Code Academy

1 LYGIS

# 18 paskaita. Flask (įžanga)

2023

Aurimas Aleksandras Nausėdas

# Šiandien išmoksite

01 Sukurti Flask aplikaciją

02 Sukurti duomenų bazę susietą su Flask aplikacija

03 Atlikti CRUD veiksmus naudojant Flask biblioteką

# Python web framework'as - Flask

Flask yra populiariausias python'o microframework'as. Jeigu projektas nėra labai didelis, arba tiesiog norime pasidaryti kažkokį GUI per naršyklę, kažką greitai prototipuoti, Flask yra labai geras pasirinkimas. Su flask yra pakankamai paprasta gaminti API's.

```python
from flask import Flask

app = Flask(__name__)

@app.route("/")
def home():
    return "<h1>Čia mano naujas puslapis</h1>"

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

# Kaip sukurti minimalią svetainę

```python
from flask import Flask

app = Flask(__name__)

@app.route("/<name>")
def user(name):
    return f"Labas, {name}"

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

# Kaip puslapyje atvaizduoti įvestą kintamąjį

main.py failas:

```python
from flask import Flask, render_template

app = Flask(__name__)

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

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

templates/index.html failas:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Mano Puslapis</title>
</head>
<body>
<h1>Labas, pasauli!</h1>
</body>
</html>
```

# Kaip sukurti ir panaudoti HTML šabloną

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Skaičiavimai</title>
</head>
<body>
<h1>Skaičiavimai: </h1>
{%for x in range(10)%}
{%if x % 2 == 0 %}
<p>{{x}}</p>
{% endif %}
{%endfor%}
</body>
</html>
```

# Į šabloną galime įdėti logikos

main.py failas:

```python
from flask import Flask, render_template

app = Flask(__name__)

@app.route("/")
def home():
    vardai = ['Jonas', 'Antanas', 'Petras']
    return render_template("index.html", sarasas=vardai)

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

templates/index.html failas:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Sarašas</title>
</head>
<body>
<h1>Žmonių sarašas: </h1>
{%for vardas in sarasas%}
<p>{{vardas}}</p>
{% endfor %}
</body>
</html>
```

# Kaip kintamuosius perkelti į šabloną

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

@app.route("/login", methods=['GET', 'POST'])
def login():
    if request.method == "POST":
        vardas = request.form['vardas']
        return render_template("greetings.html", vardas=vardas)
    else:
        return render_template("login.html")


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

# Kaip perduoti duomenis iš svetainės į programą

Pavyzdys app.py faile

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Prisijungimas</title>
</head>
<body>
<form action="#" method="post">
    <p>Vardas:</p>
    <p><input type="text" name="vardas"/></p>
    <p><input type="submit" value="submit"/></p>
</form>
</body>
</html>
```

# Kaip perduoti duomenis iš svetainės į programą

Pavyzdys templates/login.html faile

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Sveikiname</title>
</head>
<body>
<h1>{{vardas}}, sveikiname prisijungus!</h1>
</body>
</html>
```

# Kaip perduoti duomenis iš svetainės į programą

Pavyzdys templates/greetings.html faile

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Mano puslapis</title>
</head>
<body>
<div>
  <h1>Sveiki,</h1>
  <p>Čia yra mano super puslapis!</p>
</div>
<div class="container">
  {% block content %}{% endblock %}
</div>
<body>
</html>
```

# Naudojame base.html šabloną

Pavyzdys templates/base.html faile

```
{% extends "base.html" %}
{% block content %}
<form action="#" method="post">
    <p>Vardas:</p>
    <p><input type="text" name="vardas"/></p>
    <p><input type="submit" value="submit"/></p>
</form>
{% endblock %}
```

# Naudojame base.html šabloną

Pavyzdys templates/login.html faile

```
{% extends "base.html" %}
{% block content %}
<form action="#" method="post">
    <p>Vardas:</p>
    <p><input type="text" name="vardas"/></p>
    <p><input type="submit" value="submit"/></p>
</form>
{% endblock %}
```

# Naudojame base.html šabloną

Pavyzdys templates/greetings.html faile

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Mano puslapis</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"></script>
</head>
<body>
<!-- Navigation -->
<nav class="navbar navbar-expand-lg navbar-light bg-light static-top mb-5 shadow">
  <div class="container">
    <a class="navbar-brand" href="#">
    Mano puslapis</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarResponsive"
            aria-controls="navbarResponsive" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
    <div class="collapse navbar-collapse" id="navbarResponsive">
      <ul class="navbar-nav ml-auto">
        <li class="nav-item">
          <a class="nav-link" href="/login">Prisijungti</a>
        </li>
      </ul>
    </div>
  </div>
</nav>
<div class="jumbotron text-center">
  <h1>Sveiki,</h1>
  <p>Čia yra mano super puslapis!</p>
</div>

<div class="container">
{% block content %}{% endblock %}
</div>

</body>
</html>
```

# Stiliui panaudojame "Bootstrap"

Pavyzdys templates/base.html faile

```python
import os
from flask import Flask
from flask_sqlalchemy import SQLAlchemy

basedir = os.path.abspath(os.path.dirname(__file__))
# pilnas kelias iki šio failo.
app = Flask(__name__)

app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///'+os.path.join(basedir, 'data.sqlite')
# nustatėme, kad mūsų duomenų bazė bus šalia šio failo esants data.sqlite failas
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
# neseksime kiekvienos modifikacijos
db = SQLAlchemy(app)
# sukuriame duomenų bazės objektą
# sukurkime modelį užklausos formai, kuris sukurs duomenų bazėje lentelę


class Query(db.Model):
    # DB lentelei priskiria pavadinimą, jei nenurodysite, priskirs automatiškai pagal klasės pavadinimą.
    __tablename__ = 'messages'
    id = db.Column(db.Integer, primary_key=True)  # stulpelis, kurio reikšmės integer. Taip pat jis bus primary_key.
    name = db.Column(db.String(80), nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)
    message = db.Column(db.Text, nullable=False)

    def __init__(self, name, email, message):
        self.name = name
        self.email = email
        self.message = message

    def __repr__(self):
        return f'{self.name} - {self.email}'
```

# Duomenų bazės sukūrimas

Flask leidžia mums dirbti su duomenų bazėmis, praktiškai nesitepant rankų į SQL užklausas. Viskuo pasirūpina modulis Flask-SQLAlchemy. Iš principo tai yra SQLAlchemy, optimizuota flaskui. Diegiasi *pip install Flask-SQLAlchemy*.

```python
from app import db, Message

db.create_all()  # sukurs mūsų lentelę DB

# Iš karto inicijuosime testams keletą įrašų:
jonas = Message('Jonas', 'jonas@mail.com', 'Kažkoks labai rimtas atsiliepimas.')
antanas = Message('Antanas', 'antanas@mail.lt', 'Antano nuomonė labai svarbi.')
juozas = Message('Juozas', 'juozukas@friends.lt', 'Aš labai piktas, nes blogai.')
bronius = Message('Bronius', 'bronka@yahoo.com', 'Aš tai linksmas esu, man patinka.')

# Pridėsime šiuos veikėjus į mūsų DB
db.session.add_all([jonas, antanas, juozas, bronius])
# .commit išsaugo pakeitimus
db.session.commit()

print(jonas.id)
print(antanas.id)
print(bronius.id)
print(juozas.id)

# 1
# 2
# 4
# 3
```

# Duomenų bazės įrašo sukūrimas

Sukuriame naują failą query.py

```python
from app import db, Message

db.create_all()  # sukurs mūsų lentelę DB

# Iš karto inicijuosime testams keletą įrašų:
jonas = Message('Jonas', 'jonas@mail.com', 'Kažkoks labai rimtas atsiliepimas.')
antanas = Message('Antanas', 'antanas@mail.lt', 'Antano nuomonė labai svarbi.')
juozas = Message('Juozas', 'juozukas@friends.lt', 'Aš labai piktas, nes blogai.')
bronius = Message('Bronius', 'bronka@yahoo.com', 'Aš tai linksmas esu, man patinka.')

# Pridėsime šiuos veikėjus į mūsų DB
db.session.add_all([jonas, antanas, juozas, bronius])
# .commit išsaugo pakeitimus
db.session.commit()

print(jonas.id)
print(antanas.id)
print(bronius.id)
print(juozas.id)

# 1
# 2
# 4
# 3
```

# Duomenų bazės įrašo sukūrimas

Sukuriame naują failą query.py

```
from app import db, Message

all_messages = Message.query.all()
print(all_messages)

#  [Jonas - jonas@mail.com, Antanas - antanas@mail.lt, Juozas - juozukas@friends.lt, Bronius - bronka@yahoo.com]
```

```
message_1 = Message.query.get(1)
print(message_1)

# Jonas - jonas@mail.com
```

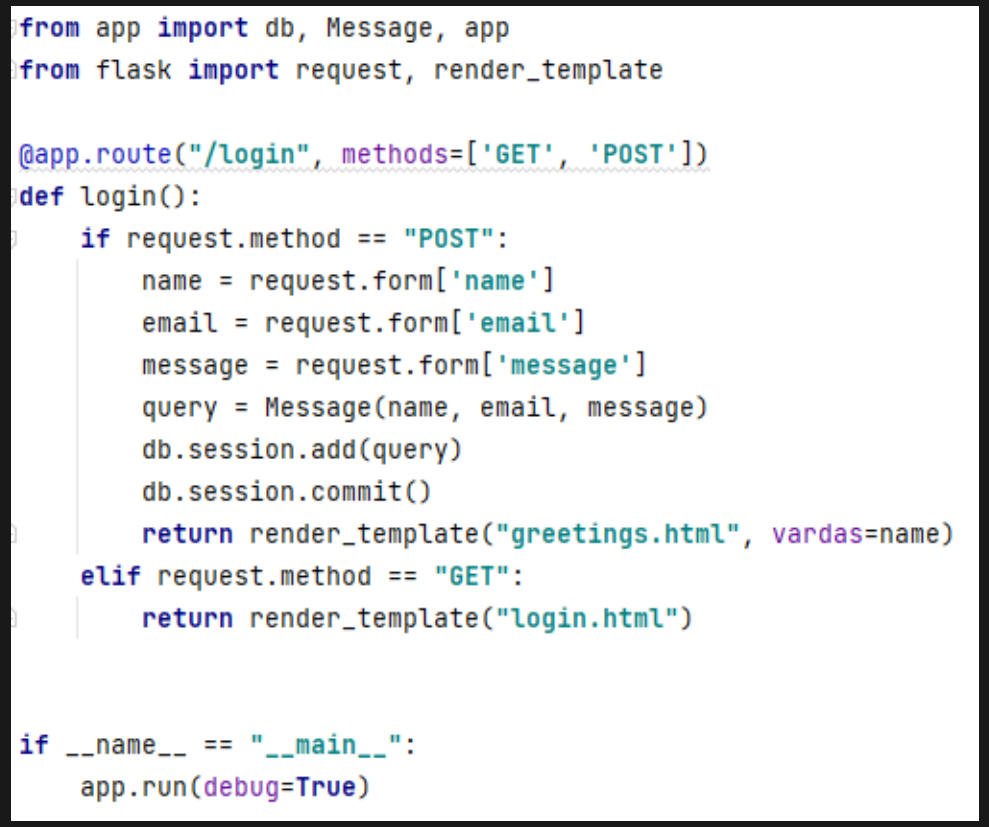
```
message_antanas = Message.query.filter_by(name='Antanas')
print(message_antanas.all())

# [Antanas - antanas@mail.lt]
```

```
antanas = Message.query.get(2)
antanas.email = 'geras.zmogus@lrs.lt'
db.session.add(antanas)
db.session.commit()
print(Message.query.all())

# [Jonas - jonas@mail.com, Antanas - geras.zmogus@lrs.lt, Juozas - juozukas@friends.lt, Bronius - bronka@yahoo.com]
```

```
jonas = Message.query.get(1)
db.session.delete(jonas)
db.session.commit()
print(Message.query.all())

# [Antanas - geras.zmogus@lrs.lt, Juozas - juozukas@friends.lt, Bronius - bronka@yahoo.com]
```

# Paprastos CRUD operacijos

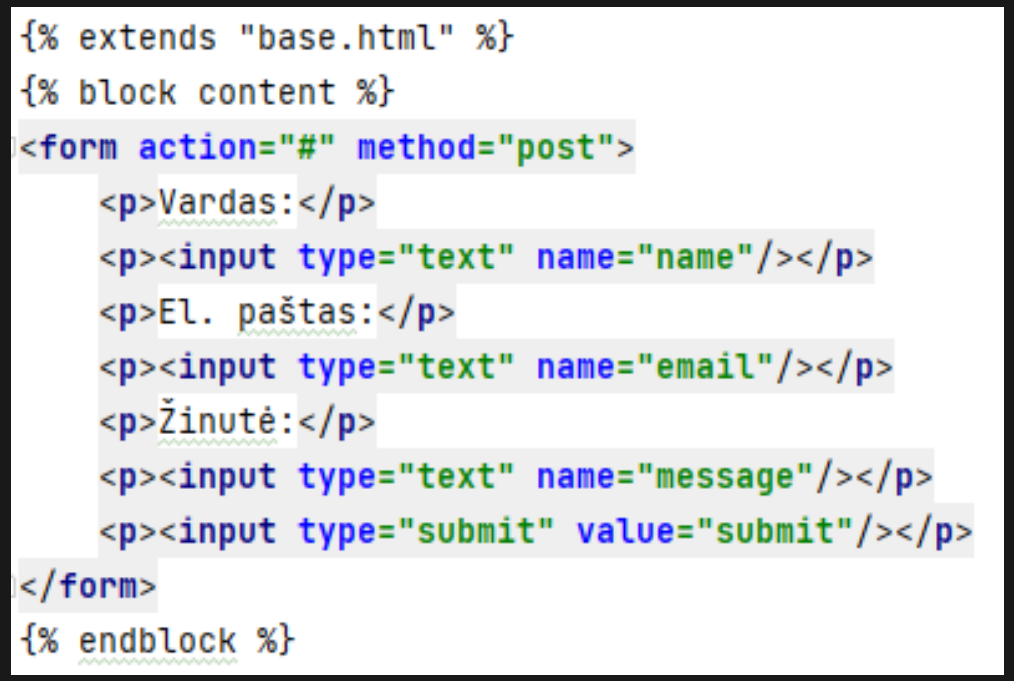
Sukuriame naują failą crud.py ir jame išmėginame CRUD operacijas



```python
from app import db, Message, app
from flask import request, render_template

@app.route("/login", methods=['GET', 'POST'])
def login():
    if request.method == "POST":
        name = request.form['name']
        email = request.form['email']
        message = request.form['message']
        query = Message(name, email, message)
        db.session.add(query)
        db.session.commit()
        return render_template("greetings.html", vardas=name)
    elif request.method == "GET":
        return render_template("login.html")


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

# Programa su duomenų įrašymu į DB

Pavyzdys main.py faile



```
{% extends "base.html" %}
{% block content %}
<form action="#" method="post">
    <p>Vardas:</p>
    <p><input type="text" name="name"/></p>
    <p>El. paštas:</p>
    <p><input type="text" name="email"/></p>
    <p>Žinutė:</p>
    <p><input type="text" name="message"/></p>
    <p><input type="submit" value="submit"/></p>
</form>
{% endblock %}
```

# Programa su duomenų įrašymu į DB

Pavyzdys templates/login.html faile

## Užduotis nr. 1

Sukurti programą, kuri turėtų statinį puslapį, pvz. localhost:5000 su norimu tekstu (rekomenduojama naudoti šablonus)

## Užduotis nr. 2

Sukurti programą, kuri įvedus norimą žodį adreso eilutėje (po / simbolio) ir paspaudus ENTER, atspausdintų jį penkis kartus

## Užduotis nr. 3

Sukurti programą, kuri puslapyje localhost:5000/keliamieji parodytų visus keliamuosius metus nuo 1900 iki 2100 metų

## Užduotis nr. 4

Sukurti programą, kuri leistų įvesti metus ir paspaudus patvirtinimo mygtuką parodytų, ar jie yra keliamieji

## Namų darbas

Užbaigti klasėje nepadarytas užduotis

**Išspręsti paskaitos uždaviniai** (įkelti pirmadienį) https://github.com/aurimas13/Python-Beginner-Course/tree/main/Programs

# Naudinga informacija