Python średnio zaawansowany

Dzień 15

Blok nr 5:

Aplikacja webowa

AGENDA

Architektura aplikacji internetowych

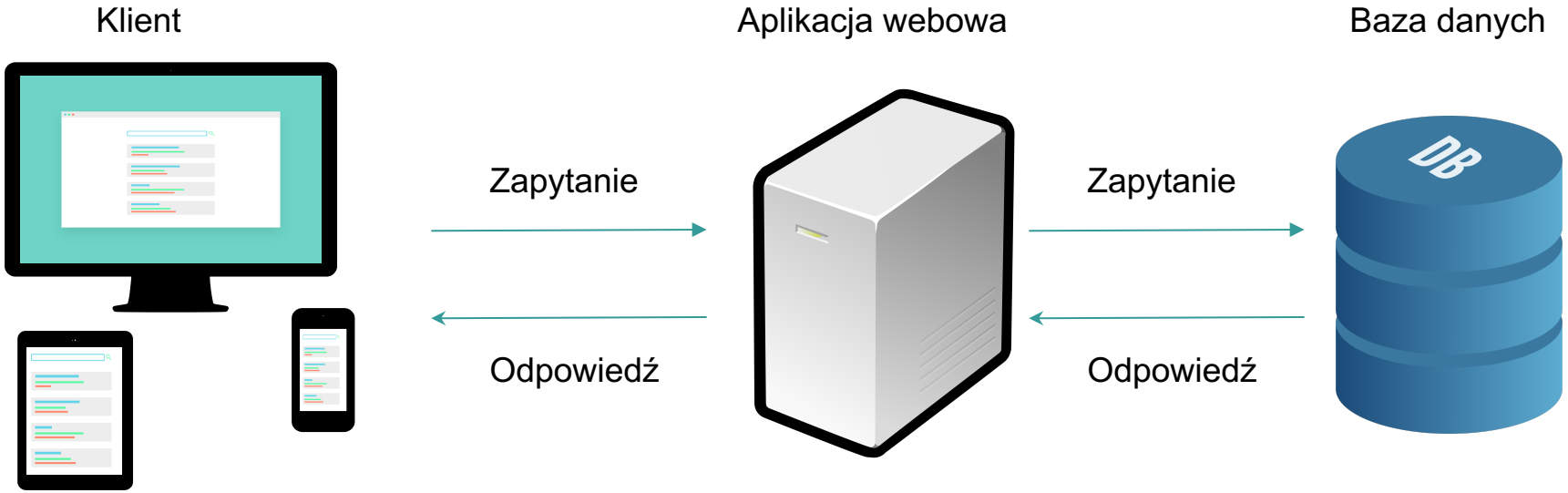
Wzorzec MVC

Język szablonów - Template language

Ankieta

Architektura aplikacji internetowych

Nieskomplikowana aplikacja internetowa



Klienci aplikacji internetowej

przeglądarka internetowa na PC

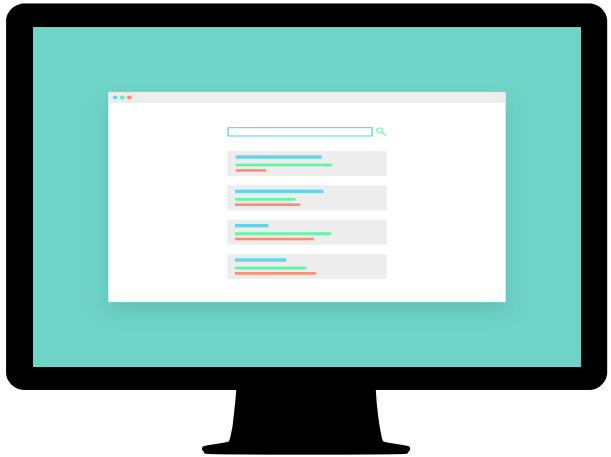
przeglądarka internetowa na urządzeniu mobilnym

dedykowana aplikacja mobilna

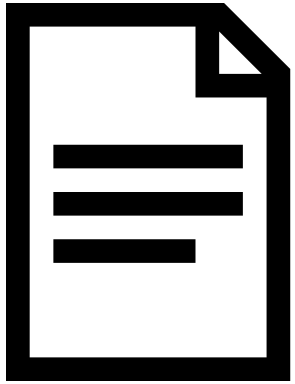
inne aplikacje komunikujące się przez REST API

Aplikacja internetowa

Frontend



Backend



Aplikacja internetowa

Frontend

HTML
CSS
Javascript

Backend

Python
Javascript
Java
PHP
Go
Ruby
C#

Wzorzec MVC

Model View Controller (MVC)

Wzorzec **MVC** zakłada podział aplikacji na trzy główne części, aby uprościć projektowanie aplikacji posiadających graficzne interfejsy użytkownika:

Model – to reprezentacja problemu bądź logiki aplikacji + dane

Widok – opisuje, jak wyświetlić pewną część modelu w ramach interfejsu użytkownika; może składać się z podwidoków odpowiedzialnych za mniejsze części interfejsu.

Kontroler – przyjmuje dane wejściowe od użytkownika i reaguje na jego poczynania, zarządzając aktualizacje modelu oraz odświeżenie widoków.

Model View Controller

Ze względu na ilość elementów składowych aplikacji webowej, logika biznesowa powinna znajdować się w kodzie w ustalonym miejscu - najlepiej w modelach, podążając w myśl idei “fat models, thin views”.

Ilość kodu odpowiedzialnego za logikę biznesową w poszczególnych elementach

Model > Kontroler > Widok

Framework

Definicja frameworka

Framework to szkielet do budowy aplikacji:

definiuje jej strukturę i mechanizm działania,

dostarcza zestaw komponentów i bibliotek ogólnego przeznaczenia do wykonywania określonych zadań.

Programista tworzy aplikację, rozbudowując i dostosowując poszczególne komponenty do wymagań realizowanego projektu, tworząc w ten sposób gotową aplikację.

Framework Flask

Microframework (cały kod aplikacji może znajdować się w jednym pliku!)

Nieskomplikowany w konfiguracji

Posiada ogromną rzeszę użytkowników

Łatwa rozszerzalność poprzez pluginy

Strona frameworku: http://flask.pocoo.org/

Flask “Hello world!”

Flask „Hello world!”

Dobre praktyki:

- pracuj w środowisku wirtualnym (**venv**)
- w środowisku deweloperskim ustaw: `debug=True` (!)
- w środowisku produkcyjnym ustaw: `debug=False` (!)

`Praktyka: 01-03.py`

Język szablonów - Template language

Template language

Template language najczęściej jest wykorzystywany do tworzenia widoków w aplikacji webowej.

Ponieważ widok jest odpowiedzialny za to jak wyświetlić model lub pewną jego część, zatem posiadanie generycznego szablonu znacznie zredukuje ilość potrzebnego kodu do wyświetlenia **każdego** zestawu danych, za który odpowiada model.

Template language we Flasku

Framework **Flask** domyślnie wykorzystuje Jinja **(Jinja2)** jako język szablonów

Zasada działania Jinja2 polega na umieszczaniu w plikach źródłowych (html) znaczników, które następnie są zastępowane generowaną przez aplikację treścią.

System umożliwia stosowanie struktur kontrolnych: testów (if)*, pętli (for).

Strona frameworku: http://jinja.pocoo.org/

Z Jinja korzystają m.in: Mozilla, SourceForge, Instagram

Template language – przykład zastosowania

Internetowy portal ogłoszeniowy:

- strona główna z listą kategorii
- strona z listą podkategorii
- strona z listą ogłoszeń w wybranej kategorii
- strona ze szczegółami wybranego ogłoszenia

Template language – przykład zastosowania

Przepis na zbudowanie szablonu:

{{ wyrażenie }} - w tym miejscu podstawiona zostanie wartość wyrażenia

{% blok %} - tak markujemy blok, np., na potrzeby pętli, warunku; pamiętaj o zamknięciu bloku

Przykład:

`{{ abcd }}` – podstawi wartość zmiennej abcd

`{{ url_for('hello') }}` – podstawi URL wywołujący funkcję hello()

Template language – przykład zastosowania

Przepis na użycie szablonu:

zaimportuj `render_template` z modułu Flask

wywołaj metodę `render_template()`:

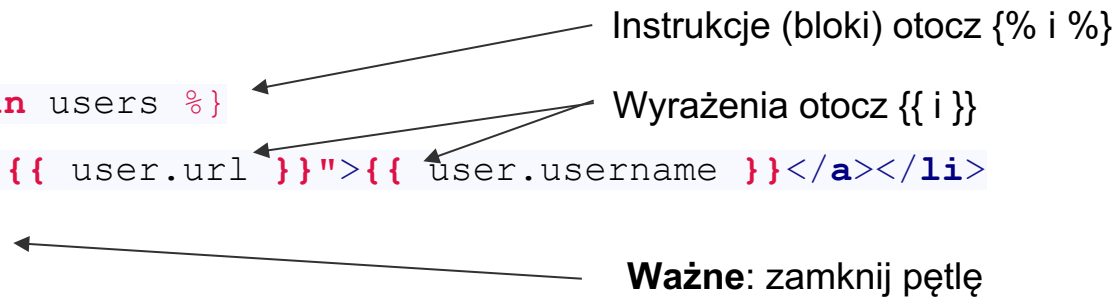
```
render_template('nazwa_pliku_szablonu.html', <kontekst>=dane)
```

Szablony znajdują się w folderze „templates” w strukturze aplikacji.

Template language – przykład zastosowania (Jinja)

Przykładowy szablon – w tym przykładzie została przekazana lista `**users**`

```
<ul>
    {% for user in users %}
    <li><a href="{{ user.url }}">{{ user.username }}</a></li>
    {% endfor %}
</ul>
```



Praktyka: 04-06.py

Template language

Template language pozwala na zapisanie prostej logiki **(warunki)** ale nie powinien być do tego wykorzystywany!

Dzięki!