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:

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

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

app > crud.py > get_pokemon_by_name

```python
# app/crud.py

import aioredis
from sqlalchemy.future import select
from sqlalchemy.ext.asyncio import AsyncSession
from .models import Pokemon
from .schemas import PokemonCreate
from .config import REDIS_URL

redis = aioredis.from_url(REDIS_URL)

async def get_pokemon_by_name(session: AsyncSession, name: str) -> Pokemon:
    cache_key = f"pokemon:{name}"
    cached_pokemon = await redis.get(cache_key)
    if cached_pokemon:
        return Pokemon.parse_raw(cached_pokemon)

    result = await session.execute(select(Pokemon).filter(Pokemon.name == name))
    pokemon = result.scalars().first()
    if pokemon:
        await redis.set(cache_key, pokemon.json(), ex=3600)
    return pokemon

async def get_all_pokemons(session: AsyncSession, name: str = None) -> list[Pokemon]:
    query = select(Pokemon)
    if name:
        query = query.filter(Pokemon.name.ilike(f"%{name}%"))
    result = await session.execute(query)
    return result.scalars().all()

async def create_pokemon(session: AsyncSession, pokemon: PokemonCreate) -> Pokemon:
```

Updated working Cache:

app > crud.py > get_pokemon_by_name

```python
import redis.asyncio as redis
from sqlalchemy.future import select
from sqlalchemy.ext.asyncio import AsyncSession
from models import Pokemon
from schemas import PokemonCreate
from config import REDIS_URL

redis_client = redis.from_url(REDIS_URL)

async def get_pokemon_by_name(session: AsyncSession, name: str) -> Pokemon:
    cache_key = f"pokemon:{name}"
    cached_pokemon = await redis_client.get(cache_key)
    if cached_pokemon:
        return Pokemon.parse_raw(cached_pokemon)

    result = await session.execute(select(Pokemon).filter(Pokemon.name == name))
    pokemon = result.scalars().first()
    if pokemon:
        await redis_client.set(cache_key, pokemon.json(), ex=3600)
    return pokemon

async def get_all_pokemons(session: AsyncSession, name: str = None) -> list[Pokemon]:
    query = select(Pokemon)
    if name:
        query = query.filter(Pokemon.name.ilike(f"%{name}%"))
    result = await session.execute(query)
    return result.scalars().all()

async def create_pokemon(session: AsyncSession, pokemon: PokemonCreate) -> Pokemon:
    db_pokemon = Pokemon(**pokemon.dict())
    session.add(db_pokemon)
    await session.commit()
    await session.refresh(db_pokemon)
    await redis_client.delete(f"pokemon:{db_pokemon.name}")
    return db_pokemon
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS uvicorn

```
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
.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.
```