

Advanced Programming with Python

sessions and cookies

Pepe García jgarciah@faculty.ie.edu



SCHOOL OF
SCIENCE &
TECHNOLOGY

Plan for today

- Review last day's homework
- HTML Forms
- Sessions and Cookies
- Authentication



Forms in HTML are pieces of the UI which we use to send data from client to server.

Forms in HTML are pieces of the UI which we use to send data from client to server.

```
<form action="/handle-login" method="POST">
```

```
</form>
```

Forms in HTML are pieces of the UI which we use to send data from client to server.

```
<form action="/handle-login" method="POST">  
    <input type="text" name="username"/>  
  
</form>
```

Forms in HTML are pieces of the UI which we use to send data from client to server.

```
<form action="/handle-login" method="POST">  
    <input type="text" name="username"/>  
    <input type="password" name="password"/>  
  
</form>
```

Forms in HTML are pieces of the UI which we use to send data from client to server.

```
<form action="/handle-login" method="POST">
    <input type="text" name="username"/>
    <input type="password" name="password"/>
    <input type="submit"/>
</form>
```

Whenever a form is submitted, it will send an HTTP requests to the URL specified in `action`, with the method specified in `method`, and sending the form data in the body of the request

Receiving form data in flask

```
from flask import request
```

```
@app.route("/handle-login", method=["POST"])
```

```
def handle_login():
```

```
    username = request.form["username"]
```

```
    password = request.form["password"]
```

```
# Now, do whatever you want with user and password, for ex
```

```
# new session for the user
```



Adding login to our users backoffice

Adding login to our user's backoffice

Let's add login functionality to our user's backoffice!

There's a feature of most web applications that we haven't yet discussed, **sessions**.

Session in the web allow to create sections of websites that are private, for which users need to authenticate in order to access.

Sessions are hold by making HTTP use a special kind of header called cookie.

Cookies are headers that the server sends alongside the HTTP response, that the client **will send back** in subsequent requests!

Whiteboard

Let's whiteboard the whole flow of cookies and sessions

Using sessions in flask

sessions in flask are handled by importing the session object from flask:

```
from flask import request, session
```

```
@app.route("/handle-login", method=["POST"])
```

```
def handle_login():
```

```
    username = request.form["username"]
```

```
    password = request.form["password"]
```

```
    session["username"] = username # this way, we can identify
```



Using sessions in flask

sessions in flask are handled by importing the session object from flask:

```
from flask import session
```

```
@app.route("/")
```

```
def index():
```

```
    if "user_id" in session: # session behaves like a dictionary
```

```
        return render_template("index.html")
```

```
    else:
```

```
        return render_template("login.html")
```

