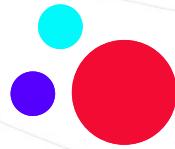


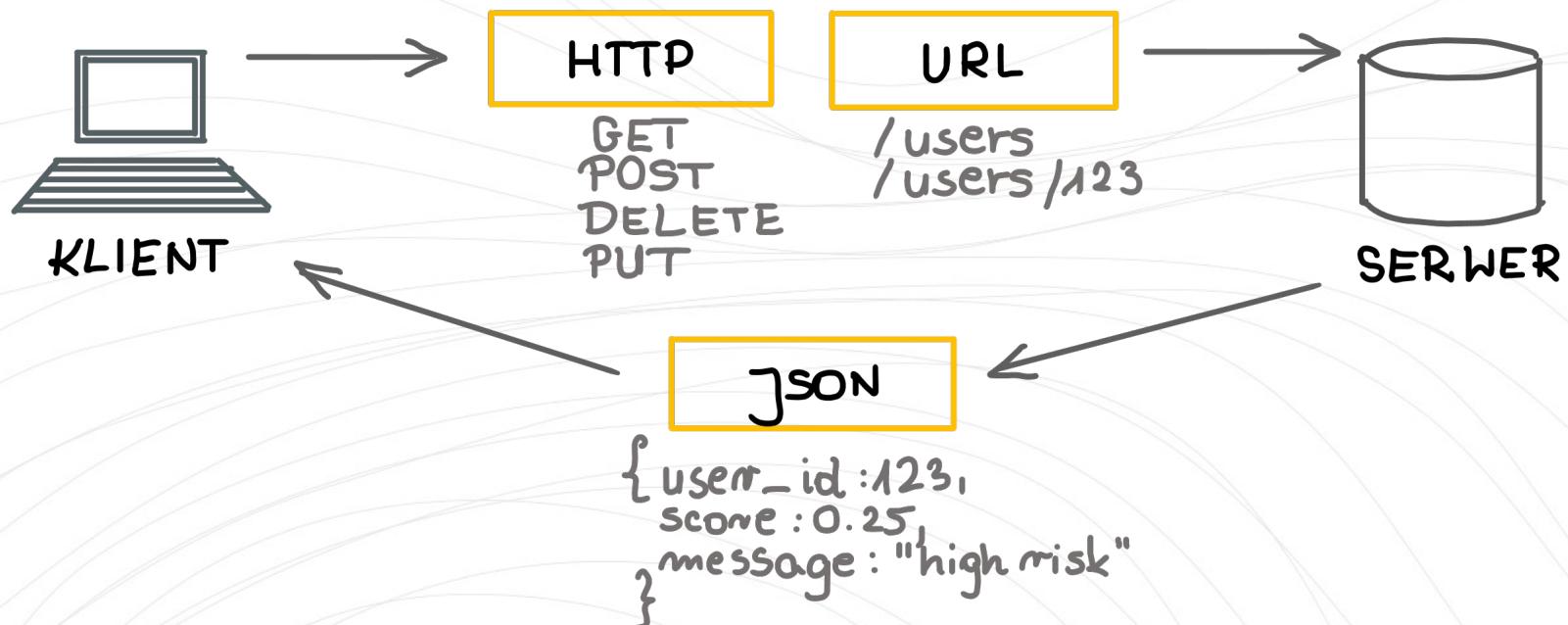
Python Podstawy - Intel

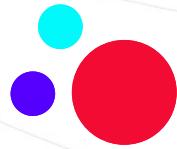
infoShare Academy



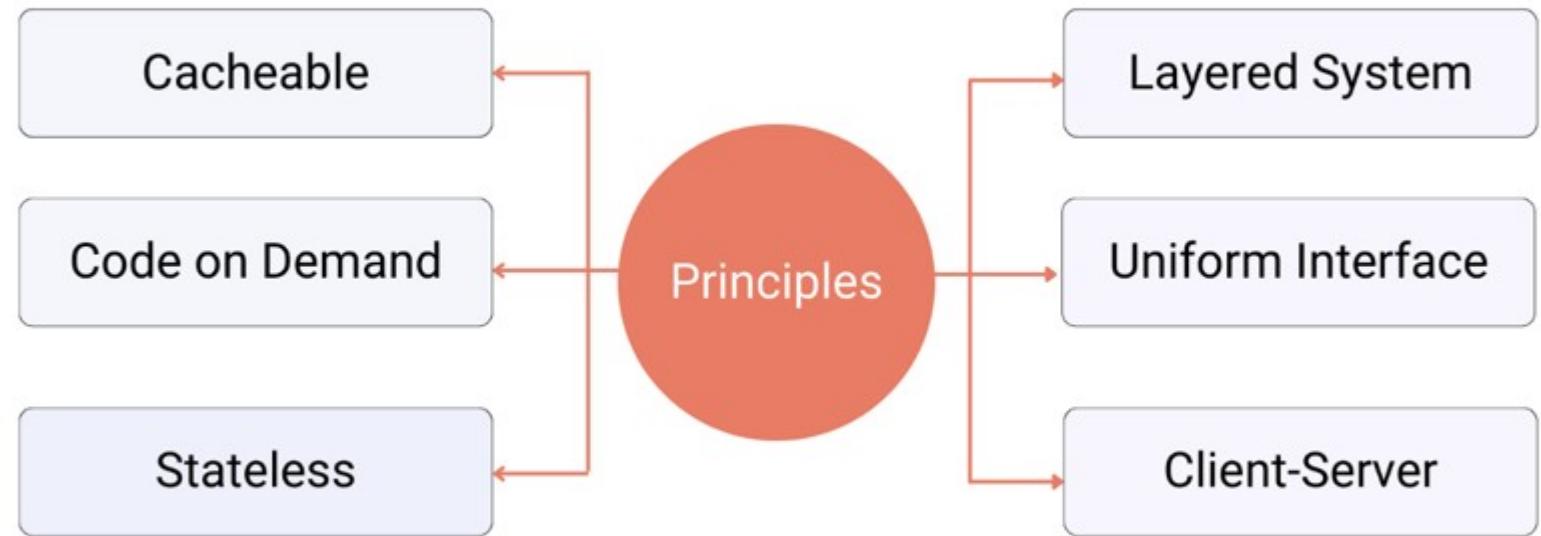
API

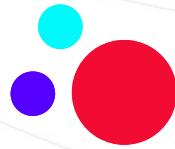
Application Programming Interface





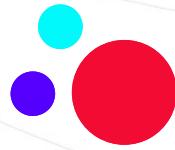
REST





JSON? XML? YML?

XML	JSON	YAML
<pre><Servers> <Server> <name>Server1</name> <owner>John</owner> <created>123456</created> <status>active</status> </Server> </Servers></pre>	<pre>{ Servers: [{ name: Server1, owner: John, created: 123456, status: active }] }</pre>	<pre>Servers: - name: Server1 owner: John created: 123456 status: active</pre>



HTTP Requests

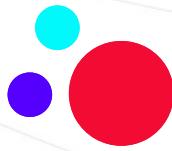
GET /customers – zwraca listę obiektów

GET /customers/1 – zwraca pojedynczy obiekt

POST /customers – tworzy nowy zasób

PUT /customers/1 – aktualizuje dany zasób

DELETE /customers/1 – usuwa zasób

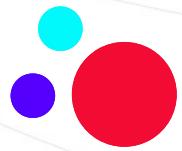


HTTP Response

HTTP Status Codes

Code	Description	Code	Description
200	OK	400	Bad Request
201	Created	401	Unauthorized
202	Accepted	403	Forbidden
301	Moved Permanently	404	Not Found
303	See Other	410	Gone
304	Not Modified	500	Internal Server Error
307	Temporary Redirect	503	Service Unavailable

```
{  
  "timestamp": "2020-02-17T09:30:00.111+0000",  
  "status": 500,  
  "error": "Internal Server Error",  
  "message": "Error processing the request!",  
  "path": "/customers"  
}
```



JSON

```
JSON Object → {  
    "company": "mycompany",  
    "companycontacts": {  
        "phone": "123-123-1234",  
        "email": "myemail@domain.com"  
    },  
    "employees": [ ← JSON Array  
        {  
            "id": 101,  
            "name": "John",  
            "contacts": [  
                "email1@employee1.com",  
                "email2@employee1.com"  
            ]  
        },  
        {  
            "id": 102, ← Number Value  
            "name": "William",  
            "contacts": null ← Null Value  
        }  
    ]  
}
```

String Value

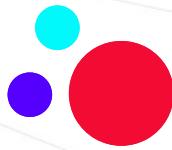
Object Inside Object

JSON Array

Array Inside Array

Number Value

Null Value



Autentykacja vs Autoryzacja

Autentykacja

kim jesteś

Autoryzacja

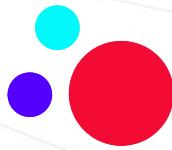
co możesz zrobić

MFA / 2FA

Coś, co wiesz (hasło, PIN)

Coś, co masz (klucz U2F, karta, telefon)

Coś, czym jesteś (odcisk palca, skan tęczówki, twarz)



Autentykacja – klucz API

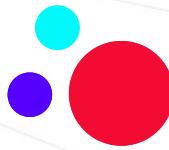
```
curl -GET https://api.example.com/endpoint -H "Authorization: api-key YOUR_API_KEY"
```

Zalety

- # proste do wdrożenia
- # łatwe do monitorowania
- # skuteczne do ograniczania ilości żądań

Wady

- # brak ważności
- # ograniczone bezpieczeństwo
- # nieodpowiednie do autoryzacji użytkowników



Autentykacja – Basic Auth

```
curl -u USERNAME:PASSWORD https://api.intel.com
```

```
curl -H "Authorization: Basic $(echo -n 'USERNAME:PASSWORD' | base64)" https://api.intel.com
```

Zalety

silniejsze bezpieczeństwo

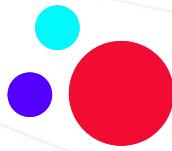
szeroka kompatybilność

prostota wdrożenia

Wady

brak ważności

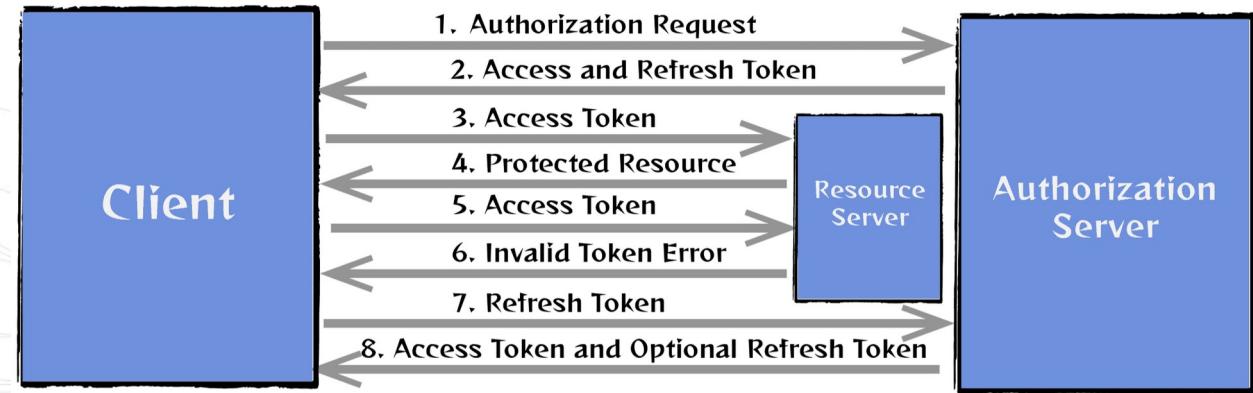
ekspozycja poświadczeń (HTTP vs HTTPS)



Autentykacja – Tokeny, OAuth, JWT

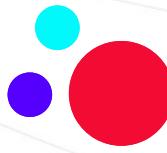
Zalety

- # małe ryzyko skompromitowania tokena
- # izolacja uprawnień
- # natychmiastowe unieważnianie



Wady

- # konieczność obsłużenia dodatkowego procesu
- # implementacja serwera autoryzacji
- # wymagany bezpieczny magazyn



Swagger

Swagger
Supported by SMARTBEAR

<https://petstore.swagger.io/v2/swagger.json> Explore

Swagger Petstore 1.0.7 OAS 2.0

[Base URL: petstore.swagger.io/v2]
<https://petstore.swagger.io/v2/swagger.json>

This is a sample server Petstore server. You can find out more about Swagger at <http://swagger.io> or on [irc.freenode.net, #swagger](#). For this sample, you can use the api key `special-key` to test the authorization filters.

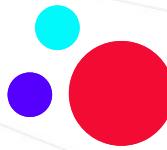
[Terms of service](#)
[Contact the developer](#)
[Apache 2.0](#)
[Find out more about Swagger](#)

Schemes [HTTPS](#) ▾ [Authorize](#)

pet Everything about your Pets

Find out more ^

POST	/pet/{petId}/uploadImage uploads an image	
POST	/pet Add a new pet to the store	
PUT	/pet Update an existing pet	
GET	/pet/findByStatus Finds Pets by status	
GET	/pet/findByTags Finds Pets by tags	



Postman

inventory by status - My Workspace

File Edit View Help

← → Home Workspaces API Network

My Workspace New Import

Collections + Search collections

Environments

History

APIs

Flows

Petstore by Fy's fork

pet

- POST add new pet
- POST upload image
- PUT update existing pet
- GET pet by status
- GET pet by id
- POST update pet
- DEL delete pet

store

- POST place an order
- GET order by id
- DEL delete order
- GET Inventory by status

user

Flows Home Overview GET inventory by status +

HTTP Petstore / store / inventory by status

GET {{petstore_url}} /store/inventory

Overview Params Authorization Headers (7) Body Scripts Settings

inventory by status

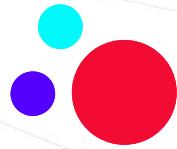
Add request description...

Set up

Variables, params, and headers

Response History





Własne API, Flask na sterydach ;)



Flask-RESTX

<https://flask-restx.readthedocs.io/>

DZIĘKUJĘ NA DZIŚ
Python Podstawy – Intel