



# Zarządzanie Danymi

*Flask oraz Bazy danych*

**dr inż. Łukasz Piątek**

Katedra Sztucznej Inteligencji

Wyższa Szkoła Informatyki i Zarządzania z siedzibą w Rzeszowie

# Agenda



- Implementacja przykładowej aplikacji bazodanowej:
  - prosta aplikacja *Flask* (z *Flask\_SQLAlchemy*) – umożliwiająca obsługę *CRUD* (ang., *Create*, *Read*, *Update* oraz/lub *Delete*)
  - Baza danych *sqlite*,
  - Środowisko programistyczne *Microsoft VS Studio Code*.

# Podstawowa aplikacja *Flask DB* (1)



## ■ Utworzenie środowiska (*Anaconda Prompt*):

Anaconda Prompt

```
(base) C:\Users\lpiatek>conda create --name flask_databases_env python=3.9
```

## ■ Instalacja bibliotek (*Flask*, *Flask-SQLAlchemy*):

Anaconda Prompt

```
(flask_databases_env) C:\Users\lpiatek>pip install Flask_
```

Anaconda Prompt

```
(flask_databases_env) C:\Users\lpiatek>pip install Flask-SQLAlchemy
```

# Podstawowa aplikacja *Flask DB* (2)



## ■ *Microsoft VS Studio Code*:

```
basicmodel.py ×
basicmodel.py > Student
1  from flask import Flask
2  from flask_sqlalchemy import SQLAlchemy
3  import os
4
5  basedir = os.path.abspath(os.path.dirname(__file__))
6
7  app = Flask(__name__)
8  app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:/// ' + os.path.join(basedir, 'data.sqlite')
9  app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
10
11 db = SQLAlchemy(app)
12
13 class Student(db.Model):
14
15     __tablename__ = 'students'
16
17     id = db.Column(db.Integer, primary_key=True)
18     name = db.Column(db.Text)
19     semester = db.Column(db.Integer)
20
21     def __init__(self, name, semester):
22         self.name = name
23         self.semester = semester
24
25     def __repr__(self):
26         return f'Student "{self.name}" is studying at the "{self.semester}" semester.'
```

**Rys.1.** Implementacja prostej aplikacji *Flask DB* w środowisku *Microsoft VS Studio Code* (*basicmodel.py*)

# Podstawowa aplikacja *Flask DB* (3)



## ■ *Microsoft VS Studio Code*:

```
setupdatabase.py X
setupdatabase.py > ...
1  from basicmodel import app, db, Student
2  |
3  app.app_context().push()
4
5  db.create_all()
6
7  kamila = Student('Kamila', 6)
8  piotr = Student('Piotr', 5)
9
10 print(kamila.id)
11 print(piotr.id)
12
13 # Adding the list of objects (Students):
14 db.session.add_all(
15 |     [kamila, piotr]
16 )
17
18 # An alternative:
19 # db.session.add(kamila)
20 # db.session.add(piotr)
21
22 db.session.commit()
23
24 print(kamila.id)
25 print(piotr.id)
26
27 print(kamila)
28 print(piotr)
```

***Rys.2.A. Implementacja prostej aplikacji *Flask DB* w środowisku *Microsoft VS Studio Code* (*setupdatabase.py*)***

# Podstawowa aplikacja *Flask DB* (3) (c.d.)



## ■ *Microsoft VS Studio Code:*

The screenshot shows the Microsoft VS Studio Code interface. In the Explorer panel on the left, the file `data.sqlite` is highlighted with a red box. The Terminal panel on the right shows the command `python setupdatabase.py` being executed, with the command itself also highlighted by a red box. The output of the script is displayed in the terminal, showing the creation of a SQLite database and the insertion of two records: Kamila (6th semester) and Piotr (5th semester). The output is also highlighted by a red box.

```
PC C:\Users\lpiatek\UWM_2023\ZarządzanieDanymi_19062023\01 flask_api_databases> python setupdatabase.py
None
None
1
2
Student "Kamila" is studying at the "6" semester.
Student "Piotr" is studying at the "5" semester.
PS C:\Users\lpiatek\UWM_2023\ZarządzanieDanymi_19062023\01 flask_api_databases>
```

***Rys.2.B. Rezultat wykonania skryptu (`setupdatabase.py`)***

# Podstawowa aplikacja *Flask DB* (4)



## ■ *Microsoft VS Studio Code (Initial DB, CREATE, READ):*

```
crud_operations.py X
crud_operations.py > ...
1  from basicmodel import app, db, Student
2
3  app.app_context().push()
4
5  # ***** (0) CHECK of Initial DB *****
6  print(f'\n***** (0) Preliminary check of INITIAL DATA in DB (from setupd
7  all_students = Student.query.all() # List of all students in a Table (in data.slite)
8  # CHECK
9  print(f'ALL STUDENTS:\n{all_students}')
10
11 # ***** (1) CREATE *****
12
13 print(f'\n***** (1) CREATE *****\n')
14 new_student = Student('Lukasz', 3)
15 db.session.add(new_student)
16 db.session.commit()
17 # CHECK
18 all_students = Student.query.all()
19 print(f'ALL STUDENTS:\n{all_students}')
20
21 # ***** (2) READ *****
22
23 print(f'\n***** (2) READ *****\n')
24 all_students = Student.query.all() # List of all students in a table
25 # ***** CHECK *****
26 print(f'ALL STUDENTS:\n{all_students}')
27
```

**Rys.3.A. Implementacja prostej aplikacji *Flask DB* w środowisku *Microsoft VS Studio Code* (*crud\_operations.py*) – funkcje *query.all*, *add***

# Podstawowa aplikacja *Flask DB* (4) (c.d.)



- Wykonanie skryptu (*Initial DB, CREATE, READ*), w tym:
  - wyświetlenie zawartości początkowej bazy danych (początkowo w pliku *setupdatabase.py*),
  - dodanie nowego *Studenta*,
  - ponowne odczytanie zawartości bazy

```
***** (0) Preliminary check of INITIAL DATA in DB (from setupdatabase.py) *****
ALL STUDENTS:
[Student "Kamila" is studying at the "6" semester., Student "Piotr" is studying at the "5" semester.]

***** (1) CREATE *****
ALL STUDENTS:
[Student "Kamila" is studying at the "6" semester., Student "Piotr" is studying at the "5" semester., Student "Lukasz" is studying at the "3" semester.]

***** (2) READ *****
ALL STUDENTS:
[Student "Kamila" is studying at the "6" semester., Student "Piotr" is studying at the "5" semester., Student "Lukasz" is studying at the "3" semester.]
```

***Rys.3.B. Rezultat wykonania skryptu (*crud\_operations.py*) – funkcje: *query.all* oraz *add****



# Podstawowa aplikacja *Flask DB* (4) (c.d.2)



## ■ *Microsoft VS Studio Code (SELECT by ID, by NAME):*

```
crud_operations.py X
crud_operations.py > ...
27
28 # ***** (3A) Select by ID *****
29 print('\n***** (3A) SELECT by ID (ex. for ID=1) *****')
30 #student_first = Student.query.get(1) - query.get - deprecated in > SQLAlchemy 1.0
31 student_first = db.session.get(Student, 1)
32 # ***** CHECK *****
33 print(f'STUDENT 1; Print "only" NAME:\n{student_first.name}')
34 print(f'STUDENT 1; Print ALL DATA:\n{student_first}')
35 all_students = Student.query.all()
36 print(f'ALL STUDENTS:\n{all_students}')
37
38 # ***** (3B) FILTERS *****
39 print('\n***** (3B) SELECT by NAME (ex. for NAME=Lukasz) *****')
40 student_lukasz = Student.query.filter_by(name='Lukasz')
41 # ***** CHECK *****
42 print(f'STUDENT (Lukasz):\n{student_lukasz.all()}')
```

***Rys.3.C. Implementacja prostej aplikacji *Flask DB* w środowisku *Microsoft VS Studio Code* (*crud\_operations.py*) – funkcje: *get* oraz *filter\_by****

# Podstawowa aplikacja *Flask DB* (4) (c.d.3)



## ■ Wykonanie skryptu (*SELECT by ID, by NAME*):

```
***** (3A) SELECT by ID (ex. for ID=1) *****
STUDENT 1; Print "only" NAME:
Kamila
STUDENT 1; Print ALL DATA:
Student "Kamila" is studying at the "6" semester.
ALL STUDENTS:
[Student "Kamila" is studying at the "6" semester., Student "Piotr" is studying at the "5" semester., Student "Lukasz" is studying at the "3" semester.]

***** (3B) SELECT by NAME (ex. for NAME=Lukasz) *****
STUDENT (Lukasz):
[Student "Lukasz" is studying at the "3" semester.]
```

***Rys.3.D. Rezultat wykonania skryptu (*crud\_operations.py*) – znalezienie Studenta z zastosowaniem funkcji: *get* oraz *filter\_by****

# Podstawowa aplikacja *Flask DB* (4) (c.d.4)



## ■ *Microsoft VS Studio Code (UPDATE, DELETE):*

```
crud_operations.py > ...
43
44 print('\n***** (4) UPDATE *****')
45 # ***** (4) UPDATE *****
46 first_student = db.session.get(Student, 1)
47 first_student.name = 'Natalia'
48 first_student.semester = 7
49 db.session.add(first_student)
50 db.session.commit()
51 # ***** CHECK *****
52 all_students = Student.query.all()
53 print(f'ALL STUDENTS:\n{all_students}')
54
55 print('\n***** (5) DELETE *****')
56 # ***** (5) DELETE *****
57 second_student = db.session.get(Student, 2)
58 db.session.delete(second_student)
59 db.session.commit()
60 # ***** CHECK *****
61 all_students = Student.query.all()
62 print(f'ALL STUDENTS:\n{all_students}')
```

**Rys.3.E. Implementacja prostej aplikacji *Flask DB* w środowisku *Microsoft VS Studio Code* (*crud\_operations.py*) – funkcje *get* / *add* oraz *delete***

# Podstawowa aplikacja *Flask DB* (4) (c.d.5)



## ■ Wykonanie skryptu (*UPDATE*, *DELETE*):

```
***** (4) UPDATE *****  
ALL STUDENTS:  
[Student "Natalia" is studying at the "7" semester., Student "Piotr" is studying at the "5" semester., Student "Lukasz" is studying at the "3" semester.]  
  
***** (5) DELETE *****  
ALL STUDENTS:  
[Student "Natalia" is studying at the "7" semester., Student "Lukasz" is studying_at the "3" semester.]
```

***Rys.3.F.* Rezultat wykonania skryptu (*crud\_operations.py*) – operacje *update* (*get / add*) oraz *delete***





**Wyższa Szkoła Informatyki i Zarządzania  
ul. Sucharskiego 2, 35-225 Rzeszów, Polska**



*Dziękuję za uwagę...*

