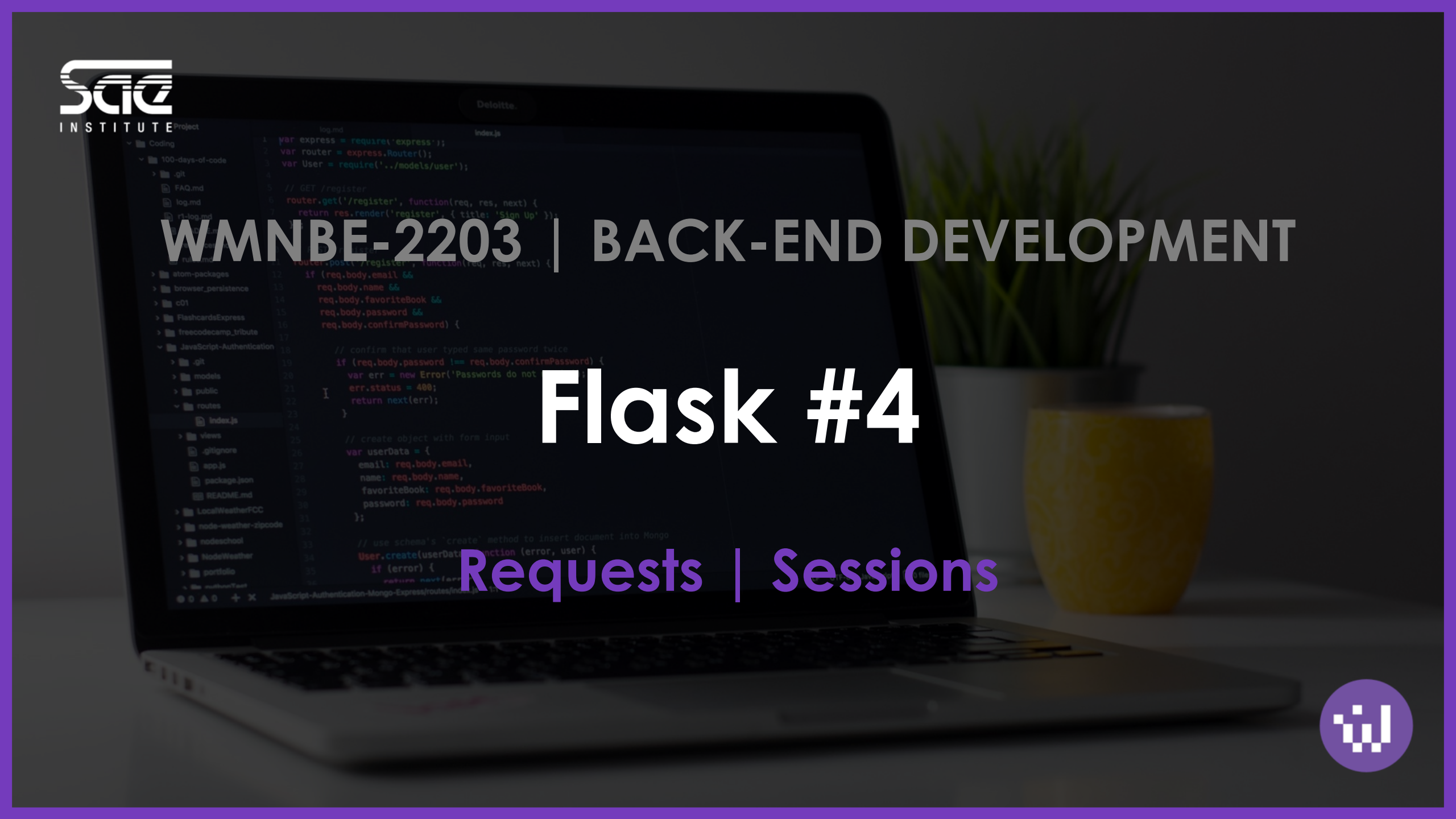

WMNBE-2203 | BACK-END DEVELOPMENT

Flask #4

Requests | Sessions

ΠΕΡΙΕΧΟΜΕΝΑ

SAE INSTITUTE

Περιεχόμενα

- **HTTP** methods
  - **methods** argument
  - **app.get()** & **app.post()**
  - **request** object
- Sessions
  - **session** object
  - **app.secret_key**
  - **flash()**
  - **get_flashed_messages()**

HTTP METHODS

HTTP methods

methods argument

Ένα **route** στο **Flask** "απαντά" μόνο σε **GET requests**, εκτός και αν οριστεί διαφορετικά.

Αν θέλουμε κάποιο **route** / **view function** να διαχειρίζεται και **GET** αλλά και **POST requests**, πρέπει να δώσουμε κατάλληλη τιμή στην παράμετρο **methods** του **route** *decorator*:

```
@app.route('/route', methods=['GET', 'POST'])
def view_function():
  …
```

HTTP methods

methods argument

Προφανώς, μπορούμε να έχουμε και **view functions** που διαχειρίζονται μόνο **POST requests**.

```
@app.route('/route', methods=['POST'])
def view_function():
  …
```

Αν κάποιος επιχειρίσει να ζητήσει ένα **route** με μέθοδο που δεν έχει οριστεί στην παράμετρο **methods**, το **Flask** επιστρέφει ένα *405 Method not allowed* σφάλμα.

HTTP methods

app.get() & app.post()

Στην έκδοση 2.x του **Flask** έχουμε έναν νέο τρόπο να ορίσουμε ποιες μεθόδους διαχειρίζεται ποιο **view function**.

```
@app.get('/register')
def show_registration_form():
  return render_template('registration_form.html')

@app.post('/register')
def register_user():
  ...
```

HTTP methods

request object

Το **Flask** δίνει πρόσβαση σε όλες τις πληροφορίες ενός **request**, μέσα από το **request** object.

Από εκεί, μπορεί να δει κάποιος πληροφορίες όπως **method**, **path** κ.α.

```
from flask import Flask, request

…

@app.route('/route')
def view_function():
  app.logger.debug(request.path)     # /route
  app.logger.debug(request.method)   # GET | POST
  …
```

HTTP methods

request.args

Μέσα από το **request** object μπορώ να έχω πρόσβαση και στις *querystring* παραμέτρους.

```
# /products/search?query=pants

@app.get('/products/search')
def search():
  query = request.args.get('query')
  …
```

… αν και η λογική των *clean / friendly URLs* καθιστά τη χρήση των παραμέτρων αυτών, πλέον, σπάνια.

HTTP methods

request.form

Τα *form data* είναι, επίσης, διαθέσιμα από αντίστοιχο **dictionary**.

```
@app.route('/login', methods=['GET', 'POST'])
def login():
  if request.method == 'GET':
    return render_template('login-form.html')
  elif request.method == 'POST':
    username = request.form.get('username')
    password = request.form.get('password')
    …
```

SESSIONS

Sessions

session object

Μιας και το **HTTP** πρωτόκολλο είναι *stateless*, συχνά χρειαζόμαστε κάποιο μηχανισμό που να επιτρέπει την αποθήκευση στοιχείων, για κάθε χρήστη ξεχωριστά.

Στο **Flask**, όπως και σε όλα τα δημοφιλή web frameworks, υπάρχει η έννοια του **session** object.

Το **session** object λειτουργεί παρόμοια με ένα **dictionary**, όπου μπορούν να αποθηκευτούν *key-value pairs*.

Sessions

session *id* & *cookie*

Για να επιτευχθεί η λειτουργικότητα αυτή, ο server δημιουργεί ένα ψηφιακά υπογεγραμμένο **session** *cookie*, που καταστρέφεται αυτόματα μόλις κλείσει το παράθυρο του browser, με κάποιο **session** *id*.

Αυτό αποστέλλεται από τον client σε κάθε **request** και "ταυτοποιεί" τον client στον οποίο "ανήκει" το **session**.

Αν ο χρήστης δεν υποβάλει κάποιο **request** για κάποιο διάστημα (συχνά είναι στα *20* λεπτά) το **session** στο server, για το συγκεκριμένο χρήστη, καταστρέφεται.

Sessions

app.secret_key

Για να μπορέσει κάποιος να χρησιμοποιήσει το **session** object, αρκεί να ορίσει το **secret_key** κάτω από το **Flask** αντικείμενο.

```
# Set the secret key to some random bytes.
# Keep this really secret!
app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'
```

Sessions

app.secret_key

Ένας καλός τρόπος να παράξει κάποιος ένα τέτοιο κλειδί είναι να εκτελέσει το ακόλουθο σε γραμμή εντολής

```
# Recommended by the Flask docs
$ python -c 'import secrets; print(secrets.token_hex())'

# for generating a string of random bytes
$ python -c 'import os; print(os.urandom(24))'

# or as hexadecimal, an appropriate form
# when storing the key in external files
$ python -c 'import os; print(os.urandom(24).hex())'
```

Sessions

Persistent sessions

Αν θέλουμε για οποιοδήποτε λόγο το **session** *cookie* να παραμείνει και μετά το κλείσιμο του παραθύρου του browser, μπορούμε να ορίσουμε τα εξής:

```
app.config["PERMANENT_SESSION_LIFETIME"] = timedelta(days=3)
# Default: timedelta(days=31) (2678400 seconds)

@app.before_request
def make_session_permanent():
    session.permanent = True
```

Sessions

Παράδειγμα

```
from flask import session
from datetime import datetime

app = Flask(__name__)

app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'

@app.get('/')
def home():
  if not session['timestamp']:
    session['timestamp'] = datetime.now()
```

Sessions

flash()

Το μηχανισμό του **session** χρησιμοποιεί και μια ιδιαίτερα χρήσιμη λειτουργία του **Flask**, για την εμφάνιση ενημερωτικών μηνυμάτων στο χρήστη.

Με th χρήση της μεθόδου **flash(message)**, αποθηκεύεται ένα μήνυμα στο **session** μέχρι να χρησιμοποιηθεί.

```
@app.post('contact')
def send_message():
  msg = request.form.get('message')
  …
  flash('Your message has been sent!')
  return redirect(url_for('home'))
```

Sessions

get_flashed_messages()

Για να ανακτηθούν τα τυχόν υπάρχοντα μηνύματα σε ένα **template**, πρέπει να γίνει χρήση της συνάρτησης **get_flashed_messages()**.

Η συνάρτηση αυτή επιστρέφει όλα τα αποθηκευμένα μηνύματα, τα οποία θα διαγραφούν αμέσως αφού "χρησιμοποιηθούν".

```
{% with messages = get_flashed_messages() %}
    {% if messages %}
        <div>Messages:</div>
        {% for message in messages %}
          <div>{{ message }}</div>
        {% endfor %}
    {% endif %}
{% endwith %}
```

Sessions

Παράδειγμα

```python
@app.post('/register')
def register_user():
    username = request.form.get('username')
    firstname = request.form.get('firstname')
    lastname = request.form.get('lastname')
    password = request.form.get('password')

    flash('Registration was successful')
    session['username'] = username

    return redirect(url_for('home'))

@app.get('/logout')
def logout():
    session.pop('username')
    return redirect(url_for('home'))
```

Χρήσιμα links

- The Request Object — Flask Documentation (2.1.x)
https://flask.palletsprojects.com/en/2.1.x/quickstart/#…

- Sessions — Flask Documentation (2.01.x)
https://flask.palletsprojects.com/en/2.1.x/quickstart/#…

- Message Flashing — Flask Documentation (2.1.x)
https://flask.palletsprojects.com/en/2.1.x/quickstart/#…

Extra info

- Configuring Your Flask Application | by Todd Birchard | Hackers and Slackers
  https://hackingandslacking.com/configuring-your-flas…
- python - Where should I place the secret key in Flask? - Stack Overflow
  https://stackoverflow.com/questions/30873189/where…
- python - Where do I get a SECRET_KEY for Flask? - Stack Overflow
  https://stackoverflow.com/questions/34902378/where…
- python - demystify Flask app.secret_key - Stack Overflow
  https://stackoverflow.com/questions/22463939/demy…

THANK YOU!