Ivan Bilyarov – Q13681281

Assignment link for GitHub - https://github.com/3bilyi81/COM519\_Assignment\_2.git

**Introduction**

Most of the time when I go to websites like Netflix, Hulu etc. I cannot see the whole catalogue of movies/tv series that they have. Instead I have to use google to find out or make an account, subscribe to the service provider and then see what exactly they provide. To make it more convenient for myself and other people I decided to create a website that lists all the movies/tv series for Netflix, Hulu, Disney + and Amazon Prime using database from Kaggle. I got inspired by the old theater/cinema designs and their way of listing movies.

**MongoDB**

MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need.

MongoDB’s document model is simple for developers to learn and use, while still providing all the capabilities needed to meet the most complex requirements at any scale. We provide drivers for 10+ languages, and the community has built dozens more.

MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time

The document model maps to the objects in your application code, making data easy to work with

Ad hoc queries, indexing, and real time aggregation provide powerful ways to access and analyze your data

MongoDB is a distributed database at its core, so high availability, horizontal scaling, and geographic distribution are built in and easy to use

MongoDB is free to use. Versions released prior to October 16, 2018 are published under the AGPL. All versions released after October 16, 2018, including patch fixes for prior versions, are published under the Server Side Public License (SSPL) v1.

High availability through built-in replication and failover

Horizontal scalability with native sharding

End-to-end security

Native document validation and schema exploration with Compass

Management tooling for automation, monitoring, and backup

Fully elastic database as a service with built-in best practices

**NodeJS**

As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.

This is in contrast to today's more common concurrency model, in which OS threads are employed. Thread-based networking is relatively inefficient and very difficult to use. Furthermore, users of Node.js are free from worries of dead-locking the process, since there are no locks. Almost no function in Node.js directly performs I/O, so the process never blocks. Because nothing blocks, scalable systems are very reasonable to develop in Node.js.

Node.js is similar in design to, and influenced by, systems like Ruby's Event Machine and Python's Twisted. Node.js takes the event model a bit further. It presents an event loop as a runtime construct instead of as a library. In other systems, there is always a blocking call to start the event-loop. Typically, behavior is defined through callbacks at the beginning of a script, and at the end a server is started through a blocking call like EventMachine::run(). In Node.js, there is no such start-the-event-loop call. Node.js simply enters the event loop after executing the input script. Node.js exits the event loop when there are no more callbacks to perform. This behavior is like browser JavaScript — the event loop is hidden from the user.

HTTP is a first-class citizen in Node.js, designed with streaming and low latency in mind. This makes Node.js well suited for the foundation of a web library or framework.

Node.js being designed without threads doesn't mean you can't take advantage of multiple cores in your environment. Child processes can be spawned by using our child\_process.fork() API, and are designed to be easy to communicate with. Built upon that same interface is the cluster module, which allows you to share sockets between processes to enable load balancing over your cores.

**Express**

ExpressJS is a web application framework that provides you with a simple API to build websites, web apps and back ends. With ExpressJS, you need not worry about low level protocols, processes, etc.

Express provides a minimal interface to build our applications. It provides us the tools that are required to build our app. It is flexible as there are numerous modules available on npm, which can be directly plugged into Express.

Express was developed by TJ Holowaychuk and is maintained by the Node.js foundation and numerous open source contributors.

Unlike its competitors like Rails and Django, which have an opinionated way of building applications, Express has no "best way" to do something. It is very flexible and pluggable.

**Design**

I looked at images of the old movie theaters and cinemas from way back in the day. I used the internet for that, mainly google images. I noticed that the most common colors used are black, white and red, so I used them in my design. Another thing I noticed is that everything seems to be blocky, so I tried to implement that into my design as well. I wanted to give the user the authentic experience of old movie theater/ cinema.

I created home page explaining what the website is for. I added pages that list all the movies/tv series for each platform. I added Update page on which the user can update listings or the delete listings if they are out of date. And I added a contact page in case the user wants to ask a question about the listings, the website or anything related to it.

**Code**

For the database code, I looked at the class notes but I also watched youtube tutorials from channel named Academind. I have referenced them below. I had to install mongodb, node js and express. I imported the database from the JSON file into a collection. Created URL and requested all the pages needed. Then I created three functions for Getting, Inserting, Updating and Deleting Data. And then I exported the whole module.

For the html/css part, I used W3Schools for help. I set the resolution of the website to 1280x720 because almost every device nowadays has at least high definition, so it will be supported on most displays whether it’s a tablet, phone, monitor, laptop etc. This is the reason I did not add a breakpoint because I doubt that there is many devices out there or at least many people out there who will use lower resolution than 1280x720. And it should fit on bigger displays as well. There might be a bit of space on the side when viewed on bigger display but that means ads can be put on to fill up the space.

**Testing**

I used the live server extension in Visual Studio Code to test the html and css part and it was done through google chrome. I could not test the database code because I could not get it to work. I kept getting errors and I could not fix them. It did not even allow me to put the JSON file containing the movie listings into a collection. I’ve written the command as a comment in there, so you can see what I have used to try to do this. It is in the app.js file. I think that all the other commands should be working because they perform the post and get method requests. However, since there is no database connection I kept getting a blank page.

**Conclusion**

Overall, it’s great start. However, I could have improved the design by adding textures and images. Maybe curtains as a background and a film roll as a logo to make it look an actual theater/cinema. Maybe position the forms better. I could have worked more on the database, like allowing the user to register an account, add rating/review for the listings that the user can input and much more. Which also lead to my next point that the website is not secure at the moment due to the fact that everyone can use it and change the information on it without having to create an account.

**References:** Kaggle.com. 2021. *Netflix Movies And TV Shows*. [online] Available at: <https://www.kaggle.com/shivamb/netflix-shows> [Accessed 14 January 2021].; W3schools.com. 2021. *W3schools Online Web Tutorials*. [online] Available at: <https://www.w3schools.com/> [Accessed 14 January 2021].; Youtube.com. 2021. [online] Available at: <https://www.youtube.com/watch?v=ZKwrOXl5TDI> [Accessed 14 January 2021].; Youtube.com. 2021. [online] Available at: <https://www.youtube.com/watch?v=-JcgwLIh0Z4> [Accessed 14 January 2021].; MongoDB. 2021. What Is Mongodb?. [online] Available at: <https://www.mongodb.com/what-is-mongodb> [Accessed 15 January 2021].; Node.js. 2021. About | Node.Js. [online] Available at: <https://nodejs.org/en/about/> [Accessed 15 January 2021].; Tutorialspoint.com. 2021. Expressjs - Overview - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/expressjs/expressjs\_overview.htm> [Accessed 15 January 2021].