Web Programming

Login

Leander Jehl | University of Stavanger

Login

- This lecture: Simple login using sessions.
  - Has some security flaws
- Deployment alternatives:
  - Flask-Login
  - OAuth provider, e.g. firebase.google.com

Password Hashes

```
from werkzeug.security import generate_password_hash, check_password_hash
```

- Create a salted password hash to store

```
hash = generate_password_hash("Joe123")
```

Includes a random **salt**, so no two passwords have the same hash

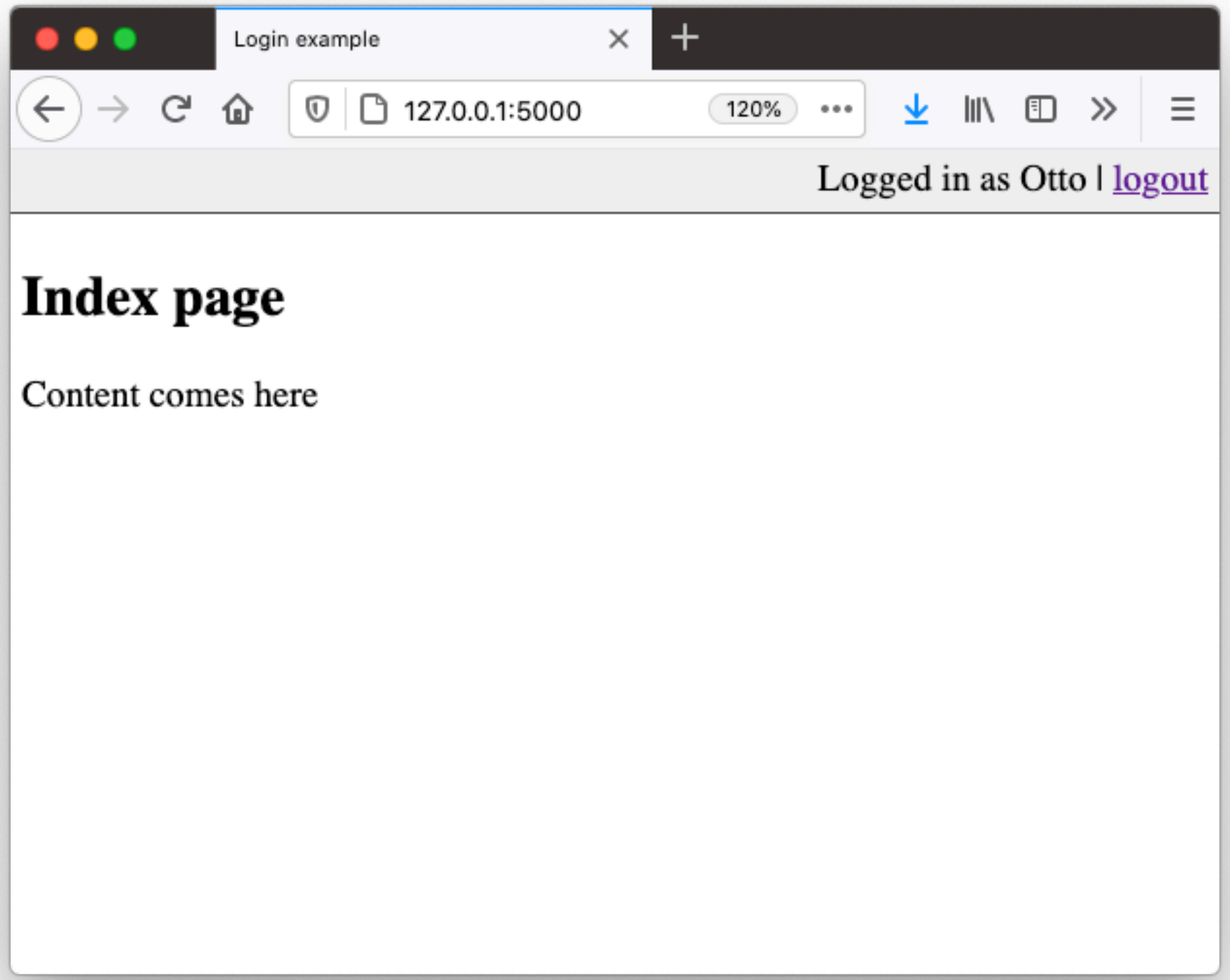
```
"pbkdf2:sha256:150000$oMxlb0Oa$125a8c19b39e0fc7e903e7775a45e40667663ed01382f9b5adcb5e0eb3d80937"
```

- Check password

```
ok = check_password_hash(hash,”Joe123")
```

Example

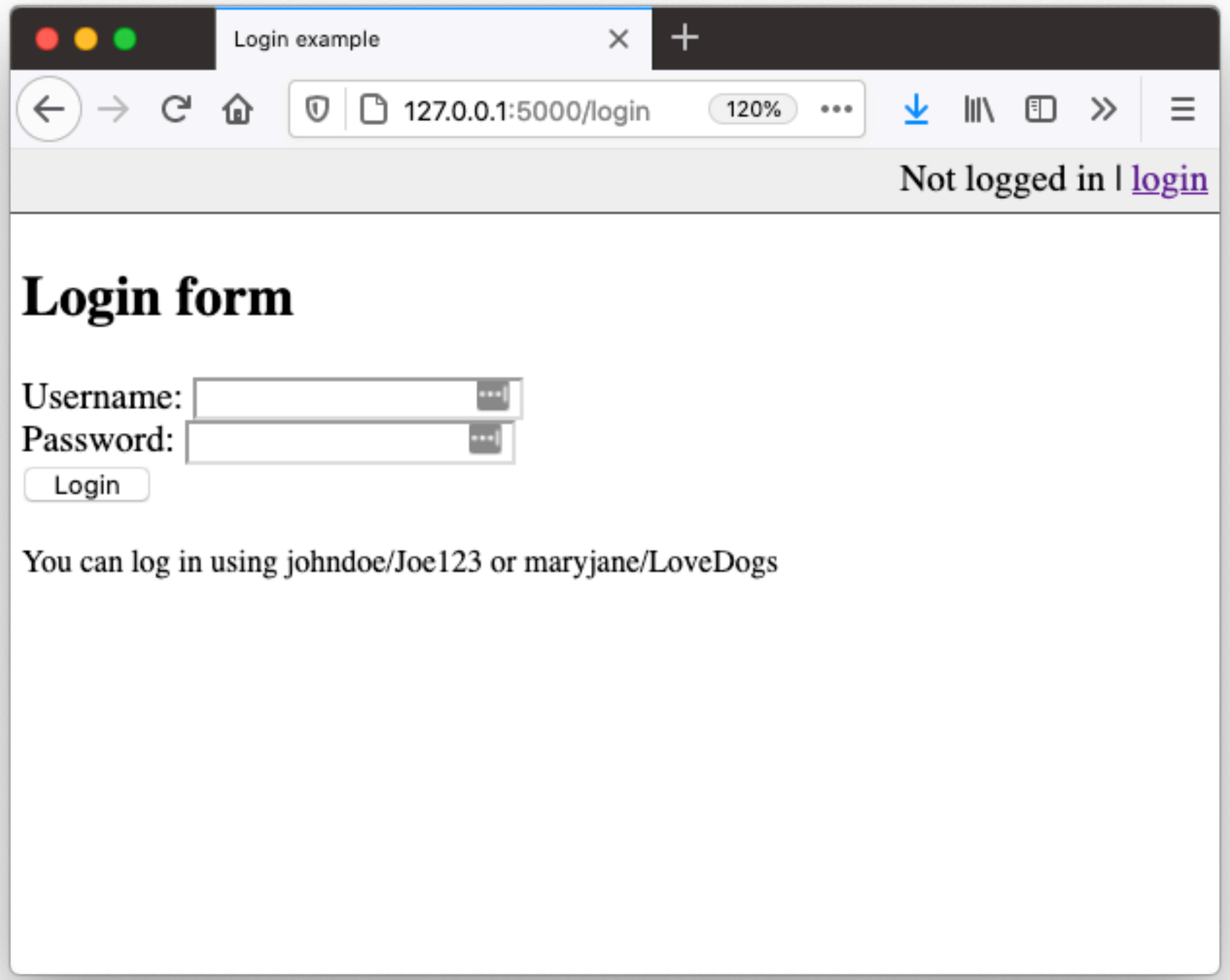
examples/python/flask/9_login/app.py



Logged in as Otto | logout

Index page

Content comes here



Not logged in | login

Login form

Username:
Password:
Login

You can log in using johndoe/Joe123 or maryjane/LoveDogs

Example

examples/python/flask/9_login/app.py

- on login, check password hash and add username to session

```python
@app.route("/login", methods=["GET", "POST"])
def login():
    username = request.form["username"]
    password = request.form["password"]

    if valid_login(username,password):
        session["username"] = username
        return redirect(url_for("index"))
```

Example

examples/python/flask/9_login/app.py

- on logout, remove username from session

```python
@app.route("/logout")
def logout():
    session.pop("username")
    return redirect(url_for("index"))
```

Exercise #1, #2, #3

github.com/dat310-spring21/course-info/tree/master/
exercises/python/flask5

Walkthrough in lecture video!

Limitation

- To further improve security session should include:
  - Unique token for every time you login
- Further, requests should contain CSRF token.
  - https://owasp.org/www-community/attacks/csrf
  - https://portswigger.net/web-security/csrf