = True)
```

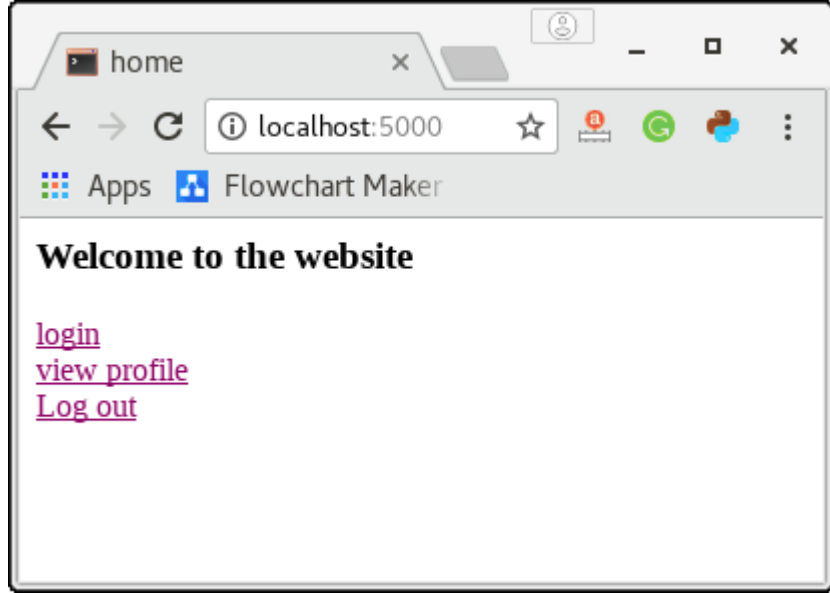
## getsession.html

```
<html>
<head>
<title>getting the session</title>
</head>
<body>
<p>The session is set with value: <strong>{{name}}</strong></p>
</body>
</html>
```

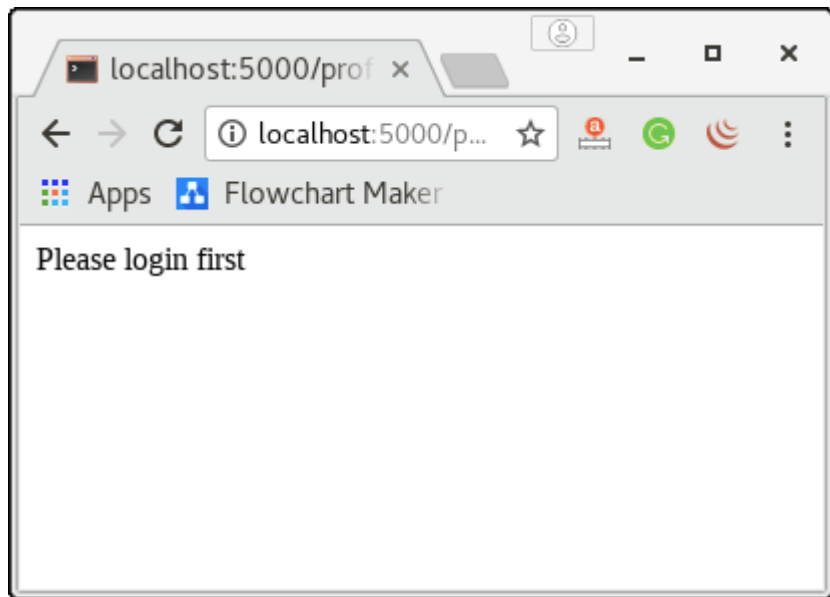


## Login Application in the flask using Session

Here, we will create a login application in the flask where the following home page is shown to the user.

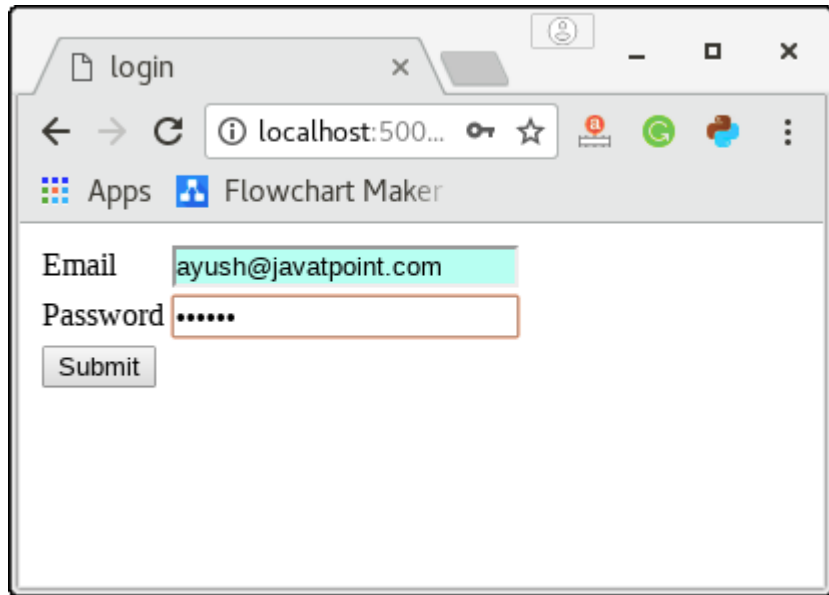


If we click the **view\_profile** directly without login, then it will show some warning as we can't visit the profile directly without login.



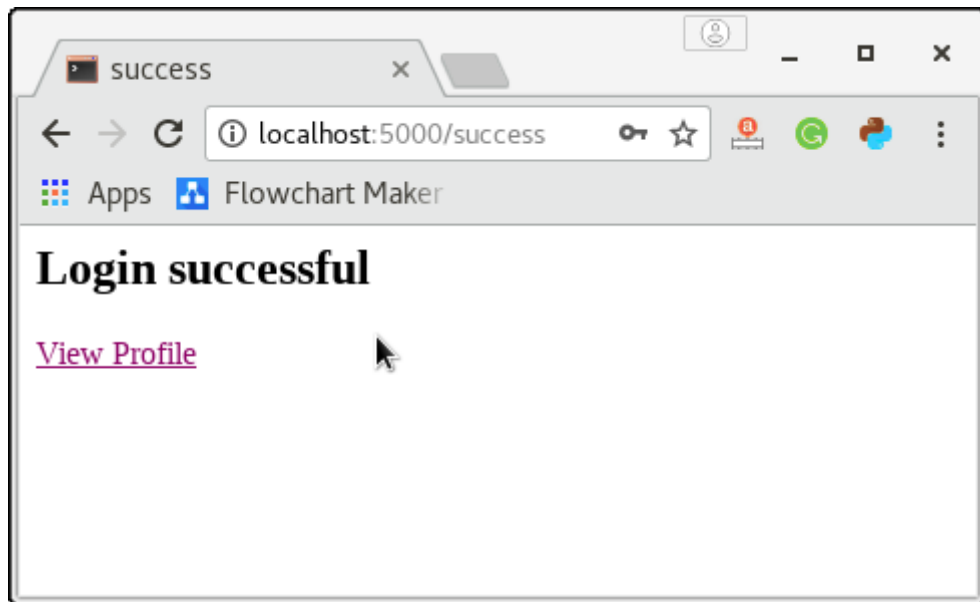
If we visit the login page then, the application shows the login page to the user where the user is prompted to enter the email id and password. Here, our application redirects the user to success page for any random valid email id and password as given in the below

image.

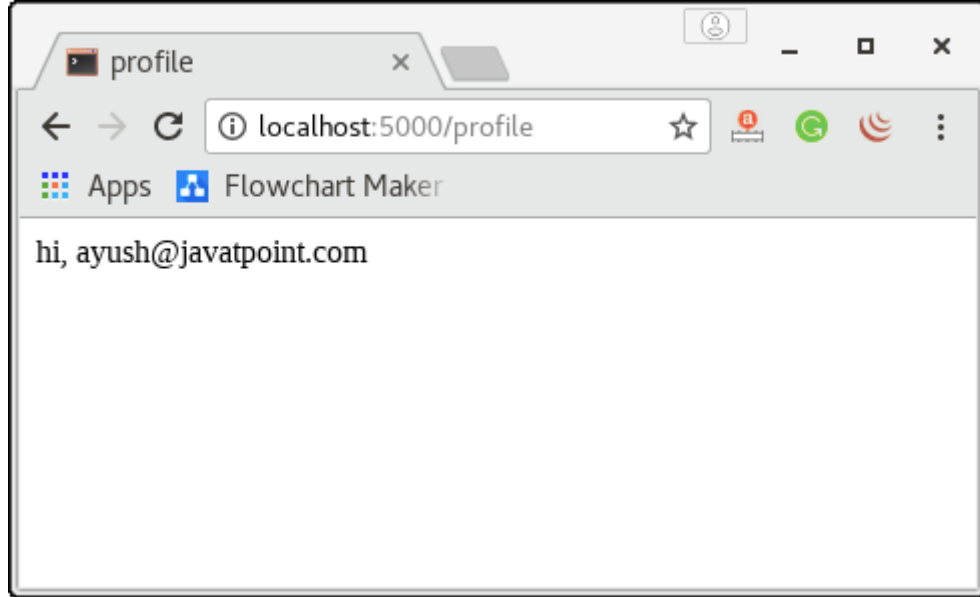


A screenshot of a web browser window with a single tab titled 'login'. The address bar shows 'localhost:500...'. The page contains a login form with two input fields: 'Email' with the value 'ayush@javatpoint.com' and 'Password' with masked characters '.....'. A 'Submit' button is located below the password field. The browser's taskbar at the bottom shows 'Apps' and 'Flowchart Maker'.

On submit, the user is redirected to the **success.html** as given in the below image.

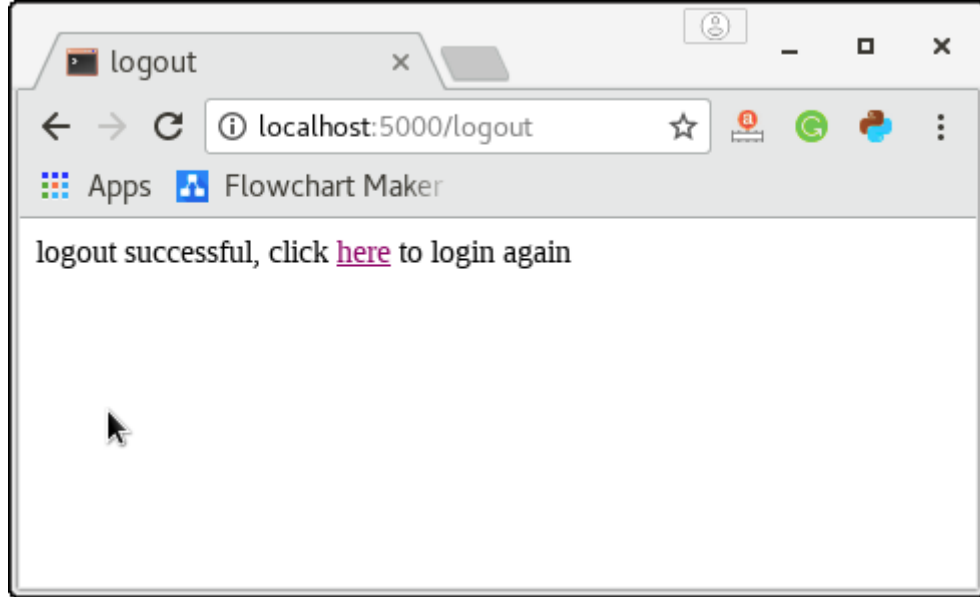


Now, if we visit the profile page, it will show the message with the session name as the session variable 'email' is set to its value.



As we now logged in to the application, we can now view the user's profile without logging in to the application as we have already logged in to the application. To test this, go back to the home page and click the link view\_profile, we will get the result as shown in the above image.

The application facilitates us to log out the session, for this purpose, go back to the home page click on the logout button. It will destroy all the session variables set for this particular server.



Now, the user has to login again with the email and password to view the profile on the application.

This is a simple login application built using python flask that implements the session. Here, the flask script **login.py** acts like the main controller which contains the view functions (home(), login(), success(), logout(), and profile()) which are associated with the URL mappings (/ , /login, /success, /logout, /profile) respectively.

**login.py**

```
from flask import *
app = Flask(__name__)
app.secret_key = "ayush"

@app.route('/')
def home():
    return render_template("home.html")

@app.route('/login')
def login():
    return render_template("login.html")

@app.route('/success', methods = ["POST"])
def success():
    if request.method == "POST":
        session['email']=request.form['email']
        return render_template('success.html')

@app.route('/logout')
def logout():
    if 'email' in session:
        session.pop('email',None)
        return render_template('logout.html');
    else:
        return '<p>user already logged out</p>'

@app.route('/profile')
def profile():
    if 'email' in session:
        email = session['email']
        return
    render_template('profile.html',name=email)
    else:
        return '<p>Please login first</p>'

if __name__ == '__main__':
    app.run(debug = True)
```

**home.html**



```
<html>
<head>
<title>home</title>
</head>
<body>
<h3>Welcome to the website</h3>
<a href = "/login">login</a><br>
<a href = "/profile">view profile</a><br>
<a href = "/logout">Log out</a><br>
</body>
</html>
```

## login.html

```
<html>
<head>
  <title>login</title>
</head>
<body>
  <form method = "post" action = "http://localhost:5000/success">
    <table>
      <tr><td>Email</td><td><input type = 'email' name = 'email'></td></tr>
      <tr><td>Password</td><td><input type = 'password' name = 'pass'></td>
    </tr>
      <tr><td><input type = "submit" value = "Submit"></td></tr>
    </table>
  </form>
</body>
</html>
```

## success.html

```
<html>
<head>
<title>success</title>
</head>
<body>
  <h2>Login successful</h2>
  <a href="/profile">View Profile</a>
</body>
</html>
```

## logout.html

```
<html>
<head>
  <title>logout</title>
</head>

<body>
<p>logout successful, click <a href="/login">here</a> to login again</p>
</body>

</html>
```

[← Prev](#)[Next →](#)

 Youtube For Videos Join Our Youtube Channel: [Join Now](#)

## Feedback

- Send your Feedback to [\[email protected\]](#)

## Help Others, Please Share



[Learn Latest Tutorials](#)