Alkalmazásminta összeállítása

Python/Flask alkalmazás Redis adatbázis használatával

Projekt felépítése:

```
.
├── Dockerfile
├── README.md
├── app.py
├── compose.yaml
└── requirements.txt
```

```
services:
  redis:
    image: redislabs/redismod
    ports:
      - '6379:6379'
  web:
      build: .
      ports:
          - "8000:8000"
      volumes:
          - .:/code
      depends_on:
          - redis
```

Telepítés a Docker Compose segítségével

```
$ docker compose up -d
[+] Running 24/24
 ⠿ redis Pulled
 ...
   ⠿ 565225d89260 Pull complete
[+] Building 12.7s (10/10) FINISHED
 => [internal] load build definition from
Dockerfile
...
[+] Running 3/3
 ⠿ Network flask-redis_default    Created
 ⠿ Container flask-redis-redis-1  Started
 ⠿ Container flask-redis-web-1    Started
```

Várható eredmény

A listán szereplő tárolókon egy futó konténernek kell szerepelnie, és a port-leképezést az alábbiak szerint:

```
$ docker compose ps
NAME                COMMAND                  SERVICE             STATUS              PORTS
flask-redis-redis-1   "redis-server --load…"   redis               running             0.0.0.0:6379->6379/tcp
flask-redis-web-1     "/bin/sh -c 'python …"   web                 running             0.0.0.0:8000->8000/tcp
```

Az alkalmazás indítása után keressük meg a http://localhost:8000 a webböngészőben, vagy futtassuk a:

```
$ curl localhost:8000
This webpage has been viewed 2 time(s)
```

Redis események figyelése

Csatlakozzunk a redis adatbázishoz a redis-cli parancs használatával, és figyeljük a kulcsokat.

```
redis-cli -p 6379
127.0.0.1:6379> monitor
OK
1646634062.732496 [0 172.21.0.3:33106] "INCRBY" "hits" "1"
1646634062.735669 [0 172.21.0.3:33106] "GET" "hits"
```