

Python średnio zaawansowany

Dzień 18





Blok nr 5:

Aplikacja webowa



AGENDA

- Panel administracyjny
- Uwierzytelnianie
- Autoryzacja i kontrola dostępu
- Zadanie projektowe



Panel administracyjny

Flask-Admin



Flask-Admin służy do budowania interfejsu administracyjnego dla istniejących modeli danych.

Przy niewielkim nakładzie pracy można uzyskać wygodny interfejs administracyjny realizujący CRUD.

Więcej informacji: https://flask-admin.readthedocs.io/



Flask-Admin

Flask-Admin pozwala także na budowanie skomplikowanych interfejsów poprzez grupowanie widoków w klasy.

Klasy realizujące widoki stają się bardzo przydatne po powiązaniu ich z konkretnymi modelami danych.

W takim podejściu uzyskujemy kompletny CRUD realizowany przez odpowiednie klasy dla każdego modelu (1:1).



Uruchomienie panelu administracyjnego

Minimalny kod uruchamiający interfejs administracyjny (http://localhost:5000/admin):

```
from flask import Flask
from flask_admin import Admin

app = Flask(__name__)

# utworzenie obiektu klasy Flask-Admin i przypięcie go do aplikacji
admin = Admin(app, name='Przeglądarka modeli')

app.run()
```

Dodawanie widoków modeli (Model Views)

Widoki modeli umożliwiają uzyskanie dedykowanych stron administracyjnych na potrzeby zarządzania dowolnym modelem danych.

Wykorzystuje się do tego obiekty klasy ModelView:

admin.add view(ModelView(models.Address, db.session))

```
from flask_admin.contrib.sqla import ModelView

# <-- tu wstaw kod z poprzedniej strony. Kod został usunięty aby zwiększyć
czytelność przykładu -->

# Wskazujemy, które modele (models.Client, model.Address) będą zarządzane
przez panel administracyjny

admin.add view(ModelView(models.Client, db.session))
```



CRUD

W efekcie otrzymaliśmy zestaw widoków (reprezentowane przez zakładki) realizujących **CRUD**:

- Zakładka 'Create' realizująca funkcjonalność tworzenia rekordów (C)
- Zakładka 'List' realizująca funkcjonalności:
 - wyświetlania, sortowania rekordów (R)
 - edycji rekordów (**U**)
 - usuwania rekordów (D)



Modyfikacje wyglądu

Modyfikacja wyglądu głównej strony panelu możliwa jest przez edycję pliku znajdującego się w folderze:

templates/admin/index.html

Więcej szczegółów:

https://flask-admin.readthedocs.io/en/latest/introduction/#working-withthe-built-in-templates



Modyfikacje zachowania

Istnieje modyfikacji zachowania panelu (ograniczenie CRUD dla poszczególnych modeli) przed dodanie własnych klas widoków dziedziczących po klasie ModelView.

Więcej szczegółów: https://flask-admin.readthedocs.io/en/latest/introduction/#customizing-built-in-views



Uwierzytelnianie



Uwierzytelnianie

Uwierzytelnianie (ang. authentication, niezalecana polska forma: autentykacja*):

 potwierdzenie zadeklarowanej tożsamości podmiotu biorącego w bieżącym procesie. Celem uwierzytelniania jest uzyskanie pewności, że dany podmiot jest w rzeczywistości tym, za który się podaje.

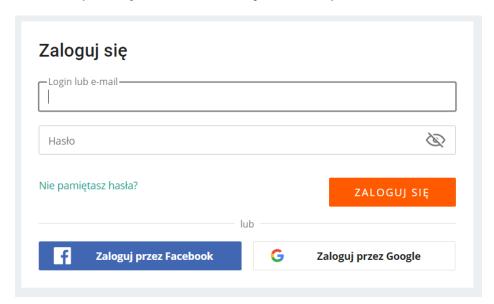
* https://pl.wikipedia.org/wiki/Uwierzytelnianie



Uwierzytelnianie

Metody uwierzytelniania:

- login + hasło (sekret) lub jego fragment
- Dwuskładnikowe (2FA): login + token (kod jednorazowy, SMS)
- Certyfikat
- Zaufana Trzecia Strona (TTP):
 - Google
 - Facebook





Autoryzacja i kontrola dostępu



Autoryzacja i kontrola dostępu

Autoryzacja (ang. authorization) - nadawanie praw dostępu do zasobu. Celem autoryzacji jest późniejsza **kontrola dostępu** (ang. access control), która potwierdza, czy dany podmiot jest uprawniony do korzystania z żądanego zasobu.

Dla określenia uprawnień danego podmiotu konieczne jest najpierw stwierdzenie jego tożsamości, dlatego w typowym zastosowaniu autoryzacja następuje dopiero po uwierzytelnieniu.



Zadanie projektowe



Zadanie projektowe

Cel:

- podłączenie się do istniejącej bazy danych ofert (pierwotnie model SQLAlchemy),
- przygotowanie panelu administracyjnego

Na kolejnych zajęciach porozmawiamy o kontroli dostępu do widoków.





Dzięki!