PROGRAMIRANJE U INTEGRISANIM TEHNOLOGIJAMA

Oznaka predmeta: PIT

Predavanje broj: 10

Nastavna jedinica: Flask,

Nastavne teme:

Flask: uvod, rutiranje, metode slanja, template, stilovi, redirekcija, abort, uzimanje podataka forme, cookie, session, message flush, file uploading.

Predavač: prof. dr Perica S. Štrbac, dipl. ing.

Literatura:

Miguel Grinberg. "Flask Web Development", O'REILLY, 2014.

Flask

- Flask predstavlja web aplikativni okvir napisan u Pythonu.
- Armin Ronacher, tvorac Flask-a, vodi međunarodnu grupu Python entuzijasta pod nazivom Pocco.
- Flask je baziran na:
  - Werkzeug WSGI toolkit-u (specifikacija za univerzalni interface između web server-a i web aplikacija)
  - Jinja2 templejt engine-u

  Oba su Pocco projekti.
- Flask je preporučljiv do nivoa srednje velikih web aplikacija.
- Osnovna ideja je da se postavi sitem za rapidni razvoj web aplikacija.
- Sve što je ranije rađeno može biti uključeno u flask.
- Podrška sloju apstrakcije je realizovana razvojem adekvatnih modula (ekstenzija) koji se uključuju u flask projekat (npr. Flask Mail, Flask WTF, Flask SQLAlchemy, Flask Sijax).
- Ovde će se razvoj flask aplikacije raditi u okruženju JetBrains-oveg paketa PyCharm.
  - Drugi način je da se koristi virtualno okruženje (pip install virtualenv).

Flask: rutiranje

- Za instalaciju flask-a uraditi:

  pip install flask

  instaliraće se sve potrebne komponente

Početni kod:

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello World!'

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

- Startovati aplikaciju i proveriti odgovor na browser-u na stranici `127.0.0.1:5000`

  Hello World!

- Često root funkcija ima naziv `index()`
- Aplikacija se može pokrenuti i u debug modu sa `app.run(debug=True)`, ostali parametri: `host(127.0.0.1)`, `port(5000)`, `options(za` Werkzeug server.)

Flask: rutiranje

- Sa uključenim debug modom mogu se videti detaljne informacije HTTP komunikacije:

```
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [04/Jan/2018 12:10:20] "GET / HTTP/1.1" 200 -
```

- Dodaje se rutiranje pomoću dekoratora:

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello_world(): return 'Hello World!'

@app.route('/ford')
def ford():
    return '<h1> FORD </h1>'

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

```
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [04/Jan/2018 15:13:02] "GET /ford HTTP/1.1" 200 -
```

- Startovati aplikaciju i proveriti stranicu `127.0.0.1:5000/ford`

Flask: rutiranje

- Parametrizacija rutiranja:

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello_world(): return 'Hello World!'

@app.route('/ford')
def ford():    return '<h1> FORD </h1>'

@app.route('/profile/<username>')
def profile(username):
    return '<h1> Hi '+username+'<h1>'

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

```
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [04/Jan/2018 15:25:51] "GET /profile/Elvis HTTP/1.1" 200 -
```

Proverite na stranici:

```
http://127.0.0.1:5000/profile/Elvis
```

Flask: rutiranje

- Rutiranje tipa podataka:

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world(): return 'Hello World!'

@app.route('/ford')
def ford():    return '<h1> FORD </h1>'

@app.route('/profile/<username>')
def profile(username):    return '<h1> Hi '+username+'<h1>'

@app.route('/post/<int:post_id>')
def show_id(post_id):
    return '<h1> ID: {}</h1>'.format(post_id)

if __name__ == '__main__': app.run()
   * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
   127.0.0.1 - - [04/Jan/2018 15:59:01] "GET /post/100 HTTP/1.1" 200 -
```

Proverite na stranici:

http://127.0.0.1:5000/post/100

Flask: metode slanja

- Informacije o metodi slanja podataka:

```
from flask import Flask, request
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'METHOD is {}'.format(request.method)

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

- Probajte na stranici 127.0.0.1:5000

METHOD is GET

- Pripremiti sledeći html kod:

```
<!DOCTYPE html>
<html>
<body>
<form method="POST" action="http://127.0.0.1:5000/what">
<input type=submit>
</form>
</body>
</html>
```

Flask: metode slanja

- Sada će se na stranici `127.0.0.1:5000/what` pomoću sledećeg koda dobijati informacija o načinu slanja podataka:

```
from flask import Flask, request

app = Flask(__name__)

@app.route('/')
def hello_world(): return 'Hello'

@app.route('/what',methods=['GET','POST'])
def what():
    if request.method == 'GET':
        return 'GET'
    else:
        return 'POST'

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

- Aktiviranjem prethodnog html koda i postavljanjem parametra za metodu slanja dobiće se odgovarajući odgovor na stranici `http://127.0.0.1:5000/what`

`POST → za method="POST"`

`GET → za method="GET"`

Flask: template

- Jinja2 delimiteri:
  - {% ... %} za naredbe
  - {{ ... }} za izraze
  - {# ... #} za komentare
  - # ... ## za linijske naredbe (Line Statements)
- Primer if-else-endif naredbe (templates/hello.hmtl)

```
<!DOCTYPE html>
<html lang="en"><head><meta charset="UTF-8"><title>Title</title>
</head><body>
   {% if marks>50 %}
      <h1> Your result is pass!</h1>
   {% else %}
      <h1>Your result is fail</h1>
   {% endif %}
</body></html>
```

```
from flask import Flask, render_template
app = Flask(__name__)
@app.route('/hello/<int:score>')
def hello_name(score):
   return render_template('hello.html', marks = score)
if __name__ == '__main__':   app.run(debug = True)
```

Flask: template

- Korišćenje template-a u sledećem primeru ima za cilj da se templejtu prosledi parametar i da se odgovor koji predstavlja render templejta vrati kao odgovor funkcije index:

```
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/profile/<name>')
def index(name):
    return render_template("profile.html", parametar_ime=name)

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

- U folderu templates kreirati profile.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
    <h1>I have been waiting for you {{ parametar_ime }}</h1>
</body>
</html>
```

Flask: template

- Sada se pozivanjem stranice `http://127.0.0.1:5000/profile/Obi One` dobija dogovor:

  I have been waiting for you Obi One

- Dodati u folder static fajl style.css za stilizovanje headera 1.

```
h1 {
    color: gold;
    font-style: italic;
}
```

- Izmeniti kod u fajlu `profile.html`


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">    <title>Title</title>
    <link
            rel="stylesheet"
            type="text/css"
            href="{{ url_for('static',filename='style.css') }}" >
</head>
<body>
    <h1>I have been waiting for you {{ parametar_ime }}</h1>
</body></html>
```

Flask: template

- Primer for naredbe (templates/result.html):

```
<!DOCTYPE html><html lang="en"><head>
<meta charset="UTF-8"><title>Title</title>
</head><body>
   <table border = 1>
      {% for key, value in result.items() %}
         <tr>
            <th> {{ key }} </th>
            <td> {{ value }} </td>
         </tr>
      {% endfor %}
   </table>
</body></html>
```

maths	70
phy	50
che	60

```
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/result')
def result():
   dict = {'phy':50,'che':60,'maths':70}
   return render_template('result.html', result = dict)
if __name__ == '__main__':
   app.run(debug = True)
```

Flask: redirekcija

- Korišćenje redirekcije:

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

@app.route('/admin')
def hello_admin():       return 'Hello Admin'

@app.route('/guest/<guest>')
def hello_guest(guest): return 'Hello %s as Guest' % guest

@app.route('/user/<name>')
def hello_user(name):
   if name =='admin':
      return redirect(url_for('hello_admin'))
   else:
      return redirect(url_for('hello_guest',guest = name))

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

- Startujte: 127.0.0.1:5000/user/admin , 127.0.0.1:5000/user/Elvis i biće urađena redirekcija na 127.0.0.1/5000/admin, 127.0.0.1/5000/guest/Elvis, respektivno.

Flask: redirekcija, abort

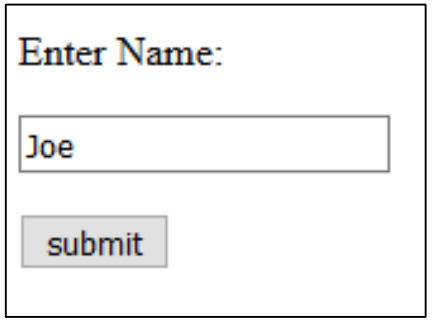
- Argumenti redirekcije:
  - **location,** URL redirekcije.
  - **status code,** slanje zaglavlja browser-u (podrazumevano 302).
  - **response,** instanca odgovora.
- Statusni kodovi:
  - `HTTP_300_MULTIPLE_CHOICES`
  - `HTTP_301_MOVED_PERMANENTLY`
  - `HTTP_302_FOUND`
  - `HTTP_303_SEE_OTHER`
  - `HTTP_304_NOT_MODIFIED`
  - `HTTP_305_USE_PROXY`
  - `HTTP_306_RESERVED`
  - `HTTP_307_TEMPORARY_REDIRECT`
- Metoda `flask.abort(code)`, kodovi su:

400 – Bad Request, **401** – Unauthenticated
403 – Forbidden, **404** – Not Found
406 – Not Acceptable, **415** – Unsupported Media Type
429 – Too Many Requests

Flask: uzimanje podataka forme

- Uzimanje podataka iz forme:

```
from flask import Flask, redirect, url_for, request
app = Flask(__name__)
@app.route('/success/<name>')
def success(name): return 'welcome %s' % name
@app.route('/login',methods = ['POST', 'GET'])
def login():
   if request.method == 'POST':
      user = request.form['nm']
      return redirect(url_for('success',name = user))
   else:
      user = request.args.get('nm')
      return redirect(url_for('success',name = user))
if __name__ == '__main__': app.run(debug = True)
```



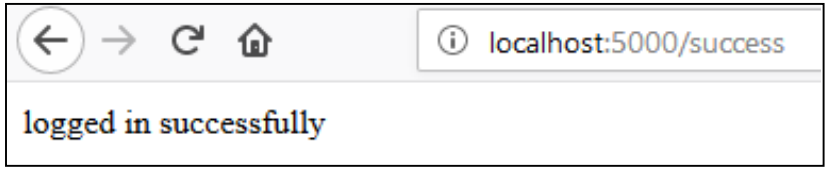
- Slanje forme:

```
<!DOCTYPE html><html lang="en">
<head><meta charset="UTF-8"><title>Title</title></head><body>
      <form action = "http://localhost:5000/login" method = "post">
         <p>Enter Name:</p>
         <p><input type = "text" name = "nm" /></p>
         <p><input type = "submit" value = "submit" /></p>
      </form></body></html>
```

Flask: redirekcija

- Neka je prethodna forma /templates/FormaLog.html i koristi se za logovanje sve dok se ne unese admin

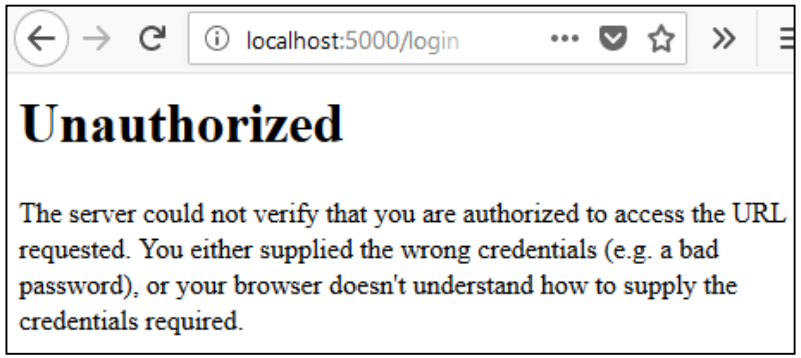
```
from flask import Flask, redirect, url_for, \
                  render_template, request
app = Flask(__name__)

@app.route('/')
def index():
    return render_template('FormaLog.html')

@app.route('/login', methods=['POST', 'GET'])
def login():
    if request.method == 'POST' and \
       request.form['nm'] == 'admin':
        return redirect(url_for('success'))
    return redirect(url_for('index'))

@app.route('/success')
def success():
    return 'logged in successfully'

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



Flask: abort

- Sada se koristi odgovor generisan metodom abort:

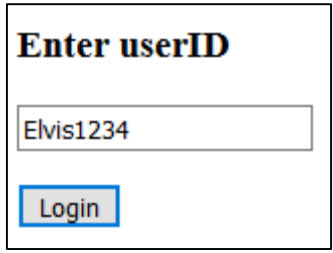
```
from flask import Flask, redirect, url_for, \
                  render_template, request, abort
app = Flask(__name__)

@app.route('/')
def index():
   return render_template('FormaLog.html')

@app.route('/login',methods=['POST','GET'])
def login():
   if request.method == 'POST':
      if request.form['nm'] == 'admin' :
         return redirect(url_for('success'))
      else:
         abort(401)
   else:
      return redirect(url_for('index'))

@app.route('/success')
def success(): return 'logged in successfully'
if __name__ == '__main__':   app.run(debug = True)
```



Flask: cookie

- Cookie je tekstualni fajl koji server postavlja na stani klijenta (ako je omogućeno u browser-u) i koji opisuje cookie (naziv, vrednost, trajanje, stazu, domen).
- Ideja je da se prevaziđe HTTP stateless protokol u smislu da se "zapamti" stanje, tako da klijent pri sledećem pristupu serveru šalje i adekvatan cookie.
- Forma `enter_cookie.html` služi za unos userID-a:

```
<!DOCTYPE html><html lang="en"><head><meta charset="UTF-8">
   <title>Title</title></head><body>
   <form action = "/setcookie" method = "POST">
      <p><h3>Enter userID</h3></p>
      <p><input type = 'text' name = 'nm'/></p>
      <p><input type = 'submit' value = 'Login'/></p>
   </form></body></html>
```



- Forma `readcookie.html` služi za link ka /getcookie ruti:

```
<!DOCTYPE html><html lang="en"><head><meta charset="UTF-8">
 <title>Title</title>
 </head>
 <body>
 <a href="/getcookie">Read cookie</a>
 </body>
 </html>
```

Flask: cookie

- U sledećem kodu na stranici 127.0.0.1:5000 vrši se unos userID-a
- Forma se prosleđuje stranici /setcookie gde se proverava metod slanja te ako je metod slanja podataka POST sledi obrada podataka inače se dojavljuje greška.
- Objekat resp se kreira funkcijom make_response, koja poziva render templejta koji omogućuje link ka stranici /getcookie, te mu se dodaje postavljanje cookie-a (metodom set_cookie) naziva userID i vrednosti koja je pročitana iz pristigle forme. Sada se objekat resp vraća kao povratna vrednost.
- Klikom na link za čitanje cookie ide se na stranicu /getcookie

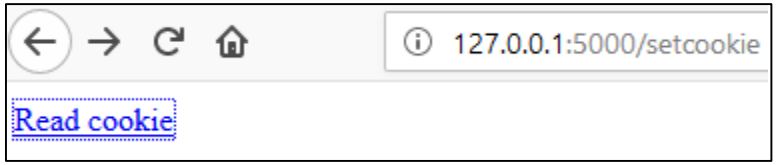
```
from flask import Flask, render_template, request, redirect, url_for,
make_response
app = Flask(__name__)

@app.route('/')
def index():
   return render_template('enter_cookie.html')

@app.route('/error/<mess>')
def error(mess):
    return '<h1> GRESKA: '+mess+ '</h1>'
```

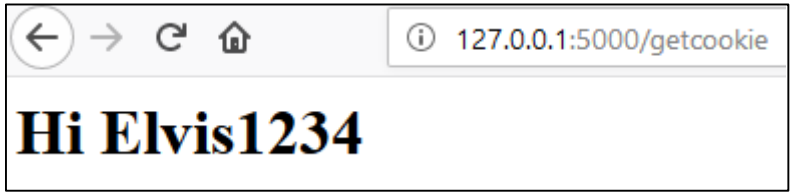
Flask: cookie

```
@app.route('/setcookie', methods=['POST','GET'])
def setcookie():
    if request.method=='POST':
        user = request.form['nm']
        resp = make_response(render_template('readcookie.html'))
        resp.set_cookie('userID', user)
        return resp
    else:
        return redirect(url_for('error', mess='SAMO POST')) # mala igra
```

```
@app.route('/getcookie')
def getcookie():
    uid = request.cookies.get('userID')
    return '<h1> Hi '+uid+ '</h1>'

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

Ako bi metod slanja forme bio GET:





Flask: session

- Sesija (`session`) predstavlja interval između trenutka prijave klijenta na server i trenutka odjave klijenta sa servera.
- Podaci sesije (session) se čuvaju na serveru .
- Svakoj sesiji se dodeljuje jedinstven Session ID potreban za enkripciju, tako da Flask aplikacija mora imati definisan `secret_key`.
- Objekt sesije je rečnik sesijskih promenljivih i pripadnih vrednosti.

```
from flask import Flask, session, request, redirect, url_for
import random
app = Flask(__name__)
lstr=list('interesantno')
random.shuffle(lstr)
app.secret_key = str(lstr)

@app.route('/')
def index():
   if 'username' in session:
      username = session['username']
      return 'Logged in as ' + username + '<br>' + \
         "<b><a href = '/logout'>click here to log out</a></b>"
   return "You are not logged in <br><a href = '/login'></b>" + \
      "click here to log in</b></a>"
```

Flask: session

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        session['username'] = request.form['username']
        return redirect(url_for('index'))
    return '''
    <form action = "" method = "post">
     <p><input type = text name = username></p>
     <p><input type = submit value = Login></p>
    </form>
    '''

@app.route('/logout')
def logout():
    # uklanja ulaz username
    # iz session recnika
    session.pop('username', None)
    return redirect(url_for('index'))

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

Flask: message flashing

- Metoda `flash(message, category)` kreira poruku vezanu za sesiju koja se može dohvatiti u templete-u sledećem zahtevu:
  - `message` poruka koja će se proslediti
  - `category` opcionalno može biti: `'error'`, `'info'`, `'warning'`
- Za uklanjanje poruke iz sesije u templejtu se poziva funkcija:

  `get_flashed_messages(with_categories, category_filter)`

  oba parametra su opcionalni,
  - `with_categories=true` vraća uređene parove poruka - kategorija
  - drugi parametar je filter za prikaz poruke npr. `category_filter=["error"]` za poruke kategorije error
- Ideja je npr. sledeća:
  - korisniku se prikazuje forma za logovanje (stranica /login)
  - korisnik nije uneo korektnu lozinku, a poslao je formu
  - kod na stranici obrade proverava prosleđenu formu, nalazi da je lozinka netačna, te generiše flash poruku i ponovo poziva stranicu /
  - sada stranica / (template index.html) proverava da li je generisana flash poruka i ispisuje je pre ispisa linka ka stranici /login.

Flask: message flashing

```
from flask import Flask, flash, redirect, render_template, request,
url_for

app = Flask(__name__)
app.secret_key = 'random string'

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

@app.route('/login', methods=['GET', 'POST'])
def login():
    error = None
    if request.method == 'POST':
        if request.form['username'] != 'admin' or \
           request.form['password'] != 'pass':
            error = 'Invalid username or password. Please try again!'
        else:
            flash('You were successfully logged in')
            return redirect(url_for('index'))
    return render_template('login.html', error=error)

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

Flask: message flashing

- Stranica /, view funkcija index() renderuje index.html

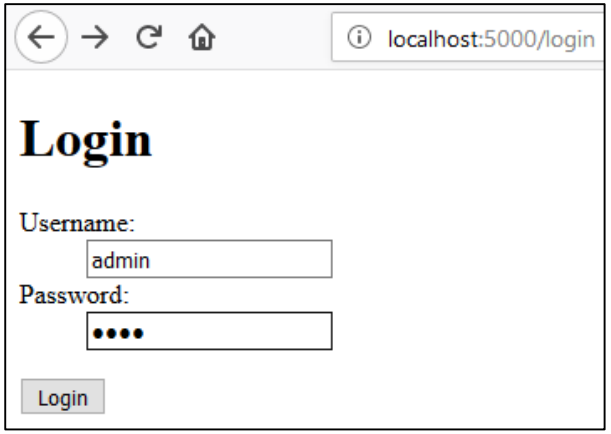
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
     {% with messages=get_flashed_messages() %}
         {% if messages %}
            <ul>
               {% for message in messages %}
               <li>{{ message }}</li>
               {% endfor %}
            </ul>
         {% endif %}
       {% endwith %}

       <h1>Flask Message Flashing Example</h1>
       <p>Do you want to <a href = "{{ url_for('login') }}">
          <b>log in?</b></a></p>
</body>
</html>
```

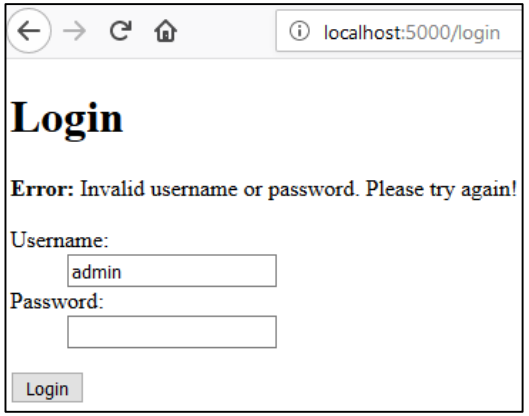
Šta biste izmenili/dodali ?

Flask: message flashing

- Template login.html prikazuje flash poruke (ako ih ima) i omogućuje unos korisničkog imena i lozinke i slanje tih podataka forme.

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="UTF-8"><title>Title</title>
</head><body>
  <h1>Login</h1>
    {% if error %}
      <p><strong>Error:</strong> {{ error }}
    {% endif %}
    <form action = "" method = post>
         <dl>
            <dt>Username:</dt>
            <dd>
               <input type = text name = username
                  value = "{{request.form.username }}">
            </dd>
            <dt>Password:</dt>
            <dd><input type = password name = password></dd>
         </dl>
         <p><input type = submit value = Login></p>
      </form>
</body></html>
```





Flask: file uploading

- Da bi se upload-ovao file u html formi:
  - postavlja se atribut forme za enkripciju podataka pri prenosu: **enctype="multipart/form-data"** (ranije rađeno)
  - postavlja se atribut forme za metod prenosa :**method**="POST"
  - postavlja se element input čiji je **type**="**file**".
- URL će dohvatiti file koristeći `request.files[]`
  - Napomena: na serveru se dospeli fajl snimi na privremenu lokaciju pre nego što se snimi na željenu lokaciju.
- Preporuka je da se naziv file-a umesto preko `request.files[file]` i svojstva `filename` koristi metoda `secure_filename()`.
- U flask-ovoj aplikaciji se postavljaju sledeći parametri:
  - `app.config['UPLOAD_FOLDER']` : folder gde će se snimati fajlovi
  - `app.config['MAX_CONTENT_LENGTH']` : maksimalna veličina fajla (bajtovi)
- U sledećem primeru se na stranici `/upload` renderuje forma uploadfile.html tako da se može odabrati file koji se upload-uje.

  Sada se odabrani fajl može proslediti na stranicu `/uploader` koje je zadužena za snimanje fajla u folder **`/uploadfiles`** pod istim osnovnim nazivom fajla.

Flask: file uploading

- Na serverskoj strani formirati u aplikacijskom folderu folder **/uploadfiles**
- Template `uploadfile.html` proverava da li postoje flash-ovane poruke u sesiji, te se iste i prikazuju (npr. nedozvoljena ekstenzija fajla), a onda se prikazuje forma koja omogućuje izbor fajla za upload i slanje fajla na server (stranica - uploader).

```
<!DOCTYPE html>
<html lang="en"> <head><meta charset="UTF-8"><title>Title</title></head>
<body>
    {% with messages = get_flashed_messages() %}
        {% if messages %}
            <ul>
                {% for message in messages %}
                    <li>{{ message }}</li>
                {% endfor %}
            </ul>
        {% endif %}
    {% endwith %}
    <form action = "http://localhost:5000/uploader" method = "POST"
        enctype = "multipart/form-data">
        <input type = "file" name = "file" />
        <input type = "submit"/>
    </form></body></html>
```

Flask: file uploading

- Na stranici /upload se prikazuje renderovani templejt opisan na prethodnoj stranici. Forma submit-om prosleđuje odabrani fajl na stranicu /uploader gde se vrše provere:
  - ekstenzija fajla
  - dužina fajla
  - slanje forme bez izabranog fajla

  te snima prihvaćeni fajl u za to predviđen folder uz zaštitu od napada XSS (Cross-Site Scripting).

```
import os
import random
from flask import Flask, flash, render_template, request, redirect,
url_for
from werkzeug.utils import secure_filename

ALLOWED_EXTENSIONS = ['txt', 'pdf', 'png', 'jpg', 'jpeg', 'gif']

app = Flask(__name__)
lstr=list('interesantno')
random.shuffle(lstr)
app.secret_key = str(lstr)
```

Flask: file uploading

```
app.config['UPLOAD_FOLDER']= './uploadfiles/'
app.config['MAX_CONTENT_LENGTH']= 10*1024*1024

def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

@app.route('/upload')
def upload_file():
    return render_template('uploadfile.html')

@app.route('/uploader', methods=['POST'])
def uploader():
    if request.method == 'POST':
        # postoji li deo file
        if 'file' not in request.files:
            flash('No file part')
            return redirect(url_for('upload_file'))
        f = request.files['file']
        # korisnik nije odabrao fajl,
        # browser salje prazan deo bez naziva fajla
        if f.filename == '':
            flash('No selected file')
            return redirect(url_for('upload_file'))
```

Flask: file uploading

```
        if allowed_file(f.filename):
            f.save(os.path.join(app.config['UPLOAD_FOLDER'],
                                secure_filename(f.filename)))
            return 'file uploaded successfully'
        else:
            flash('File extension not allowed.')
            return redirect(url_for('upload_file'))
if __name__ == '__main__':    app.run(debug=True)
```