

**Text File:**

The text file is updated as new IMBD data is added to the app (more movies, more movie categories). A cron is setup on separate server and runs on a schedule to watch for changes in this text file. When there is a change in the text file the cron invokes a script written in python to enter the data into the PostgresSQL database.

**Client (browser):**

When a user loads the app/website a HTTP get request (via AJAX) is sent to the web server to get the HTML page for the app’s initial view. This HTTP get request sends an endpoint (url) that corresponds to the web server’s API. As the app grows to contain more data the user’s interaction with the data visualization (through filters, buttons...ect) may trigger more HTTP get requests to get the data intended to display on the app’s view. A MVC framework like Angular may be used to manage the updated data and views as well.

**Web Server (HTTP server in python):**

Django will be the HTTP framework that communicates with the database via ORM and the Django REST framework will be used to build the API endpoints. Django asks the database for objects (i.e. data) and then makes a JSON file through a python script. When a get request is made to the server a JSON response is returned to the client and the HTTP request is closed.

**Database (PostgresSQL):**

This database has a table with all of the IMBD data and is structured so that each movie is a row and the columns are the various categories: year, votes, rank, title and individual genres (mystery, drama, …etc).