BACKEND STRUCTURE

Everything related to backend is in /backend/

```
/backend/
|   request_structs/
|   |   requests.py
|   const.py
|   controller.py
|   db_manager.py
|   main.py
|   utils.py
```

- main.py - module responsible for fastapi
- request_structs/requests.py - contains structs used as parameters or request body in api requests
- controller.py - contatins class Controller responsible for executing various, specific commands on the database
- db_manager.py - contains class DbManager responsible for creating, opening and maintaining connection to an SQL database file
- utils.py - utility funtions used in various modules
- const.py - const values and returns codes used in various modules

main.py

- Contains each GET,POST,DELETE,PATCH api endpoint
- If an endpoint needs access to data in database it creates a connection using class Controller from controller.py
- Each connection with controller is 'atomic'. After an endpoint end, the connection with controller is closed. This ensures that each change in database is a transaction and every change is saved
- Session tokens are fully encrypted

request_structs/requests.py

- Contains parameters and request bodies used in api endpoitns
- Each class must inherit from pydantic.BaseModel

controller.py

- Contains specific functions for executing SQL commands on the database, like SELECTs, DELETEs, UPDATEs

db_manager.py

- Contains support functions for simple SQL commands
- Each SELECT from database is returned as List[dict]

- Datetime objects in database are stored in format 'YYYY-MM-DD HH:MM:SS' and should be inserted as time in UTC timezone
- Use execute(command,params) to execute any SQL command
- Use save_to_csv(output_dir_path) to save all data into csv files in selected output directory

const.py

- Contains return codes for better transparency of an outcome of a function

utils.py

- Contains widely used functions - mostly datetime conversion and data parsing