

# Done by: Mission Shrestha \*\*\*\* 2024/07/25

## Some Additional Functionality Besides Given Task

### Caching with Redis

Implementing caching to reduce database load and improve response times.

Firstly, we install aioredis:

```
(venv) E:\1-Projects\Pokemon\Pokemon\app>pip install aioredis
Collecting aioredis
  Downloading aioredis-2.0.1-py3-none-any.whl (71 kB)
    71.2/71.2 kB 488.0 kB/s eta 0:00:00
Requirement already satisfied: async-timeout in e:\1-projects\pokemon\pokemon\venv\lib\site-packages (from aioredis) (4.0.3)
Requirement already satisfied: typing-extensions in e:\1-projects\pokemon\pokemon\venv\lib\site-packages (from aioredis) (4.12.2)
Installing collected packages: aioredis
Successfully installed aioredis-2.0.1

[notice] A new release of pip is available: 23.1.2 -> 24.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip

(venv) E:\1-Projects\Pokemon\Pokemon\app>
```

```
app >  config.py > ...
6     from dotenv import load_dotenv
7
8     load_dotenv()
9
10    DATABASE_URL = os.getenv("DATABASE_URL")
11
12    |
13    REDIS_URL = os.getenv("REDIS_URL", "redis://localhost")
14
```

Add caching to your CRUD operations (app/crud.py):

Aioredis didn't work so later changed the package to asyncio(redis)

```
app > crud.py > get_pokemon_by_name
1 # app/crud.py
2
3 import aioredis
4 from sqlalchemy.future import select
5 from sqlalchemy.ext.asyncio import AsyncSession
6 from .models import Pokemon
7 from .schemas import PokemonCreate
8 from .config import REDIS_URL
9
10 redis = aioredis.from_url(REDIS_URL)
11
12 async def get_pokemon_by_name(session: AsyncSession, name: str) -> Pokemon:
13     cache_key = f"pokemon:{name}"
14     cached_pokemon = await redis.get(cache_key)
15     if cached_pokemon:
16         return Pokemon.parse_raw(cached_pokemon)
17
18     result = await session.execute(select(Pokemon).filter(Pokemon.name == name))
19     pokemon = result.scalars().first()
20     if pokemon:
21         await redis.set(cache_key, pokemon.json(), ex=3600)
22     return pokemon
23
24 async def get_all_pokemons(session: AsyncSession, name: str = None) -> list[Pokemon]:
25     query = select(Pokemon)
26     if name:
27         query = query.filter(Pokemon.name.ilike(f"%{name}%"))
28     result = await session.execute(query)
29     return result.scalars().all()
30
31 async def create_pokemon(session: AsyncSession, pokemon: PokemonCreate) -> Pokemon:
```

## Updated working Cache:

```
app > crud.py > get_pokemon_by_name
1
2 import redis.asyncio as redis
3 from sqlalchemy.future import select
4 from sqlalchemy.ext.asyncio import AsyncSession
5 from models import Pokemon
6 from schemas import PokemonCreate
7 from config import REDIS_URL
8
9 redis_client = redis.from_url(REDIS_URL)
10
11 async def get_pokemon_by_name(session: AsyncSession, name: str) -> Pokemon:
12     cache_key = f"pokemon:{name}"
13     cached_pokemon = await redis_client.get(cache_key)
14     if cached_pokemon:
15         return Pokemon.parse_raw(cached_pokemon)
16
17     result = await session.execute(select(Pokemon).filter(Pokemon.name == name))
18     pokemon = result.scalars().first()
19     if pokemon:
20         await redis_client.set(cache_key, pokemon.json(), ex=3600)
21     return pokemon
22
23 async def get_all_pokemons(session: AsyncSession, name: str = None) -> list[Pokemon]:
24     query = select(Pokemon)
25     if name:
26         query = query.filter(Pokemon.name.ilike(f"%{name}%"))
27     result = await session.execute(query)
28     return result.scalars().all()
29
30 async def create_pokemon(session: AsyncSession, pokemon: PokemonCreate) -> Pokemon:
31     db_pokemon = Pokemon(**pokemon.dict())
32     session.add(db_pokemon)
33     await session.commit()
34     await session.refresh(db_pokemon)
35     await redis_client.delete(f"pokemon:{db_pokemon.name}")
36     return db_pokemon
37
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
FROM pg_catalog.pg_class JOIN pg_catalog.pg_namespace ON pg_catalog.pg_namespace.oid = pg_catalog.pg_class
2024-07-25 13:47:46,404 INFO sqlalchemy.engine.Engine SELECT pg_catalog.pg_class.relname
FROM pg_catalog.pg_class JOIN pg_catalog.pg_namespace ON pg_catalog.pg_namespace.oid = pg_catalog.pg_class
FROM pg_catalog.pg_class JOIN pg_catalog.pg_namespace ON pg_catalog.pg_namespace.oid = pg_catalog.pg_class
.relnamespace
WHERE pg_catalog.pg_class.relname = $1::VARCHAR AND pg_catalog.pg_class.relkind = ANY (ARRAY[$2::VARCHAR,
$3::VARCHAR, $4::VARCHAR, $5::VARCHAR, $6::VARCHAR]) AND pg_catalog.pg_table_is_visible(pg_catalog.pg_clas
s.oid) AND pg_catalog.pg_namespace.nspname != $7::VARCHAR
2024-07-25 13:47:46,406 INFO sqlalchemy.engine.Engine [generated in 0.00096s] ('pokemons', 'r', 'p', 'f',
.relnamespace
WHERE pg_catalog.pg_class.relname = $1::VARCHAR AND pg_catalog.pg_class.relkind = ANY (ARRAY[$2::VARCHAR,
$3::VARCHAR, $4::VARCHAR, $5::VARCHAR, $6::VARCHAR]) AND pg_catalog.pg_table_is_visible(pg_catalog.pg_clas
.relnamespace
WHERE pg_catalog.pg_class.relname = $1::VARCHAR AND pg_catalog.pg_class.relkind = ANY (ARRAY[$2::VARCHAR,
.relnamespace
WHERE pg_catalog.pg_class.relname = $1::VARCHAR AND pg_catalog.pg_class.relkind = ANY (ARRAY[$2::VARCHAR,
$3::VARCHAR, $4::VARCHAR, $5::VARCHAR, $6::VARCHAR]) AND pg_catalog.pg_table_is_visible(pg_catalog.pg_clas
s.oid) AND pg_catalog.pg_namespace.nspname != $7::VARCHAR
2024-07-25 13:47:46,406 INFO sqlalchemy.engine.Engine [generated in 0.00096s] ('pokemons', 'r', 'p', 'f',
'v', 'm', 'pg_catalog')
2024-07-25 13:47:46,409 INFO sqlalchemy.engine.Engine COMMIT
INFO: Application startup complete.
```