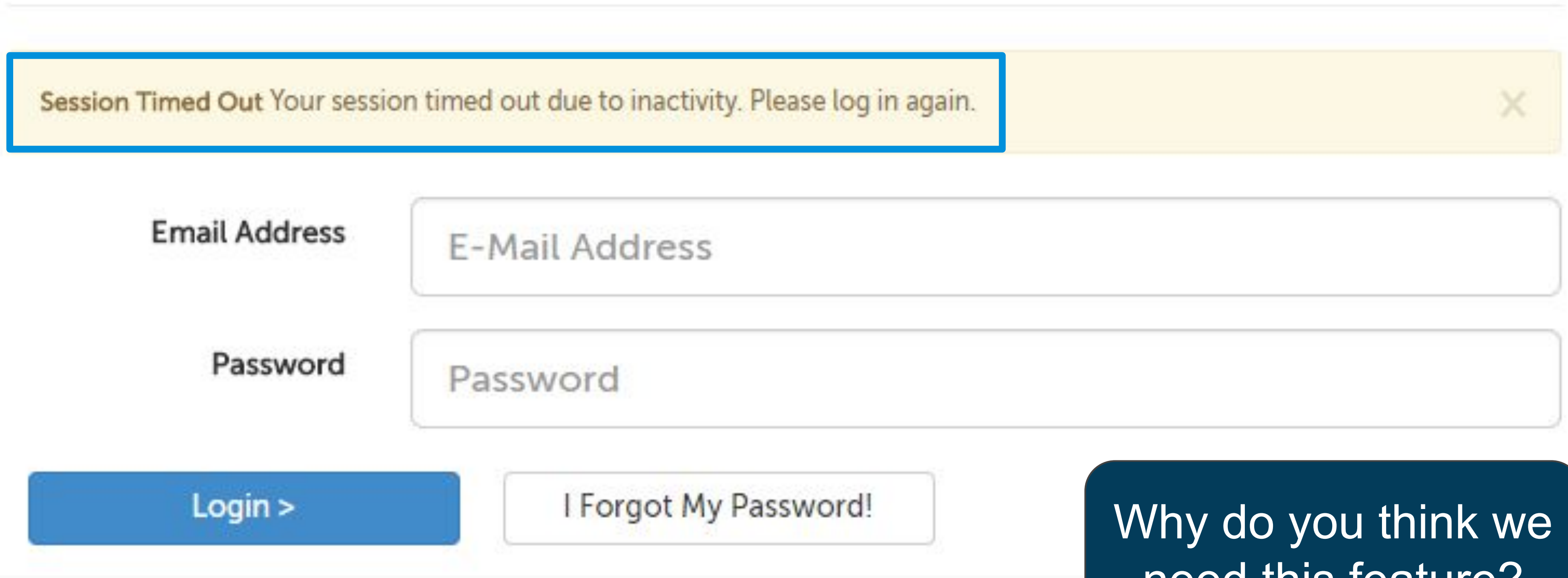
Lab 6: Session Data in Flask

GW CS 2541: Database Systems and Team Projects - 2022
Prof. Tim Wood, Ethan Baron, and Catherine Meadows

Has this ever happened to you?



Session Timed Out Your session timed out due to inactivity. Please log in again.

Email Address	E-Mail Address
Password	Password

Why do you think we need this feature?

Session Data

- "Session" refers to the time between a client logging in to the server and logging out of the server
- With Flask, Session data is stored in the client's browser on top of cookies
- Each client has their own session that is assigned a **Session ID**
- Use Cases
  - Remember a user when they log in
  - Store items in a cart while shopping online
- Sessions last for 31 days unless SESSION_PERMANENT is set to false (in which case they last until the browser or tab is closed)

Using Session with Flask

- The Session object is a dictionary object with key-value pairs of session variables and associated values
- For session data to be encrypted, also set a SECRET_KEY

To set a ‘username’ session variable:

```
session[‘username’] = “admin”
```

To release a session variable:

```
session.pop(‘username’, None)
```

To set the session secret key:

```
app.secret_key = “any string”
```

To clear all session variables:

```
session.clear()
```

Redirecting in Flask

Tip: In repl.it, view your web app in a new tab rather than the default window or your redirect() functions may not work correctly

```
from flask import Flask, redirect, url_for
app = Flask('app')

@app.route('/')
def login():
     ...


@app.route('/logout')
def logout():
     session.clear()
     return redirect('/')


app.run(host='0.0.0.0', port=8080)
```

- The redirect() function allows us to redirect users to a URL that we specify
- Instead of specifying a URL, we can also redirect to a function using url_for()
- For example, the following lines would be equivalent for our code example:

```
redirect('/')
```

```
redirect(url_for('login'))
```

Session Example

```
from flask import Flask, session, redirect
app = Flask('app')
app.secret_key = "secret"


...
@app.route('/home')
def home():
    if 'name' in session:
        return render_template("home.html")
    return redirect('/')


app.run(host='0.0.0.0', port=8080)
```

Why do we check the session to make sure a user is logged in?

home.html

```
<html>
<body>
    <h1> Welcome, {{ session['name'] }} </h1
</body>
</html>
```

We can access our session variables in templates, too!

Refresher: Form Data

```
from flask import Flask, render_template, request
app = Flask(‘app’)

@app.route(‘/’, methods=[‘GET’, ‘POST’])
def get_username():
     if request.method == ‘POST’:
          uname = request.form[“username”]
     return render_template(‘simple_form.html’)
 app.run(host=’0.0.0.0’, port=8080)
```

```
<body>
 <form action="/" method="POST">
   <input type="text" name="username">
   <input type="submit" name="submit">
 </form>
</body>
```

Common Mistakes / More Tips!

1. You must set up your database connection and create a cursor object **within each function** in your Flask app
2. If you are getting a Python indentation / tab error but everything looks aligned on your screen, this is likely due to a collaboration lag in Repl. Have every group member check the spacing on their own screen and adjust!
3. If you want styling tips or aren't sure about syntax for HTML / CSS, w3schools.com is a great resource!
4. If you need to reset your database, run the following command in the **Shell**:

```
sqlite3 <db file name> ".read <sql file name>"
```

Activity 1: Login Page

1. Create a login page that takes a username and password, verifies the user is in the database, and signs them in
   - Display an error message on the login page if authentication fails

2. Upon successful login, the user should be redirected to a home page that displays “Welcome, <NAME> ” at the top (using **session variables**!)
   a. Add a Sign Out button on the home page that clears the session and redirects the user back to the login page
   b. Users should not be able to access the home page if not signed in

Activity 2: User Login

1. Extend activity 1 so that when a username and password is determined to be in the database, also store the type of user in a session variable (The three user roles are: Student, TA, Professor)
2. When signed in, the home page should display different things based on the type of user stored in the session
    - Students can view the student roster (name, ID, and email of all students)
    - TAs can view the student roster and engagement points
    - Professors can view the student roster, engagement points, and grades