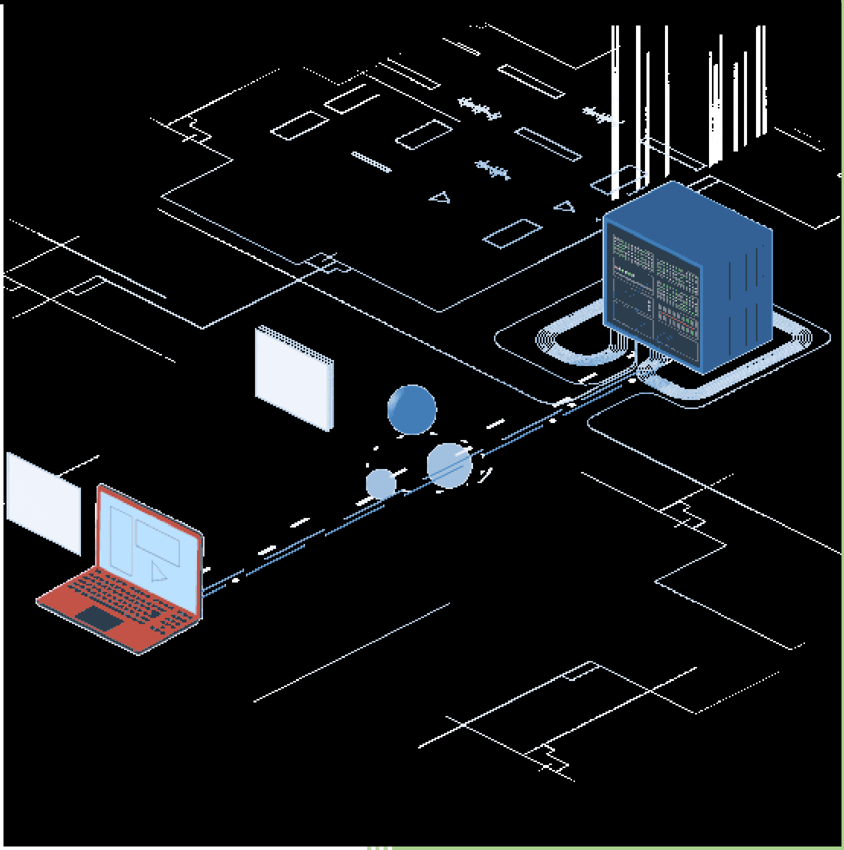
2020

CAB230 Stocks API – Server Side



CAB230

Stocks API – Server-Side Application

<Sharafat Mir>

<n10015957>

5/31/2020

Contents

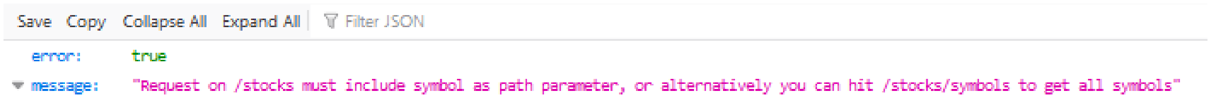
This template is adapted from one created for a more elaborate application. The original author spends most of his professional life talking to clients and producing architecture and services reports. You may find this a bit more elaborate than you are used to, but it is there to help you get a better mark

This report will probably be around 5 pages or so including screenshots

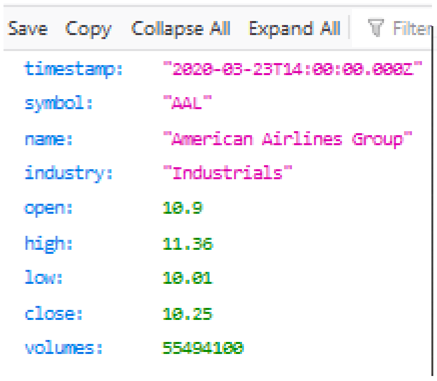
Introduction

Purpose & description

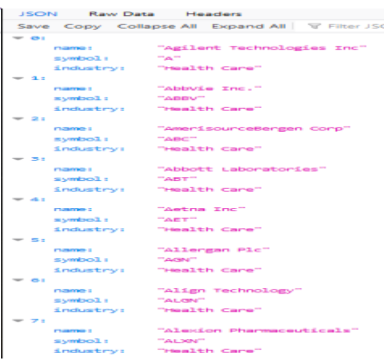
This is server-side Express application which is built to function as an API. This express application will allow React applications, any other technologies and people to request the resources such as stocks, and it's full details such as name, symbol, timestamp, open price, close price, high price, stocks industry and the volumes were sold that specific day. The application also uses the swagger documentation to help users have easy access and for testing purposes. All these resources are request by appropriate routes which will be explained in further details. The application will send all that information in the form of JSON string. This Express Application is also secure to use as there are various technologies and resources implemented and used to provide that security for the application and its users. The application also handles errors and if someone types and want information about a company which doesn't exists, they will be displayed appropriate messages as some of the examples of the application error handling and successful resources response from the API. Moreover, this Express application uses various technologies, packages, modules, and libraries and other services to provide services and all of that will be discussed below.



```
  error:     true
▾ message:   "Request on /stocks must include symbol as path parameter, or alternatively you can hit /stocks/symbols to get all symbols"
```



```
timestamp:   "2020-03-23T14:00:00.000Z"
symbol:      "AAL"
name:        "American Airlines Group"
industry:    "Industrials"
open:        10.9
high:        11.36
low:         10.01
close:       10.25
volumes:     55494100
```



```
▾ 0:
    name:       "Agilent Technologies Inc"
    symbol:     "A"
    industry:   "Health Care"
▾ 1:
    name:       "Abbvie Inc."
    symbol:     "ABBV"
    industry:   "Health Care"
▾ 2:
    name:       "AmerisourceBergen Corp"
    symbol:     "ABC"
    industry:   "Health Care"
▾ 3:
    name:       "Abbott Laboratories"
    symbol:     "ABT"
    industry:   "Health Care"
▾ 4:
    name:       "Aetna Inc"
    symbol:     "AET"
    industry:   "Health Care"
▾ 5:
    name:       "Allergan Plc"
    symbol:     "AGN"
    industry:   "Health Care"
▾ 6:
    name:       "Align Technology"
    symbol:     "ALGN"
    industry:   "Health Care"
▾ 7:
    name:       "Alexion Pharmaceuticals"
    symbol:     "ALXN"
    industry:   "Health Care"
```

More pictures of the application can be seen in the Appendix A

Completeness and Limitations

Here we want you to tell us in a couple of sentences what works and what doesn't. ***Make a claim against the standards we laid out in the assignment specification (see below) and briefly justify that claim.*** Don't give us deep details of the bugs here. Putting a positive spin on what you have achieved is fine – focus on the stuff that works. But be realistic in your claim. As for the client side, the text below is stolen straight from the assignment specification:

All the required functionality was successfully completed, and all of the routes were successfully and completely implemented. The application is deployed on Virtual Machine via Horizon VM provided by the CAB230 teaching team. The application serves the swagger docs on the home page or at the root level. The application can successfully implement all the requirements there are such as using various technologies node, express, mysq, swagger (yammal source which was supplied), knex, helmet, morgan, jwt, and others. All of these resources helped in the development of this application. The login, registration and any other requirements were all done 100%.

/stocks/symbols
The end point is fully functional.

/stocks/{symbol}
The end point is fully functional.

/stocks/authed/{symbol}
The end point is fully functional.

/user/register
The end point is fully functional.

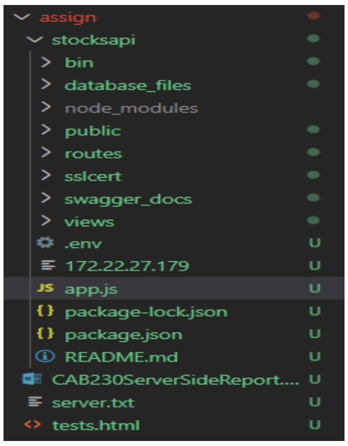
/user/login
The end point is fully functional.

Modules used
In the development of this application no external modules other than found in the JWT and Server-side practical were used.

No additional modules used

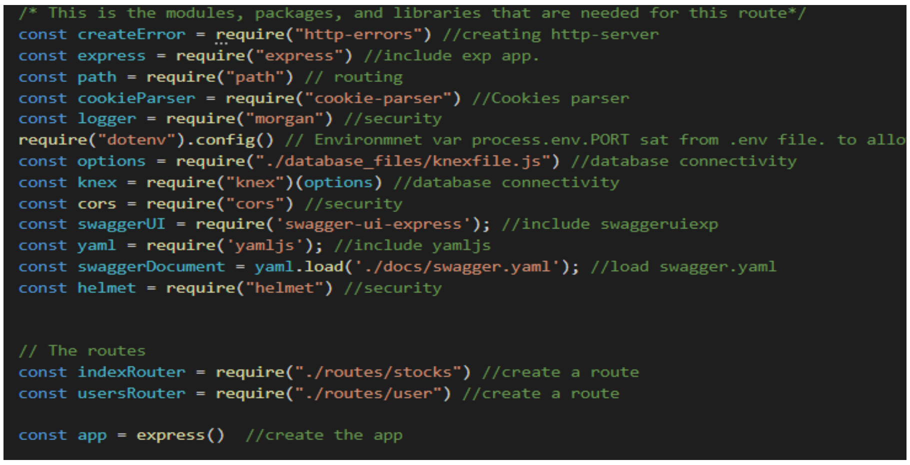
Technical Description
Architecture
Briefly describe the overall architecture of your application at a source code level. We expect that the application will closely follow the model imposed by Express generator, but there are many aspects of this application on which the generator is silent, and we want to know more about your choices. Most of this text should talk about the organization of the routing and the authentication and database connection services. You should use one or more screenshots of your directory structure. Here is one of ours, and yours may be markedly different and still fine:

The architecture of the application is organized and laid out professionally. There is folder called assign which is the project inside it as a folder called server. Inside the server folder there is the application and all the folders and files that helps the application. All of those files are; bin directory which contains the file www and each time when we run the application by typing npm start, npm runs this file. The database_files directory is a folder which has the relevant files for the database. This folder has the file called knexfile.js which uses the knex to connect to the database. Moreover, the node_modules are all the modules which are installed by the npm when we type npm install. These modules are important for the application to run. The public directory has the html file which is public to users of the api and they can access it. The routes directory controls all the endpoints apart from the login, and the registration functionality or endpoints. The swagger_docs directory contains the swagger files that are need for this application. The views directory has the files which are jade mark-up files and they are used by the application. The .env file contains some variables information that are used by the application. The app.js is the main file and all of the routes and middleware are mounted here in the app.js file. The package-lock.json is also in the root directory and used by the application. The package.json file contains all the modules and its versions are written recorded here and each time npm install the application used this. and README.md files have some information about the application use. As can be seen in the photo.

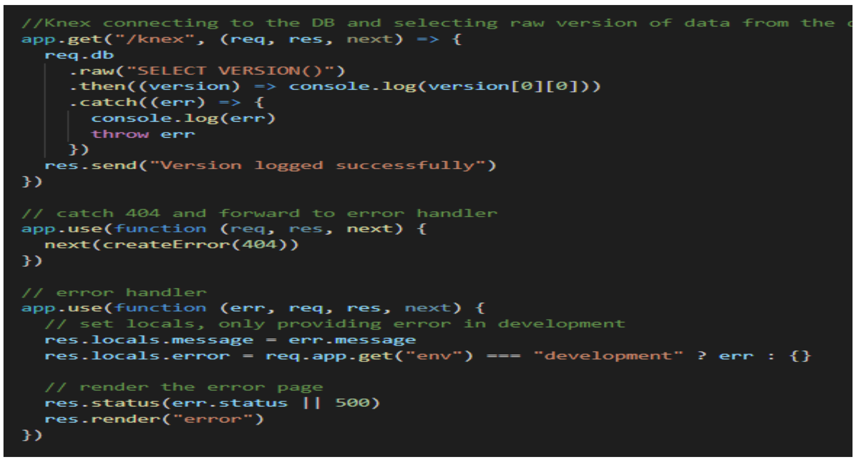


```
assign
  stocksapi
    bin
    database_files
    node_modules
    public
    routes
    sslcert
    swagger_docs
    views
    .env                          U
    172.22.27.179                 U
    app.js                        U
    package-lock.json             U
    package.json                  U
    README.md                     U
  CAB230ServerSideReport....      U
  server.txt                      U
  tests.html                      U
```

The app.js has all the applications middleware and routes mounted here. As can be seen in the photo below. The app.js file uses the routes directory and the files in it which are users, and stocks. The application uses those files for all the endpoints such as the /stocks/symbols, /stocks/symbol, and /stocks/authed/symbol. The login and registration routes are implemented in the user.js file.



```
/* This is the modules, packages, and libraries that are needed for this route*/
const createError = require("http-errors") //creating http-server
const express = require("express") //include exp app.
const path = require("path") // routing
const cookieParser = require("cookie-parser") //Cookies parser
const logger = require("morgan") //security
require("dotenv").config() // Environmnet var process.env.PORT sat from .env file. to allo
const options = require("./database_files/knexfile.js") //database connectivity
const knex = require("knex")(options) //database connectivity
const cors = require("cors") //security
const swaggerUI = require('swagger-ui-express'); //include swaggeruiexp
const yaml = require('yamljs'); //include yamljs
const swaggerDocument = yaml.load('./docs/swagger.yaml'); //load swagger.yaml
const helmet = require("helmet") //security


// The routes
const indexRouter = require("./routes/stocks") //create a route
const usersRouter = require("./routes/user") //create a route

const app = express()  //create the app
```



```
//Knex connecting to the DB and selecting raw version of data from the
app.get("/knex", (req, res, next) => {
  req.db
    .raw("SELECT VERSION()")
    .then((version) => console.log(version[0][0]))
    .catch((err) => {
      console.log(err)
      throw err
    })
  res.send("Version logged successfully")
})

// catch 404 and forward to error handler
app.use(function (req, res, next) {
  next(createError(404))
})

// error handler
app.use(function (err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message
  res.locals.error = req.app.get("env") === "development" ? err : {}

  // render the error page
  res.status(err.status || 500)
  res.render("error")
})
```

Security

- Use of Knex

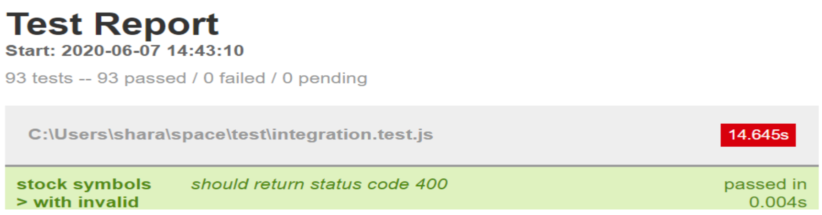
  Knex has been used in this node express application. This allow this application to connect to the database. It builds queries without raw SQL.

- Use of helmet
  Helmet has been used in this node express application. This allow the application to set the security related http responses headers.
- Use of morgan
  Morgan has been used in this node express application. Which helps and simplifies the process of logging request to this application.
- JWT and bcrypt
  Bcrypt was also used to store the user's passwords in the database in forms of some un-understandable string. This helps it will keep the passwords safe if someone unauthorized have access to it over the network they will not be able to understand it easily.

Testing

The application passes the test 100%.

The application can also be tested remotely. By changing the URL in the integration.test.js file to the URL of my application which is https://172.22.27.179 The test will work and will be successful.



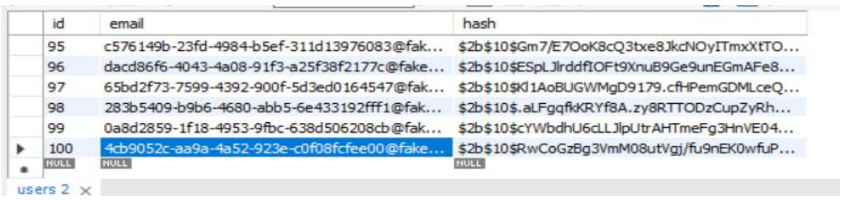
```
Test Report
Start: 2020-06-07 14:43:10
93 tests -- 93 passed / 0 failed / 0 pending

C:\Users\shara\space\test\integration.test.js                         14.645s

stock symbols     should return status code 400                      passed in
> with invalid                                                          0.004s
```

Difficulties / Exclusions / unresolved & persistent errors /

No bugs are present in the application. No errors are present in the application. All the application's required functionalities have completed.

Installation guide

To install the application, you will need to take a few steps as they're detailed below. You will have to have a connection to a database named webcomputing. In this database there should be two tables one called stocks and other called users. The stocks table will have information about stocks and the users table will have store information about users email and passwords.

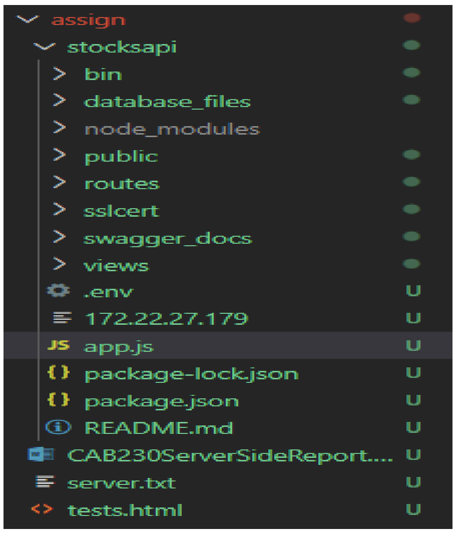


id	email	hash
95	c576149b-23fd-4984-b5ef-311d13976083@fak...	$2b$10$Gm7/E7OoK8cQ3txe8JkcNOyITmxXtTO...
96	dacd86f6-4043-4a08-91f3-a25f38f2177c@fake...	$2b$10$ESpLJlrddfIOFt9XnuB9Ge9unEGmAFe8...
97	65bd2f73-7599-4392-900f-5d3ed0164547@fak...	$2b$10$Kl1AoBUGWMgD9179.cfHPemGDMLceQ...
98	283b5409-b9b6-4680-abb5-6e433192fff1@fak...	$2b$10$.aLFgqfkKRYf8A.zy8RTTODzCupZyRh...
99	0a8d2859-1f18-4953-9fbc-638d506208cb@fak...	$2b$10$cYWbdhU6cLLJlpUtrAHTmeFg3HnVE04...
100	4cb9052c-aa9a-4a52-923e-c0f08fcfee00@fake...	$2b$10$RwCoGzBg3VmM08utVgj/fu9nEK0wfuP...
NULL	NULL	NULL

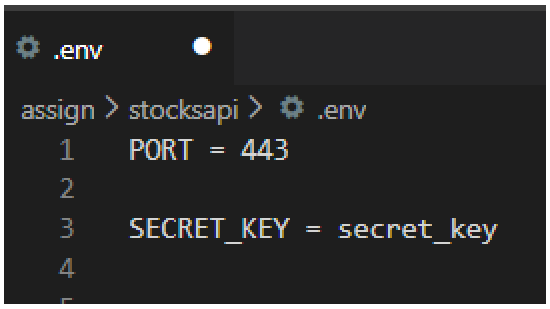
Step 1: Download the file assign.zip then unzip it.

Step 1.1: After unzipping the file you should open the project in command line interface and go to the directory assign then go to the stocks api folder which has the all the projects files such as app.js, databas_files, swagger_docs files and directories.

Step 1.2: After you're in that folder make sure there is the file called .env present in the root directory. So when you enter the stocks api you should be able to see all file including '.env' file. As can be seen in the picture. If there is no .env file you will need to create one .env file and save the .env files with the code in the left picture and needs to be saved in the root directory inside stocksapi. The code is need for .env file PORT = 443, SECRET_KEY = secret_key. Sometimes the .env files disappear when zipping and transferring.

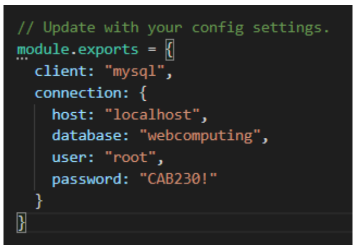


```
assign
  stocksapi
    bin
    database_files
    node_modules
    public
    routes
    sslcert
    swagger_docs
    views
    .env                          U
    172.22.27.179                 U
    app.js                        U
    package-lock.json             U
    package.json                  U
    README.md                     U
  CAB230ServerSideReport....      U
  server.txt                      U
  tests.html                      U
```



```
.env
assign > stocksapi > .env
PORT = 443

SECRET_KEY = secret_key
```

Step 1.3: Go to the knexfile.js inside database_file folder and change the database connections servers' details, and the password etc. As can be seen in the picture below

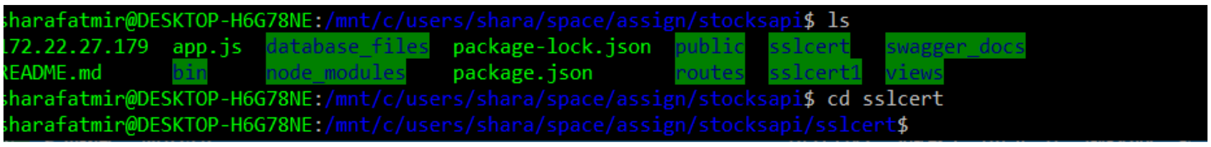


```
// Update with your config settings.
module.exports = {
  client: "mysql",
  connection: {
    host: "localhost",
    database: "webcomputing",
    user: "root",
    password: "CAB230!"
  }
}
```

Step 1.4: Then you will need to check sslcert folder. Inside this folder you should expect there to be two files which are cert.key and cert.crt. Sometimes .key or .crt files won't copy on linux OS when zipping folders OR sometimes these certificate get corrupted which means you will not be able to use the application and you will get errors.

If there is no cert.key or cert.crt then you will need to create a self-signed certificate.

To create a self-signed certificate you will need to go to the folder called sslcert as can be seen in the picture:

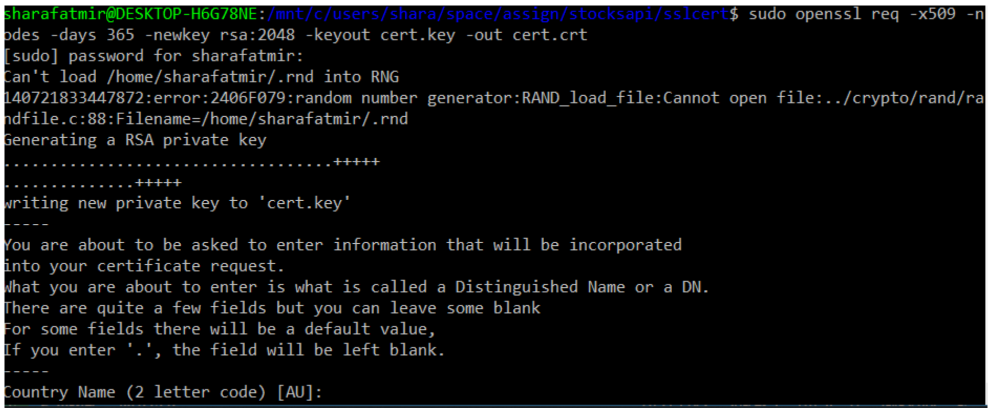


```
harafatmir@DESKTOP-H6G78NE:/mnt/c/users/shara/space/assign/stocksapi$ ls
72.22.27.179  app.js  database_files  package-lock.json  public  sslcert   swagger_docs
EADME.md      bin     node_modules    package.json       routes  sslcert1  views
harafatmir@DESKTOP-H6G78NE:/mnt/c/users/shara/space/assign/stocksapi$ cd sslcert
harafatmir@DESKTOP-H6G78NE:/mnt/c/users/shara/space/assign/stocksapi/sslcert$
```

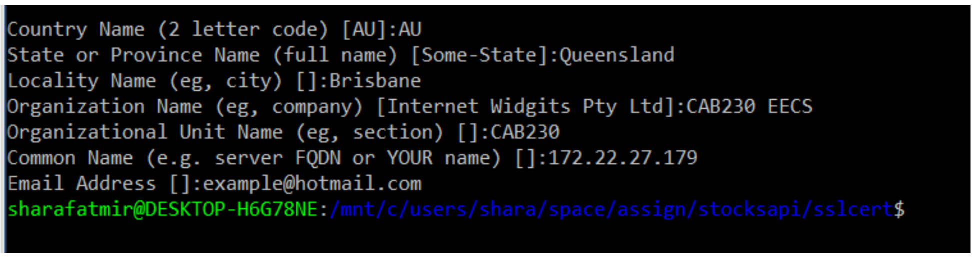
For this project we have to create the ssl certificate here and we will save it here. Because we have referenced this folder and said that the keys will be here. Inside the bin folder there is a file called www inside that file we are referencing to this key and certificate files which are saved here because we want to use it to allow us for the use of https and this will provide security.

Step 1.5: Type in the following as can be seen in the picture below to create a self-signed certificate.

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/ssl/private/node-selfsigned.key -out /etc/ssl/certs/node-
selfsigned.cr
```



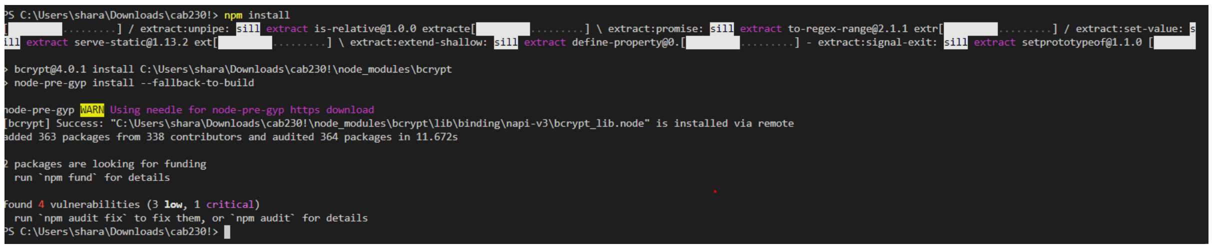
```
sharafatmir@DESKTOP-H6G78NE:/mnt/c/users/shara/space/assign/stocksapi/sslcert$ sudo openssl req -x509 -n
odes -days 365 -newkey rsa:2048 -keyout cert.key -out cert.crt
[sudo] password for sharafatmir:
Can't load /home/sharafatmir/.rnd into RNG
140721833447872:error:2406F079:random number generator:RAND_load_file:Cannot open file:../crypto/rand/ra
ndfile.c:88:Filename=/home/sharafatmir/.rnd
Generating a RSA private key
...................................+++++
.............+++++
writing new private key to 'cert.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
```



```
Country Name (2 letter code) [AU]:AU
State or Province Name (full name) [Some-State]:Queensland
Locality Name (eg, city) []:Brisbane
Organization Name (eg, company) [Internet Widgits Pty Ltd]:CAB230 EECS
Organizational Unit Name (eg, section) []:CAB230
Common Name (e.g. server FQDN or YOUR name) []:172.22.27.179
Email Address []:example@hotmail.com
sharafatmir@DESKTOP-H6G78NE:/mnt/c/users/shara/space/assign/stocksapi/sslcert$
```

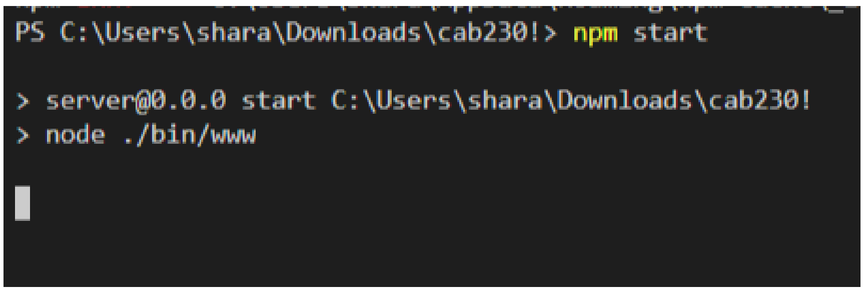
After that you have successfully created a certificate.

Step 2.1: Now you will install all the packages using npm or node package manager and modules that are required for the application to run. You will type the following command: npm install as can be seen in the picture.



```
PS C:\Users\shara\Downloads\cab230!> npm install
[         .........] / extract:unpipe: sill extract is-relative@1.0.0 extracte[         .........] \ extract:promise: sill extract to-regex-range@2.1.1 extr[         .........] / extract:set-value: s
ill extract serve-static@1.13.2 ext[         .........] \ extract:extend-shallow: sill extract define-property@0.[         .........] - extract:signal-exit: sill extract setprototypeof@1.1.0 [

> bcrypt@4.0.1 install C:\Users\shara\Downloads\cab230!\node_modules\bcrypt
> node-pre-gyp install --fallback-to-build

node-pre-gyp WARN Using needle for node-pre-gyp https download
[bcrypt] Success: "C:\Users\shara\Downloads\cab230!\node_modules\bcrypt\lib\binding\napi-v3\bcrypt_lib.node" is installed via remote
added 363 packages from 338 contributors and audited 364 packages in 11.672s

2 packages are looking for funding
  run `npm fund` for details

found 4 vulnerabilities (3 low, 1 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
PS C:\Users\shara\Downloads\cab230!>
```

Step 2.2: To run your application you will use npm or node package manager. You will type npm start as can be seen in the picture.



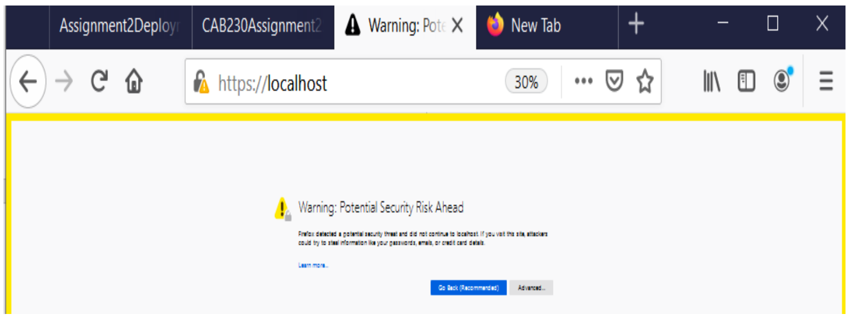
```
PS C:\Users\shara\Downloads\cab230!> npm start

> server@0.0.0 start C:\Users\shara\Downloads\cab230!
> node ./bin/www
```

Now the application is running and we can check it.

By running https://localhost

If we did all of these steps in linux we will then be able to use the assigned address which was https://172.22.27.179 but here we are working in windows. Therefore, we will use https://localhost and test it. It is successfully running as can be seen in the photo below.

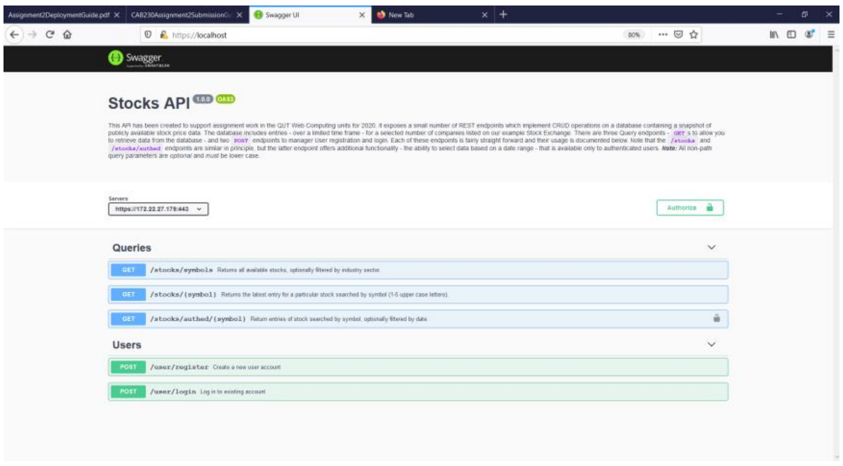
Now when you run it you will get a warning form the web browser saying the certificate is not trusted as it is self-assigned. But you will accept this risk of the certificate as this will be your own certificate.



Warning: Potential Security Risk Ahead

Firefox detected a potential security threat and did not continue to localhost. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

After accepting the risk we can see that the application is now working and as can be seen in the picture.



Stocks API 1.0.0 OAS3

This API has been created to support assignment work in the QUT Web Computing units for 2020. It exposes a small number of REST endpoints which implement CRUD operations on a database containing a snapshot of publicly available stock price data. The database includes entries - over a limited time frame - for a selected number of companies listed on our example Stock Exchange. There are three Query endpoints - `GET` s to allow you to retrieve data from the database - and two `POST` endpoints to manager User registration and login. Each of these endpoints is fairly straight forward and their usage is documented below. Note that the `/stocks` and `/stocks/authed` endpoints are similar in principle, but the latter endpoint offers additional functionality - the ability to select data based on a date range - that is available only to authenticated users. **Note:** All non-path query parameters are *optional* and *must* be lower case.

Servers: https://172.22.27.179:443

Queries

GET `/stocks/symbols` Returns all available stocks, optionally filtered by industry sector.

GET `/stocks/{symbol}` Returns the latest entry for a particular stock searched by symbol (1-5 upper case letters).

GET `/stocks/authed/{symbol}` Return entries of stock searched by symbol, optionally filtered by date.

Users

POST `/user/register` Create a new user account

POST `/user/login` Log in to existing account

References

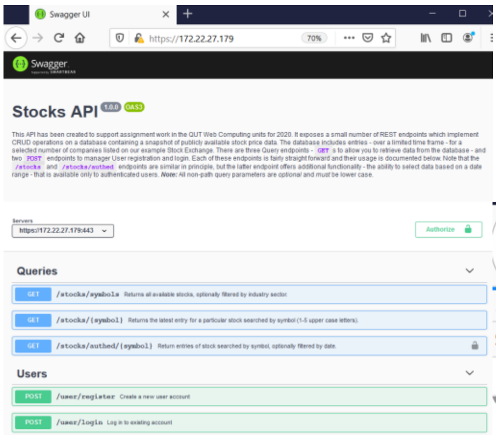
Use a standard approach to referencing – see the guidance at https://www.citewrite.qut.edu.au/cite/.

Figure1.1 is referenced from here

https://phpenthusiast.com/theme/assets/images/blog/what_is_rest_api.png

Appendices as you require them

Anything you think should be included but is better left to the end.



Swagger UI

https://172.22.27.179 70%

Swagger

Stocks API 1.0.0 OAS3

This API has been created to support assignment work in the QUT Web Computing units for 2020. It exposes a small number of REST endpoints which implement CRUD operations on a database containing a snapshot of publicly available stock price data. The database includes entries - over a limited time frame - for a selected number of companies listed on our example Stock Exchange. There are three Query endpoints - `GET` s to allow you to retrieve data from the database - and two `POST` endpoints to manager User registration and login. Each of these endpoints is fairly straight forward and their usage is documented below. Note that the `/stocks` and `/stocks/authed` endpoints are similar in principle, but the latter endpoint offers additional functionality - the ability to select data based on a date range - that is available only to authenticated users. *Note:* All non-path query parameters are optional and must be lower case.

Servers
https://172.22.27.179:443

Authorize

Queries

GET `/stocks/symbols` Returns all available stocks, optionally filtered by industry sector.

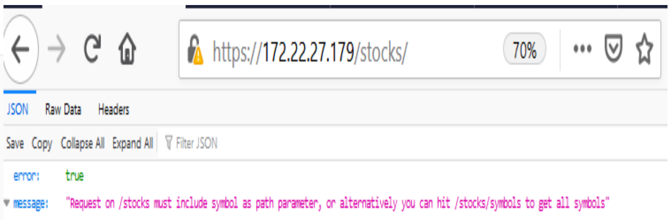
GET `/stocks/{symbol}` Returns the latest entry for a particular stock searched by symbol (1-5 upper case letters).

GET `/stocks/authed/{symbol}` Return entries of stock searched by symbol, optionally filtered by date.

Users

POST `/user/register` Create a new user account

POST `/user/login` Log in to existing account

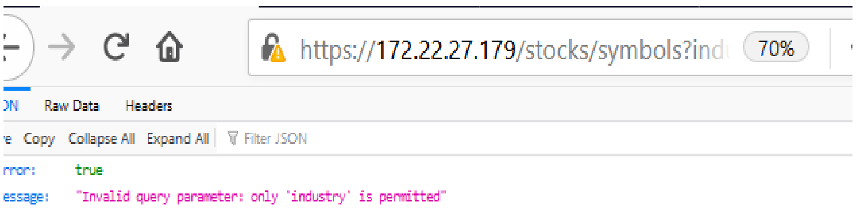


https://172.22.27.179/stocks/ 70%

JSON Raw Data Headers

Save Copy Collapse All Expand All Filter JSON

```
error:    true
message:  "Request on /stocks must include symbol as path parameter, or alternatively you can hit /stocks/symbols to get all symbols"
```

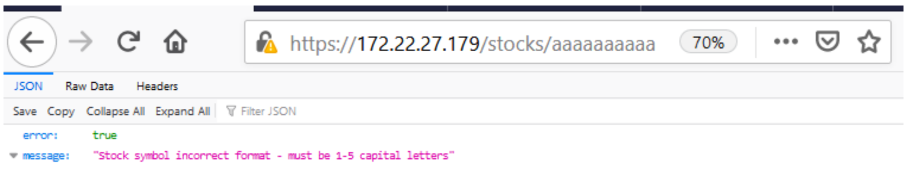


https://172.22.27.179/stocks/symbols?ind 70%

ON Raw Data Headers

e Copy Collapse All Expand All Filter JSON

```
rror:    true
essage:  "Invalid query parameter: only 'industry' is permitted"
```

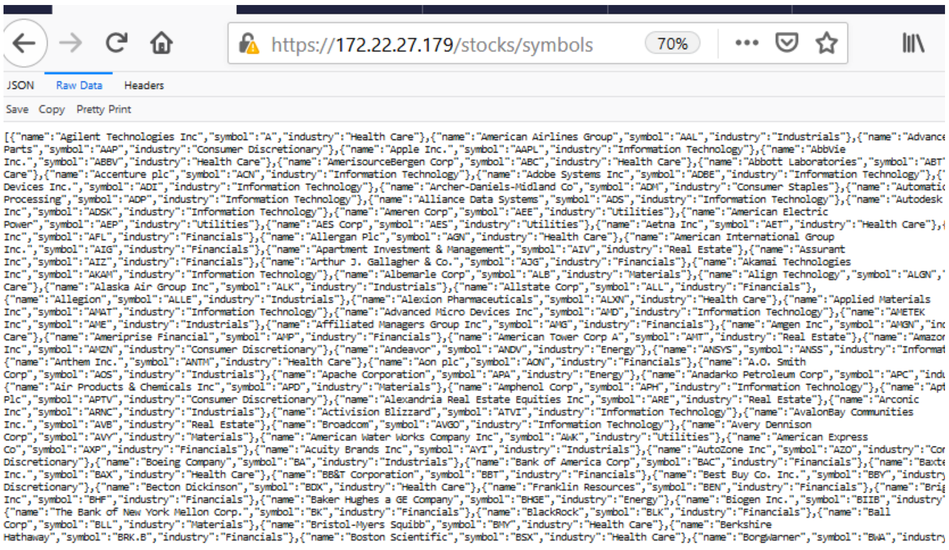


https://172.22.27.179/stocks/aaaaaaaaaa 70%

JSON Raw Data Headers

Save Copy Collapse All Expand All Filter JSON

```
error:    true
message:  "Stock symbol incorrect format - must be 1-5 capital letters"
```

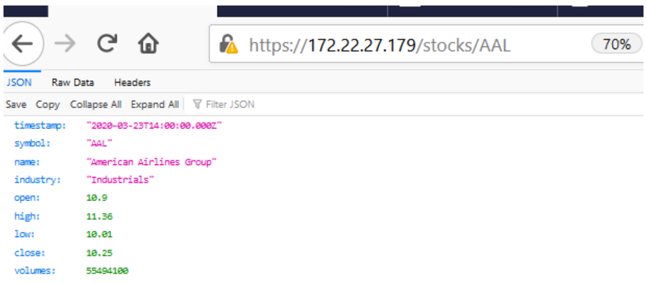


https://172.22.27.179/stocks/symbols 70%

JSON Raw Data Headers

Save Copy Pretty Print

```
[{"name":"Agilent Technologies Inc","symbol":"A","industry":"Health Care"},{"name":"American Airlines Group","symbol":"AAL","industry":"Industrials"},{"name":"Advance
Parts","symbol":"AAP","industry":"Consumer Discretionary"},{"name":"Apple Inc.","symbol":"AAPL","industry":"Information Technology"},{"name":"AbbVie
Inc.","symbol":"ABBV","industry":"Health Care"},{"name":"AmerisourceBergen Corp","symbol":"ABC","industry":"Health Care"},{"name":"Abbott Laboratories","symbol":"ABT"
Care"},{"name":"Accenture plc","symbol":"ACN","industry":"Information Technology"},{"name":"Adobe Systems Inc","symbol":"ADBE","industry":"Information Technology"},{"
Devices Inc.","symbol":"ADI","industry":"Information Technology"},{"name":"Archer-Daniels-Midland Co","symbol":"ADM","industry":"Consumer Staples"},{"name":"Automatic
Processing","symbol":"ADP","industry":"Information Technology"},{"name":"Alliance Data Systems","symbol":"ADS","industry":"Information Technology"},{"name":"Autodesk
Inc","symbol":"ADSK","industry":"Information Technology"},{"name":"Ameren Corp","symbol":"AEE","industry":"Utilities"},{"name":"American Electric
Power","symbol":"AEP","industry":"Utilities"},{"name":"AES Corp","symbol":"AES","industry":"Utilities"},{"name":"Aetna Inc","symbol":"AET","industry":"Health Care"},{
Inc","symbol":"AFL","industry":"Financials"},{"name":"Allergan Plc","symbol":"AGN","industry":"Health Care"},{"name":"American International Group
Inc.","symbol":"AIG","industry":"Financials"},{"name":"Apartment Investment & Management","symbol":"AIV","industry":"Real Estate"},{"name":"Assurant
Inc","symbol":"AIZ","industry":"Financials"},{"name":"Arthur J. Gallagher & Co.","symbol":"AJG","industry":"Financials"},{"name":"Akamai Technologies
Inc","symbol":"AKAM","industry":"Information Technology"},{"name":"Albemarle Corp","symbol":"ALB","industry":"Materials"},{"name":"Align Technology","symbol":"ALGN","
Care"},{"name":"Alaska Air Group Inc","symbol":"ALK","industry":"Industrials"},{"name":"Allstate Corp","symbol":"ALL","industry":"Financials"},
{"name":"Allegion","symbol":"ALLE","industry":"Industrials"},{"name":"Alexion Pharmaceuticals","symbol":"ALXN","industry":"Health Care"},{"name":"Applied Materials
Inc","symbol":"AMAT","industry":"Information Technology"},{"name":"Advanced Micro Devices Inc","symbol":"AMD","industry":"Information Technology"},{"name":"AMETEK
Inc","symbol":"AME","industry":"Industrials"},{"name":"Affiliated Managers Group Inc","symbol":"AMG","industry":"Financials"},{"name":"Amgen Inc","symbol":"AMGN","inc
Care"},{"name":"Ameriprise Financial","symbol":"AMP","industry":"Financials"},{"name":"American Tower Corp A","symbol":"AMT","industry":"Real Estate"},{"name":"Amazon
Inc","symbol":"AMZN","industry":"Consumer Discretionary"},{"name":"Andeavor","symbol":"ANDV","industry":"Energy"},{"name":"ANSYS","symbol":"ANSS","industry":"Informat
{"name":"Anthem Inc.","symbol":"ANTM","industry":"Health Care"},{"name":"Aon plc","symbol":"AON","industry":"Financials"},{"name":"A.O. Smith
Corp","symbol":"AOS","industry":"Industrials"},{"name":"Apache Corporation","symbol":"APA","industry":"Energy"},{"name":"Anadarko Petroleum Corp","symbol":"APC","indu
{"name":"Air Products & Chemicals Inc","symbol":"APD","industry":"Materials"},{"name":"Amphenol Corp","symbol":"APH","industry":"Information Technology"},{"name":"Apt
Plc","symbol":"APTV","industry":"Consumer Discretionary"},{"name":"Alexandria Real Estate Equities Inc","symbol":"ARE","industry":"Real Estate"},{"name":"Arconic
Inc","symbol":"ARNC","industry":"Industrials"},{"name":"Activision Blizzard","symbol":"ATVI","industry":"Information Technology"},{"name":"AvalonBay Communities
Inc.","symbol":"AVB","industry":"Real Estate"},{"name":"Broadcom","symbol":"AVGO","industry":"Information Technology"},{"name":"Avery Dennison
Corp","symbol":"AVY","industry":"Materials"},{"name":"American Water Works Company Inc","symbol":"AWK","industry":"Utilities"},{"name":"American Express
Co","symbol":"AXP","industry":"Financials"},{"name":"Acuity Brands Inc","symbol":"AYI","industry":"Industrials"},{"name":"AutoZone Inc","symbol":"AZO","industry":"Cor
Discretionary"},{"name":"Boeing Company","symbol":"BA","industry":"Industrials"},{"name":"Bank of America Corp","symbol":"BAC","industry":"Financials"},{"name":"Baxte
Inc.","symbol":"BAX","industry":"Health Care"},{"name":"BB&T Corporation","symbol":"BBT","industry":"Financials"},{"name":"Best Buy Co. Inc.","symbol":"BBY","industry
Discretionary"},{"name":"Becton Dickinson","symbol":"BDX","industry":"Health Care"},{"name":"Franklin Resources","symbol":"BEN","industry":"Financials"},{"name":"Brig
Inc","symbol":"BHF","industry":"Financials"},{"name":"Baker Hughes a GE Company","symbol":"BHGE","industry":"Energy"},{"name":"Biogen Inc.","symbol":"BIIB","industry"
{"name":"The Bank of New York Mellon Corp.","symbol":"BK","industry":"Financials"},{"name":"BlackRock","symbol":"BLK","industry":"Financials"},{"name":"Ball
Corp","symbol":"BLL","industry":"Materials"},{"name":"Bristol-Myers Squibb","symbol":"BMY","industry":"Health Care"},{"name":"Berkshire
Hathaway","symbol":"BRK.B","industry":"Financials"},{"name":"Boston Scientific","symbol":"BSX","industry":"Health Care"},{"name":"BorgWarner","symbol":"BWA","industry
```



https://172.22.27.179/stocks/AAL 70%

JSON Raw Data Headers

Save Copy Collapse All Expand All Filter JSON

```
timestamp:  "2020-03-23T14:00:00.000Z"
symbol:     "AAL"
name:       "American Airlines Group"
industry:   "Industrials"
open:       10.9
high:       11.36
low:        10.01
close:      10.25
volumes:    55494100
```