Documentation Flask Movie Database

Instructions running application:

Step 1) setting up FLASK environment variable

Run in root of project on command line to enable debugger and reloader:

>>> *export FLASK_APP=flask_application_assignment.py*

The reloader was useful for testing functionality, as it automatically executed the updated code without requiring me to manually restart flask_application_assignment.py each time.

Step 2) Host flask application locally

>>> python flask_application_assignment.py
Now you can access the movie database in your browser at http://localhost:5000/.

Queries for creating database:

I entered this command at the terminal to enter the SQLite3 shell: sqlite3 movies.db.
See documentation/database_creation.png for the queries I used to create the initial state of movies.db from step 2a.

Reasoning for methods

The assignment asks me to explain my choices regarding the methods as parameters for the @app.route() decorator. I didn't choose any though since they were provided to us, however I can explain 'why' these methods specifically were used/provided. I've done this in the code using docstrings.

Using the database:

Step by step showcase of functionality with reference to screenshots:

1) Open application in browser by going to http://localhost:5000/. (see db_state_0.png)
2) Deleted Gone Girl movie with the delete button (see db_state_1.png).
3) Clicking Add a New Movie button leads to add_movie.html (see add_movie_page.png)
4) Filling in form and clicking Add Movie button leads back to updated index.html (see db_state_2.png)
5) Clicking edit button leads to add_movie.html, but in edit mode (see add_movie_page_edit_mode.png).
6) Changing relevant field and clicking Update Movie button leads back to updated index.html (see db_state_3.png).

SQL_Shell_db_state.png confirms back-end state of movies.db matches front-end as seen in 6).

Finally see debug_method_log.png to get an overview of the methods that were called during steps 1 to 6.