Fianance Guru

TBD TBD TBD

1 Introduction

The goal of this project is to build a database of stock information. This includes stock prices as well as general information about the respective stock companies. Therefor we build a web server which acts as an interface between the user and the AlphaVantage API. While using our web server to query information from the API, the server stores all queried information in a MongoDB and thereby builds our database.
The main page of our web server is the index page `http://localhost/`, see Figure 1. Here a user can insert any kind of query to find a company of there choice, eg. one can search for a ticker or a name. This query is forwarded to the AlphaVantage API and a list of possible matching results is displayed on the search result page page `http://localhost/stock`, see Figure 2. After selecting the company of choice, the webpage redirects you to the stock details `http://localhost/:symbol:` page, where `:symbol:` is the ticker that identifies the requested company. Here all relevant information that was queried form the AlphaVantage API is displayed, see Fgiure **??**. At the same time the server stores the company information in our own database. For each company that a user can inspect, the server also offers to download the data in the XML format. We also provide a XML schema file `http://localhost/schema` that can be used to validate all downloaded XML files.

2 Usage

The web server is build with JavaScript and node. As prerequisites you need to

1. install node as well as its package manager npm
2. install python3
3. have a running MongoDB, either as a local server or by using an online cluster
4. create an account for using the AlphaVantage API.

If all prerequisites are satisfied, clone the repository and install all needed packages by running `npm install` in the main folder. Now you need to setup the port on which you want your web server to be available as well as the URI to your MongoDB database. Therefor create a file called `.env` in the main folder (see also in the next section) and adding the following two lines

```
MONGODB_URI = "<URI-TO-SERVER>"
PORT = <YOUR-PORT-NO>
API_KEY = "<YOUR-API-KEY>"
```

Now you should be able to start the server by running the command `node server` in the main folder.

3 Files

Path	Description
`views/`	Contains the Embedded-JavaScript templates for rendering the html pages.
`views/index.ejs`	Template for the search page.
`views/stock.ejs`	Template for the result page, showing the matching results.
`views/stock_results.ejs`	Template for the page, showing the stock details.
`models/stock.js`	The database schema for an entry in our database.
`Controllers/stockController.js`	Contains the backend for the three pages of our webserver. Here the API requests as well as adding/updating entries in our database is done.
`.env`	Contains two environment variables that must be set to start the webserver and connect to the MongoDB database.
`server.js`	Entry script for the webserver. Start this file with node. Contains the code for assembling the webserver and defining the available pages.
`converter.py`	Contains the code for converting json to xml. Will be called by the server script automatically.
`output.xsd`	Contains the XML schema, which we offer to validate downloaded XML files.

4 Webpage description / Example queries

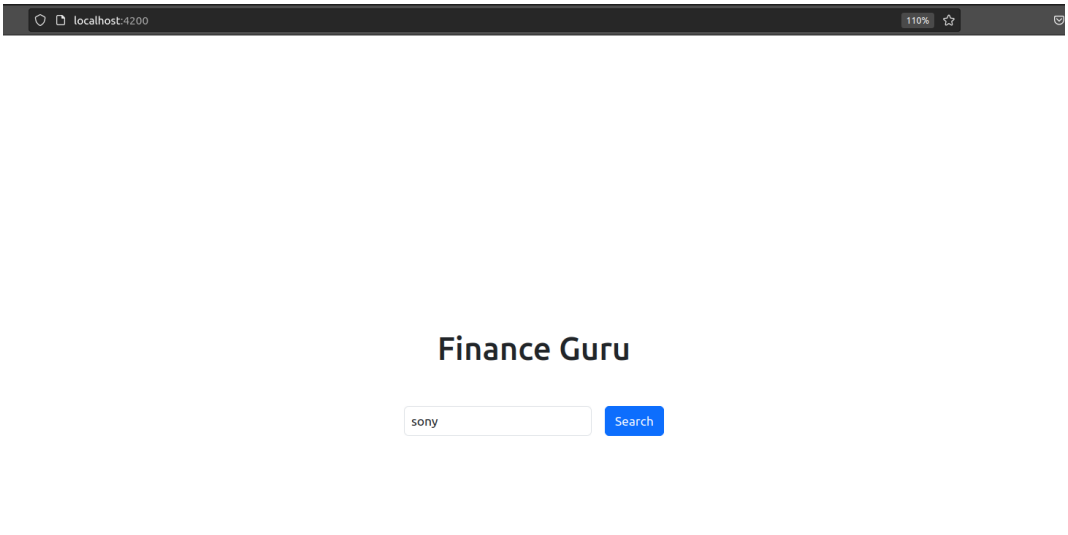

Figure 1: Index-Page

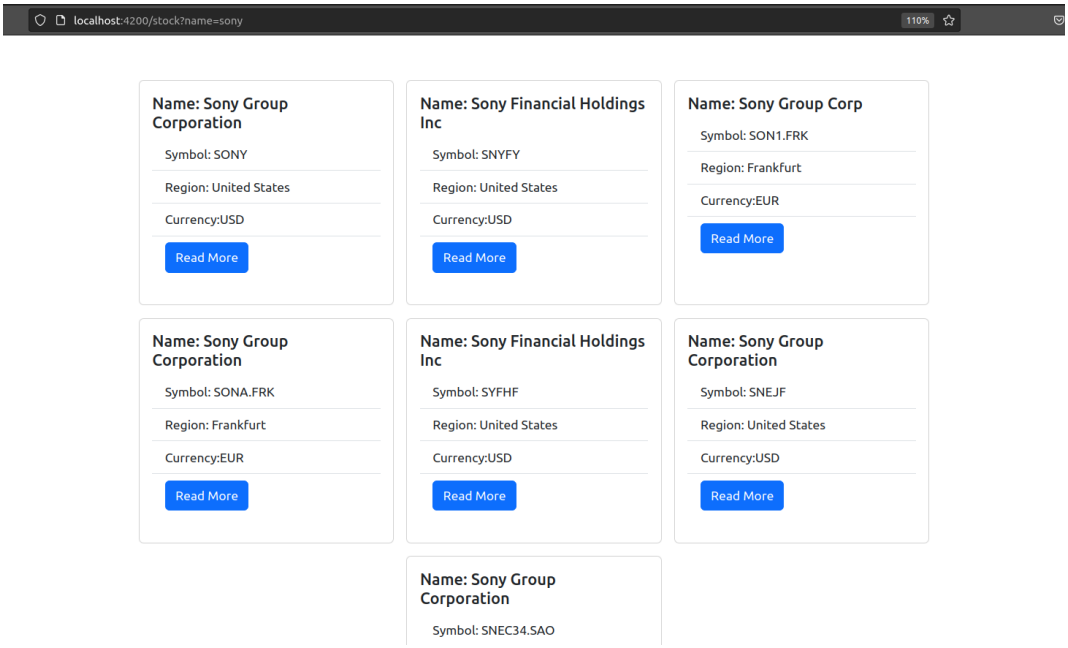

Figure 2: Search-Result-Page

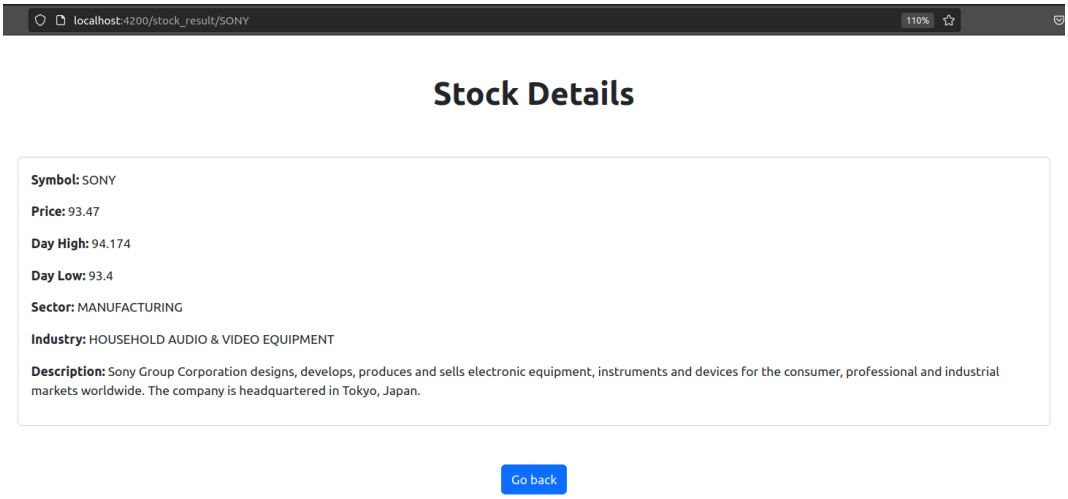

Figure 3: Index-Page

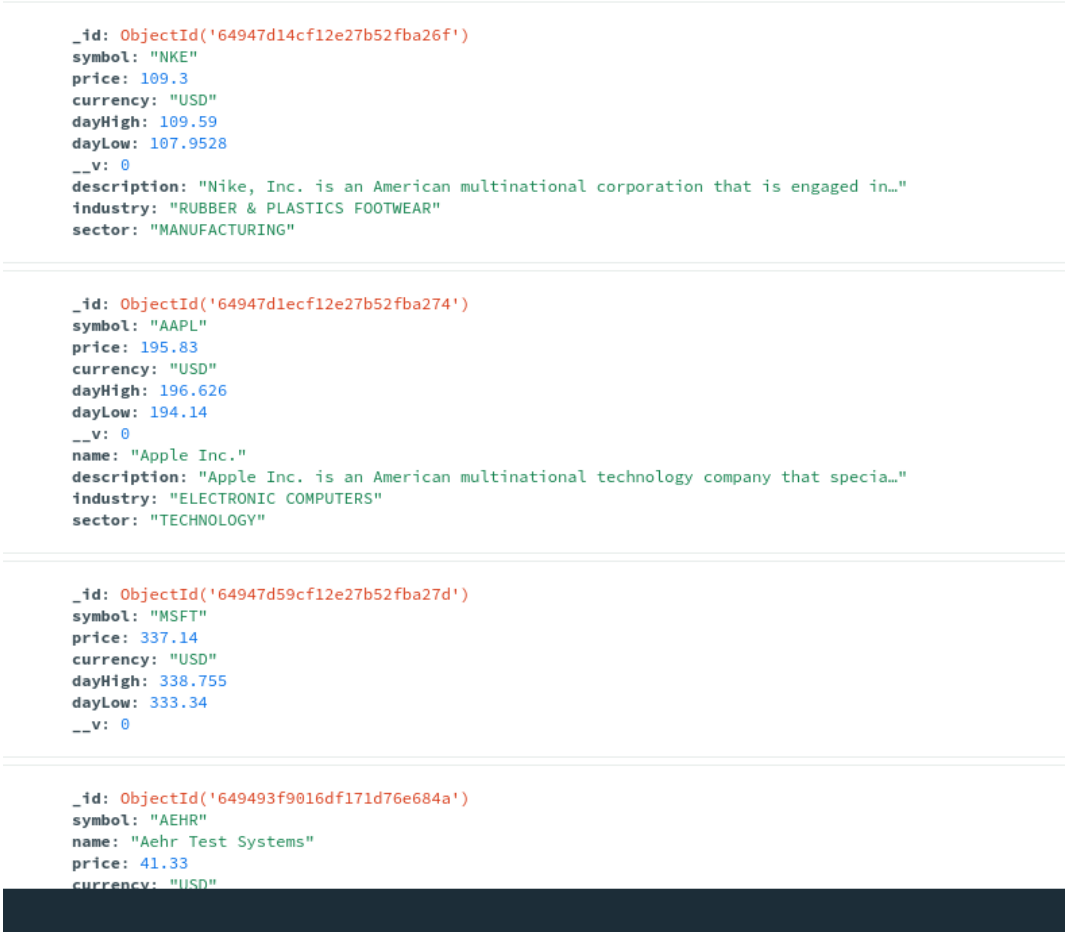

Figure 4: Example for how our database looks like. This is the only collection/table used.