Database Design Project 4 Short Report
Siwon Kim, Hyunjung Lee

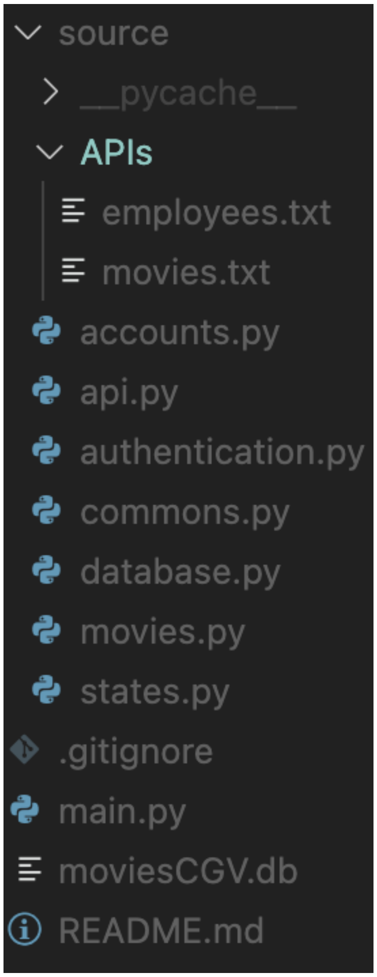
You are required to submit a short report that shows you have implemented an initial version of all three tiers (i.e., interface, application logic, and database) of the proposed system design. The functionalities implemented for each tier, however, can be of the simplest form when this report is due. To convince us that the system is in a working status, you can either show some screenshots or attach chunks of your code into the report, or do both.

- An initial version of web three-tier architecture
  - Interface: Using python, the interface might consist of a simple command-line interface that allows users to enter commands to search for movies, add movies to their watched list, or perform other basic tasks.
  - Application logic: To implement the application logic, we write Python code that interacts with the database using the sqlite3 library.
  - Database: We create an SQLite database with tables for Users, Movies, and Watched, as described in the database schema provided in our previous report.

- Functionalities
  - A database that consists of at least three relations, and you should load the tables with some data (you need to either make up some data or borrow data from other sources with permission from the data owner)
    - Admin page to view all dataset on database (if needed)
  - At least eight different types of queries (in terms of SQL statements) should be supported by your system. You must have some queries that can modify the content of your database
    - When bringing some dataset - SELECT
    - When modifying some dataset - UPDATE
    - When deleting some dataset - DELETE

    - When adding some dataset - INSERT
    - When first creating the table - CREATE TABLE
  - A text-based (or web-based) interface with at least three different appearances (or web pages)
    - Login/Register page
    - Client page - to search the movie and save the watched movie list
    - Admin page - to manage movies (add, edit, and delete)
  - Create users for your system, assign userID/password to them
  - Create views for your database, assign different privileges to different users
    - Only employee can add, edit, and delete the movie
  - Using stored procedures/functions to process application logic on the database side
    - By using stored procedures instead of SQLite to manage the admin side database
  - Any other features that are relevant to the system
    - User-side
      - View movie list
      - Search by Title/Director/Genre/Year
      - Store the movie watched
      - View watched movie List
      - View personal Information
      - Logout
    - Admin-side
      - View movie list
      - Add new movies
      - Edit the movie list - critics only
      - Delete the movie list
      - View personal information
      - Logout

- Directory Structure



```
source
  __pycache__
  APIs
    employees.txt
    movies.txt
  accounts.py
  api.py
  authentication.py
  commons.py
  database.py
  movies.py
  states.py
.gitignore
main.py
moviesCGV.db
README.md
```