

Nestrukturirana pohrana

Pristup podacima iz programskog koda
25/26

Ishod učenja 2



Drugi dio kolegija - obrada podataka

Projekt No.2 - **Dioniz**

- (+) Rad s nestrukturiranom (S3) pohranom
- (+) Rad s NoSQL bazama
- (+) Tekstualni podaci + TEME IZVAN ISHODA

Nestrukturirana pohrana

Generalna podjela podataka na strukturirane i nestrukturirane podatke

Nestrukturirana pohrana (ujedno i objektna pohrana)

S3 sučelje za komunikaciju s pohranom

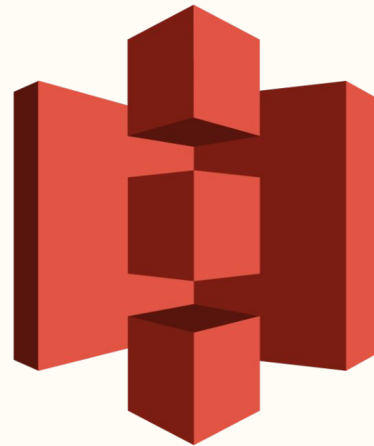
Objektna pohrana

→ upravlja, odnosno pohranjuje podatke kao **objekte**:

→ datoteke (podaci)

→ datum i vrijeme kreiranja, naziv, veličina (meta-podaci)

AWS-ov **S3** (Simple Storage Service) je utemeljeio standardizirani API za interakciju s objektima u pohrani



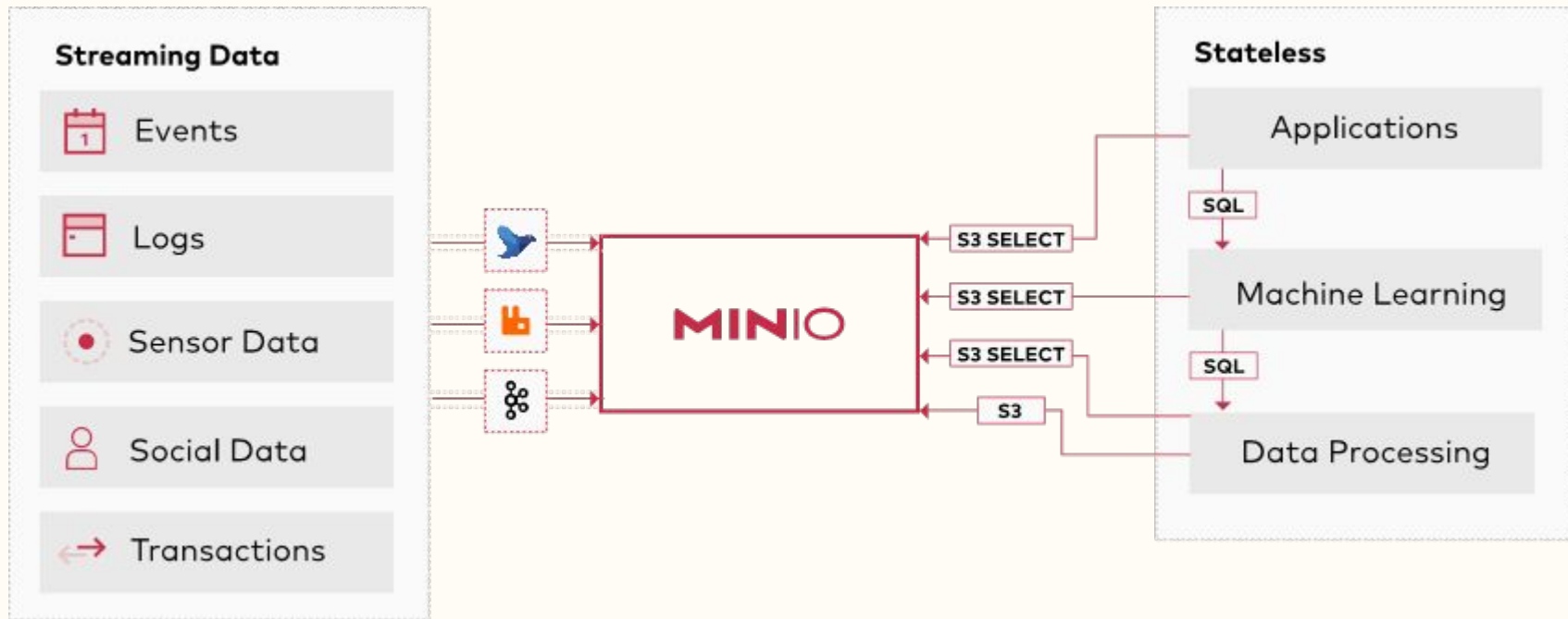
MiniO

→ MiniO je objektna pohrana otvorenog koda (eng. *open source*)

→ **S3 kompatibilna pohrana**

→ moguće ju je postaviti u različitim okruženjima (privatni server, Docker, Kubernetes...)

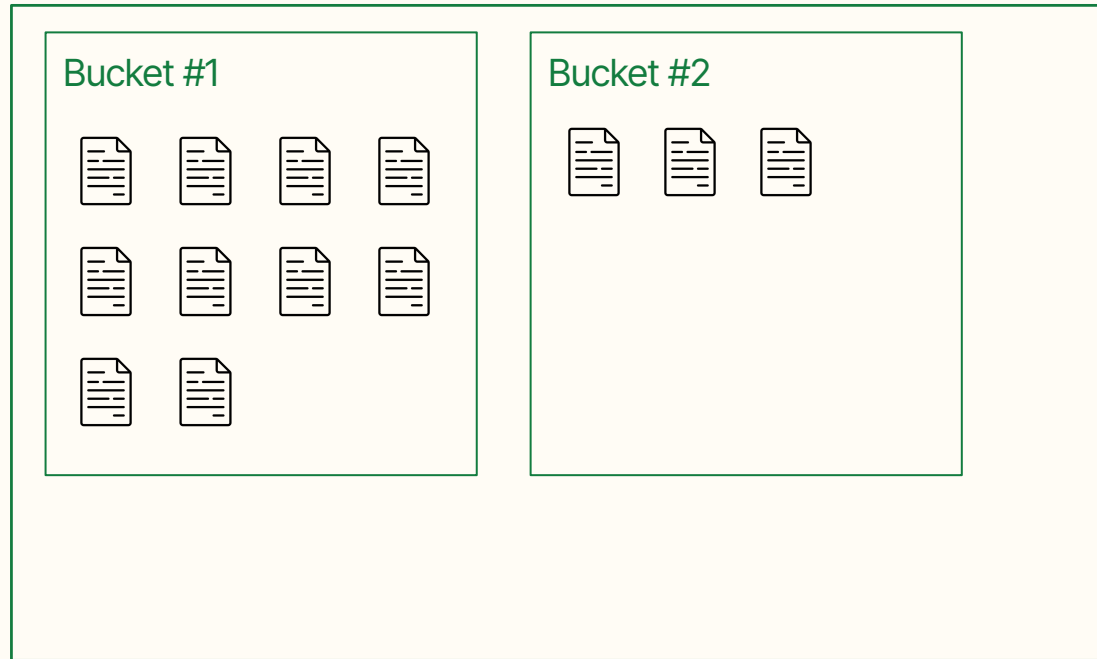
MiniO



Koncepti

Object

Bucket (kanta)



Objekt

Objekt u S3 pohrani je jedna datoteka i njeni metapodaci (vrijeme nastanka, tip sadržaja, dozvole, itd.) Objekt je definiran s tri stavke:

- (1) object data (blob) - podatak o datoteci, može biti bilo koji nestrukturirani podatak (slike, dokumenti, videi, backupi, itd.)
- (2) ključ objekta (object key) - jedinstveni identifikator objekta u bucketu
- (3) ostali metapodaci

Bucket

- grupirani objekti
- analogno mapama (direktorijima) na *lokalnom* datotečnom sustavu
- svaki bucket ima jedinstveno ime na instanci pohrane
- omogućava logičku raspodjelu podataka odnosno objekata (po projektu, odjelu, itd.)

S3 API

Buckets

- + ListBuckets
- + CreateBucket
- + DeleteBucket
- ...

Objects

- + GetObjects
- + GetObject
- + PutObject
- + DeleteObject
- ...

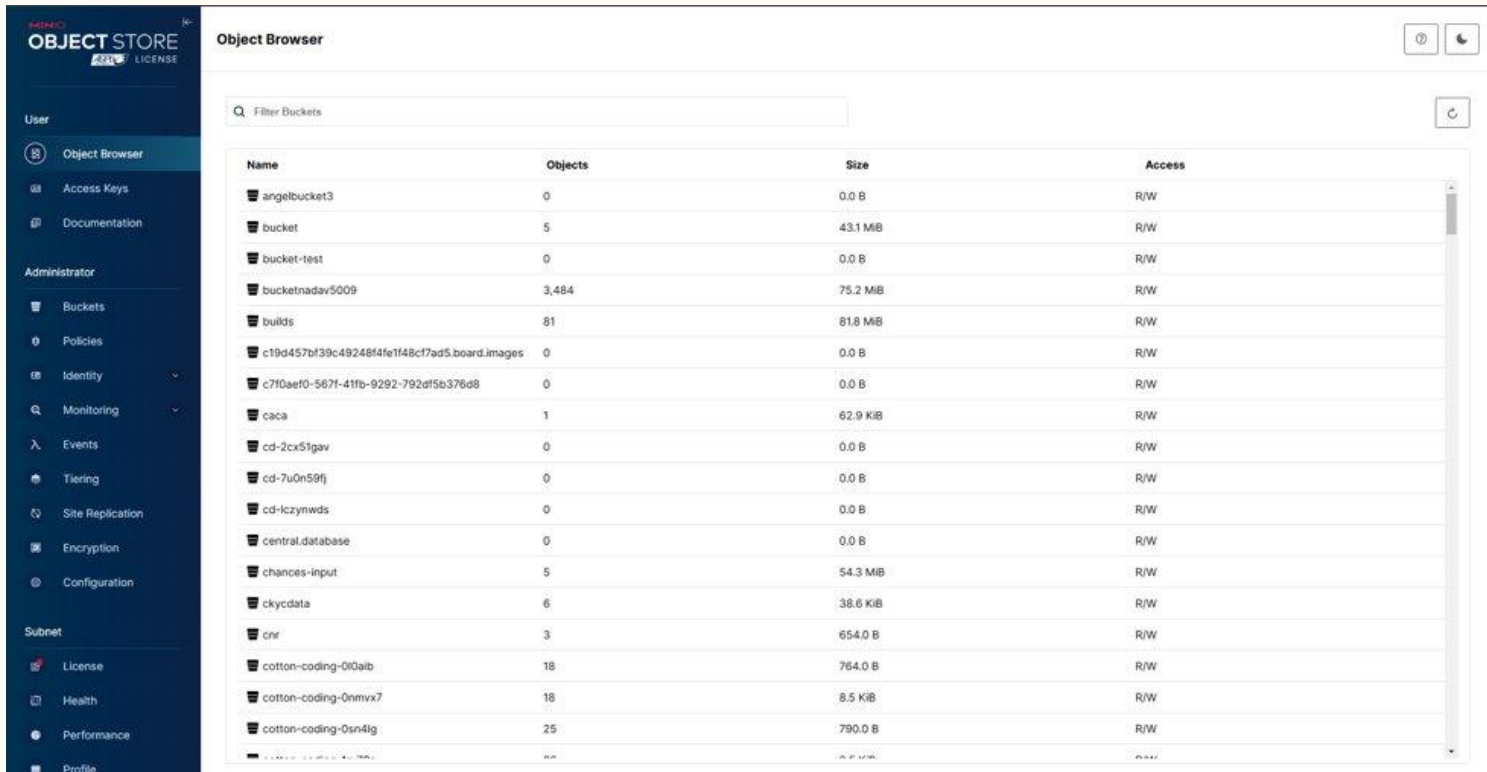
MiniO setup

- MiniO može biti postavljen kao jedinstvena ili distribuirana pohrana
- moguće je postaviti MiniO iza load balancera
- najlakše postavljanje omogućuju **kontejneri** (lokalno i u serverskom okruženju)

Postavljanje kroz Docker

```
docker run -d -p "9000:9000" -p "9001:9001"  
  --name minio  
  -e "MINIO_ROOT_USER=root"  
  -e "MINIO_ROOT_PASSWORD=password"  
quay.io/minio/minio  
server /home/shared --address :9000 --console-address :9001
```

Minio – Upravljačka ploča



The screenshot displays the Minio Object Browser interface. On the left is a dark blue sidebar with navigation links. The main area is titled 'Object Browser' and contains a table of buckets. A search bar at the top of the table is labeled 'Filter Buckets'. The table has four columns: Name, Objects, Size, and Access. The 'Access' column for all buckets shows 'R/W'. A vertical scrollbar is visible on the right side of the table.

Object Browser

Filter Buckets

Name	Objects	Size	Access
angelbucket3	0	0.0 B	R/W
bucket	5	43.1 MiB	R/W
bucket-test	0	0.0 B	R/W
bucketnada5009	3,484	75.2 MiB	R/W
builds	81	81.8 MiB	R/W
c19d457bf39c49248f4fe1f48cf7ad5.board.images	0	0.0 B	R/W
c7f0aef0-567f-41fb-9292-792df5b376d8	0	0.0 B	R/W
caCa	1	62.9 KiB	R/W
cd-2cx51gav	0	0.0 B	R/W
cd-7u0n59fj	0	0.0 B	R/W
cd-1czynwds	0	0.0 B	R/W
central.database	0	0.0 B	R/W
chances-input	5	54.3 MiB	R/W
ckycdata	6	38.6 KiB	R/W
cni	3	654.0 B	R/W
cotton-coding-0f0alb	18	764.0 B	R/W
cotton-coding-0nmvx7	18	8.5 KiB	R/W
cotton-coding-0sn4lg	25	790.0 B	R/W

Rad s Pythonom

(+) interpretirani jezik

(+) "lagan, ali spor" (?)

(+) trenutna verzija - **3.14** (pi)

Packet manager

Kao i u drugim jezicima, koristimo packet manager (pip) kako bismo instalirali dodatne pakete (biblioteke)

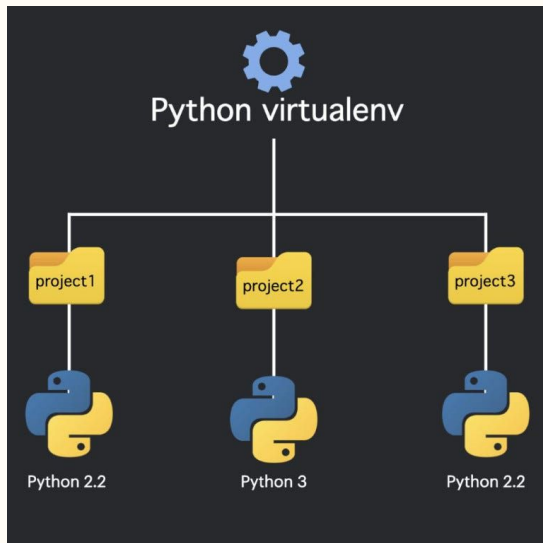


Virtualno i okruženje

Kako bismo lakše organizirali pakete i okruženje koje koristimo za Python projekt, koristit ćemo virtualno okruženje:

(+) venv

(+) conda



CONDA®

Primjer

Napravite direktorij i inicijalizirate virtualno okruženje:

```
python -m venv ./venv  
./venv/scripts/Activate
```

Aktiviranje virtualnog okruženja će postaviti putanju za izvršavanje komandi `python` i `pip`


Primjer

- (1) potrebno je instalirati boto3 paket *
- (2) potrebno je definirati sljedeće pristupne podatke:
 - + ENDPOINT
 - + ACCESS_KEY
 - + SECRET_KEY


Primjer

→ moguće je koristiti korisničko ime i lozinku admin korisnika



→ poželjnije je na upravljačkoj ploči definirati **ACCESS_KEY** i **SECRET_KEY**

 **Create Access Key**


Access Key

 aZHrwP4CsI4PKJ3h6VQU

Secret Key

Restrict beyond user policy

OFF  ON

You can specify an optional JSON-formatted IAM policy to further restrict Access Key access to a subset of the actions and resources explicitly allowed for the parent user. Additional access beyond that of the parent user cannot be implemented through these policies.

Expiry

▼

Name

Enter a name

Description

Enter a description

Comments

Enter a comment

Clear

Create

Primjer

```
s3 = boto3.client(  
    "s3",  
    endpoint_url="https://my-endpoint.example.com",  
    aws_access_key_id="ACCESS",  
    aws_secret_access_key="SECRET"  
)
```

Primjer

1 - Izradite jednostavnu skriptu koja prenosi datoteke u bucket.

2 - Izradite jednostavnu skriptu koja dohvaća podatke u bucketu

Pokušajte pročitati ulazne podatke kao i definiciju direktorija za preuzimanje kroz komandno-linijske argumente

Primjer - HINT

S3 definira sljedeće važne metode:

- + `upload_file(Filename, Bucket, Key)`
korisno kada se prenosi datoteka na lokalnoj putanji.
- + `upload_fileobj(Fileobj, Bucket, Key)`
korisno kada postoji datotečni tok (eng. *stream*), recimo prilikom web uploada
- + `download_file(Bucket, Key, Filename)`
Korisno za dohvat datoteke u lokalni `Filename`
- + `download_fileobj(Bucket, Key, Fileobj)`
Korisno za dohvat datotečno toka u `Fileobj`

Zadatak

Istražite kako biste napravili jednostavnu aplikaciju s REST API sučeljem koje omogućuje prijenos i dohvat objekata (preko jedinstvenog ključa)

Možete koristiti biblioteke poput:

- Django (djangorestframework)
- Flask
- **FastAPI**