Store Finder API: Retrieve the 5 closest stores to location

Author:Fernando Castro

The following document contains:

*Instructions to run the API
*A general overview of the classes and files used
*Other relevant information regarding decisions

Summary of Technologies used:
***Java 11, Spring Boot, React.js, Maven**

***Github repository:** https://github.com/cachojava/storefinder

How to use the API(without the frontend):

*Download the code from the repository (if you use the first command you only need the jar file)
*You can run the API with any of the following commands:

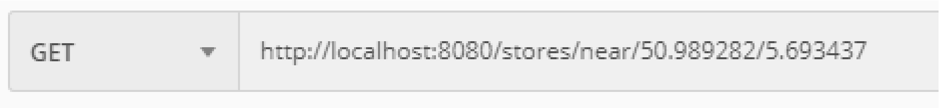
```
1) java -jar storefinder-0.0.1-SNAPSHOT.jar
2) mvn spring-boot:run
```

The API has one endpoint:

GET http://localhost:8080/stores/near/{latitude}/{longitude}

Latitude = Latitude from the position from which we want to know the closest stores
Longitude = Longitude from the position from which we want to know the closest stores

Example:



GET http://localhost:8080/stores/near/50.989282/5.693437

Validation and formatting:

Valid latitude is between -90 and 90 up to 6 decimals, example: 50.989282
Valid longitude is between -180 and 180 up to 6 decimals, example: 5.768530

No special characters or letters are permittied for latitude and longitude, both latitude and longitude are required and input is validated on the backend and frontend

Response:

The response contains all the store information from the json file adding:

Distance: Distance in kilometers calculated using the Harvesine Formula https://en.wikipedia.org/wiki/Haversine_formula between location and a specific store

formattedDistance: the same distance rounded to 3 decimals and added the KM symbol for being displayed

Example:

```
[
   {
      "city": "Urmond",
      "postalCode": "6129 EZ",
      "street": "Paalweg",
      "street2": "14-15",
      "street3": "",
      "addressName": "Jumbo Urmond Paalweg",
      "uuid": "udsKYx4X4RgAAAFILHUYwKxK",
      "longitude": "5.768530",
      "latitude": "50.989282",
      "complexNumber": "32110",
      "showWarningMessage": true,
      "todayOpen": "08:00",
      "locationType": "Supermarkt",
      "collectionPoint": false,
      "sapStoreID": "4911",
      "todayClose": "20:00",
      "distance": 5.256014054325421,
      "formattedDistance": "5.257 KM"
   },
   {
      "city": "Stein",
      "postalCode": "6171 JC",
      "street": "Raadhuisplein",
      "street2": "42",
      "street3": "",
      "addressName": "Jumbo Stein Raadhuisplein",
      "uuid": "jAoKYx4X49UAAAFWNpFQnZCz",
      "longitude": "5.765977",
      "latitude": "50.968011",
      "complexNumber": "32241",
      "showWarningMessage": true,
      "todayOpen": "08:00",
      "locationType": "SupermarktPuP",
      "collectionPoint": true,
      "sapStoreID": "4604",
      "todayClose": "21:00",
      "distance": 5.602258545608764,
      "formattedDistance": "5.603 KM"
   },
   {
      "city": "Geleen",
      "postalCode": "6163 CG",
      "street": "Bloemenmarkt",
      "street2": "32",
      "street3": "",
      "addressName": "Jumbo Geleen Penders",
      "uuid": "lEkKYx4X78oAAAFIHBcYwKxK",
      "longitude": "5.814687",
      "latitude": "50.984474",
      "complexNumber": "33530",
      "showWarningMessage": true,
      "todayOpen": "08:00",
      "locationType": "SupermarktPuP",
      "collectionPoint": true,
      "sapStoreID": "3656",
      "todayClose": "20:00",
        "distance": 8.503960152107634,
        "formattedDistance": "8.504 KM"
    },
    {
        "city": "Born",
        "postalCode": "6121 JB",
        "street": "Kapelweg",
        "street2": "4a",
        "street3": "",
        "addressName": "Jumbo Born Kapelweg",
        "uuid": "ohwKYx4XUc4AAAFIXQ0YwKxK",
        "longitude": "5.809002",
        "latitude": "51.032432",
        "complexNumber": "33514",
        "showWarningMessage": true,
        "todayOpen": "08:00",
        "locationType": "SupermarktPuP",
        "collectionPoint": true,
        "sapStoreID": "3600",
        "todayClose": "20:00",
        "distance": 9.401543238497156,
        "formattedDistance": "9.402 KM"
    },
    {
        "city": "Geleen",
        "postalCode": "6166 CV",
        "street": "Pastoor Vonckenstraat",
        "street2": "9",
        "street3": "",
        "addressName": "Jumbo Geleen Pastoor Vonckenstraat",
        "uuid": "b3QKYx4XQDYAAAFIrZUYwKxK",
        "longitude": "5.840834",
        "latitude": "50.971110",
        "complexNumber": "31556",
        "showWarningMessage": true,
        "todayOpen": "08:00",
        "locationType": "SupermarktPuP",
        "collectionPoint": true,
        "sapStoreID": "6377",
        "todayClose": "20:00",
        "distance": 10.51481370656669,
        "formattedDistance": "10.515 KM"
    }
]
```

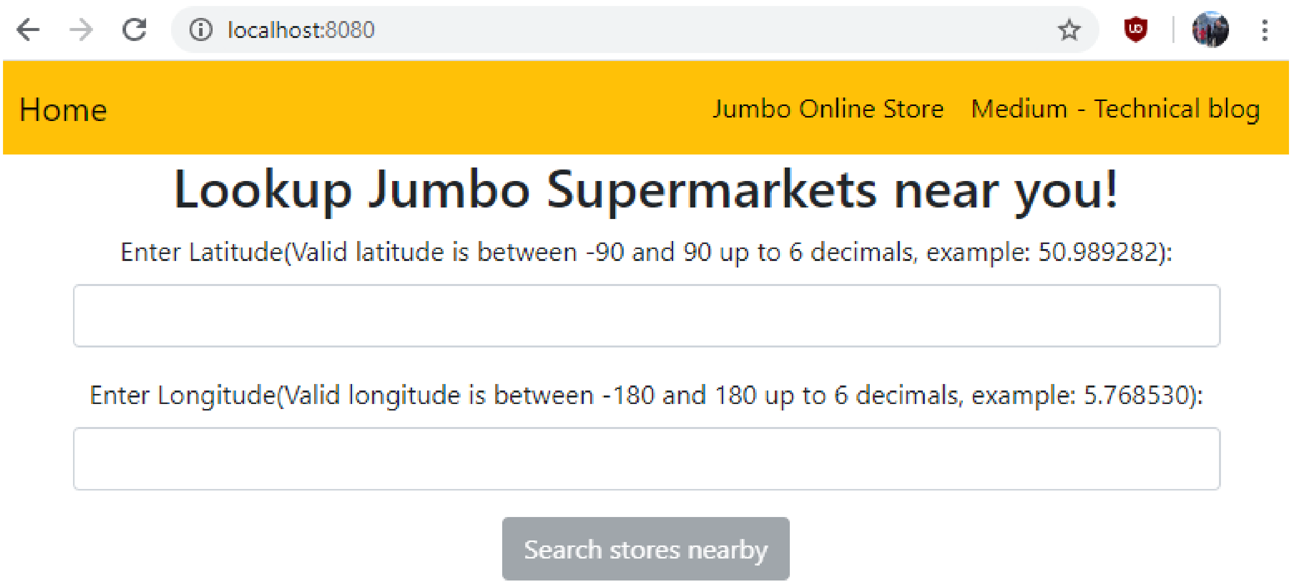
Run app including frontend

IMPORTANT: In order to setup the frontend I use a maven plugin which uses npm and yarn to manage javascript libraries, if you dont have those tools installed the plugin will look into installing them, this can make the first run to be slow, and occupying much more space than the one needed to just run the api (I think it can occupy around 300mb with node libraries and that the jar file size for running the API is below 20mb)

*

1)Start the application with the following command: mvn spring-boot:run -Pprod

2)Go to localhost:8080



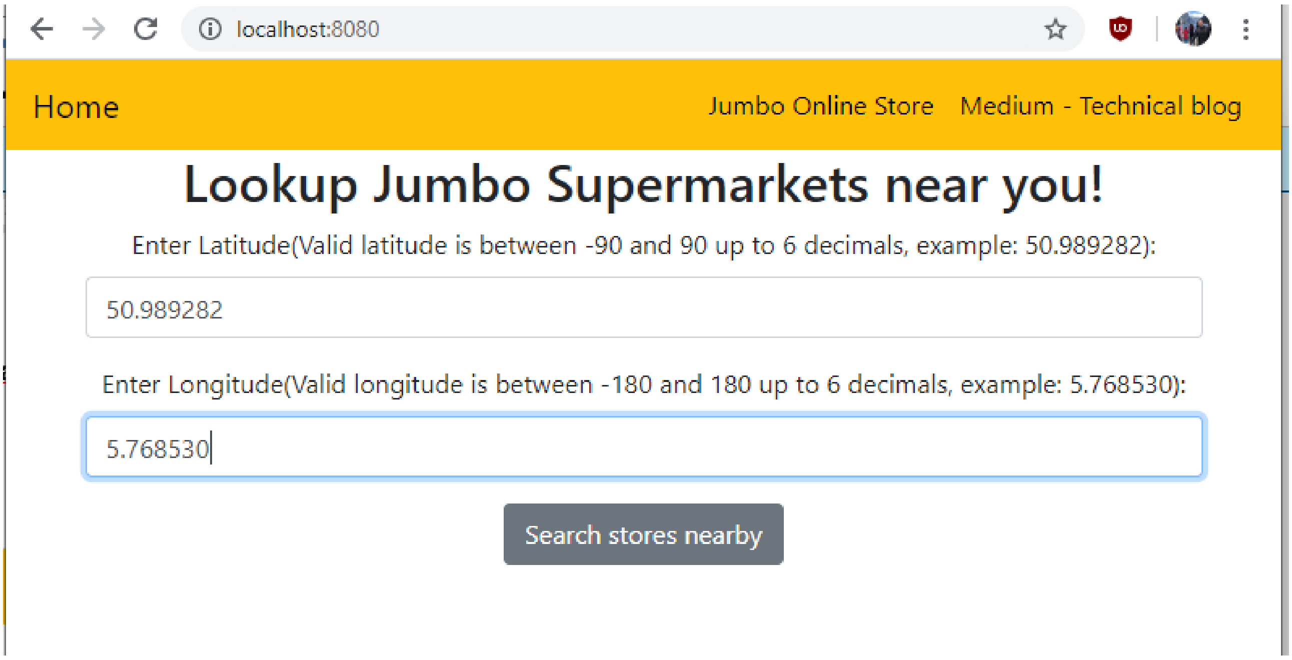
Lookup Jumbo Supermarkets near you!

Enter Latitude(Valid latitude is between -90 and 90 up to 6 decimals, example: 50.989282):

Enter Longitude(Valid longitude is between -180 and 180 up to 6 decimals, example: 5.768530):

First box is for the latitude and second one is for the longitude, the search button will be enabled once the two boxes have a valid input (is validated using a regex)

Example:



Lookup Jumbo Supermarkets near you!

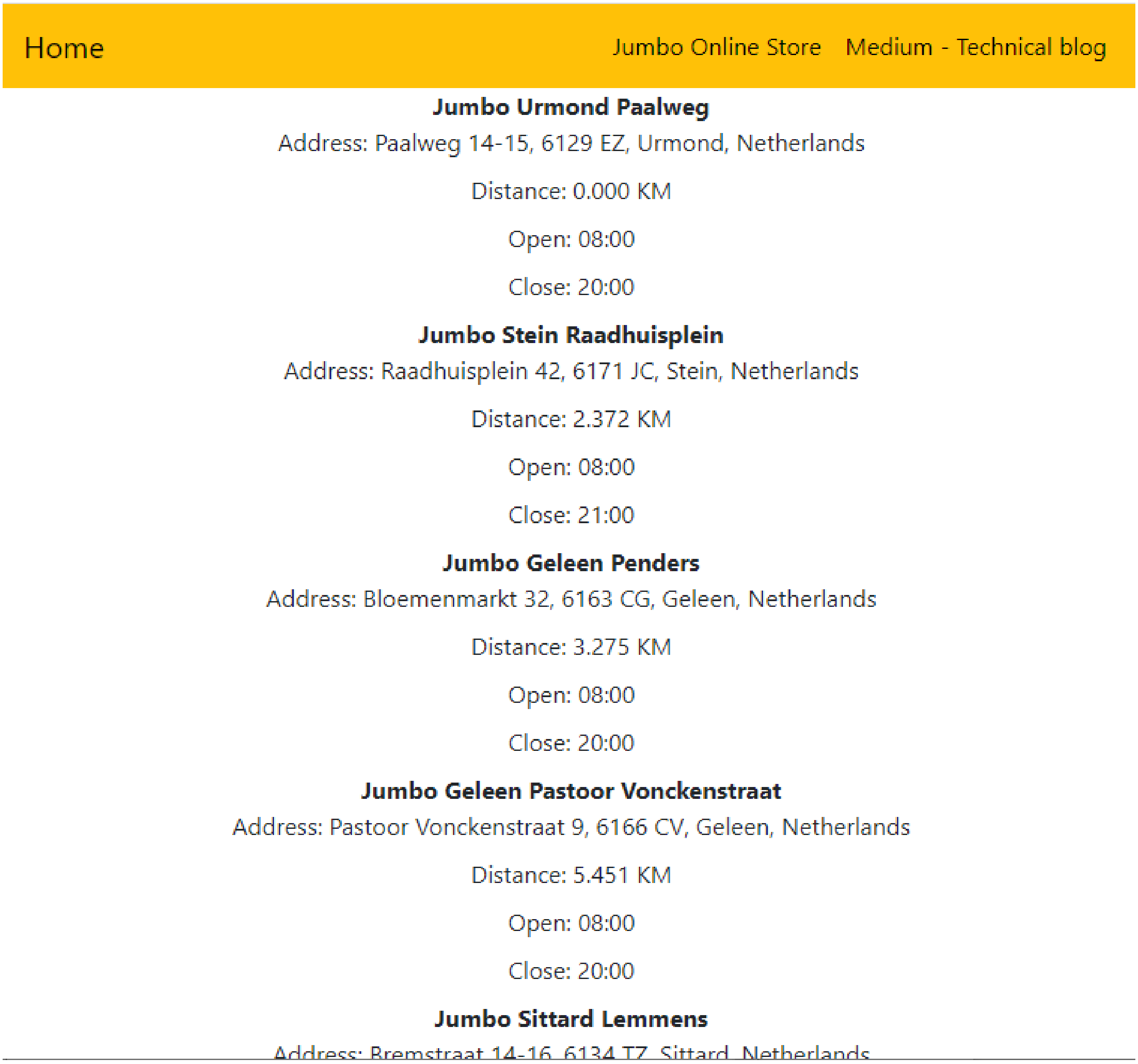
Enter Latitude(Valid latitude is between -90 and 90 up to 6 decimals, example: 50.989282):

50.989282

Enter Longitude(Valid longitude is between -180 and 180 up to 6 decimals, example: 5.768530):

5.768530

If we press the search button you will see a list like the following:



Jumbo Urmond Paalweg

Address: Paalweg 14-15, 6129 EZ, Urmond, Netherlands

Distance: 0.000 KM

Open: 08:00

Close: 20:00

Jumbo Stein Raadhuisplein

Address: Raadhuisplein 42, 6171 JC, Stein, Netherlands

Distance: 2.372 KM

Open: 08:00

Close: 21:00

Jumbo Geleen Penders

Address: Bloemenmarkt 32, 6163 CG, Geleen, Netherlands

Distance: 3.275 KM

Open: 08:00

Close: 20:00

Jumbo Geleen Pastoor Vonckenstraat

Address: Pastoor Vonckenstraat 9, 6166 CV, Geleen, Netherlands

Distance: 5.451 KM

Open: 08:00

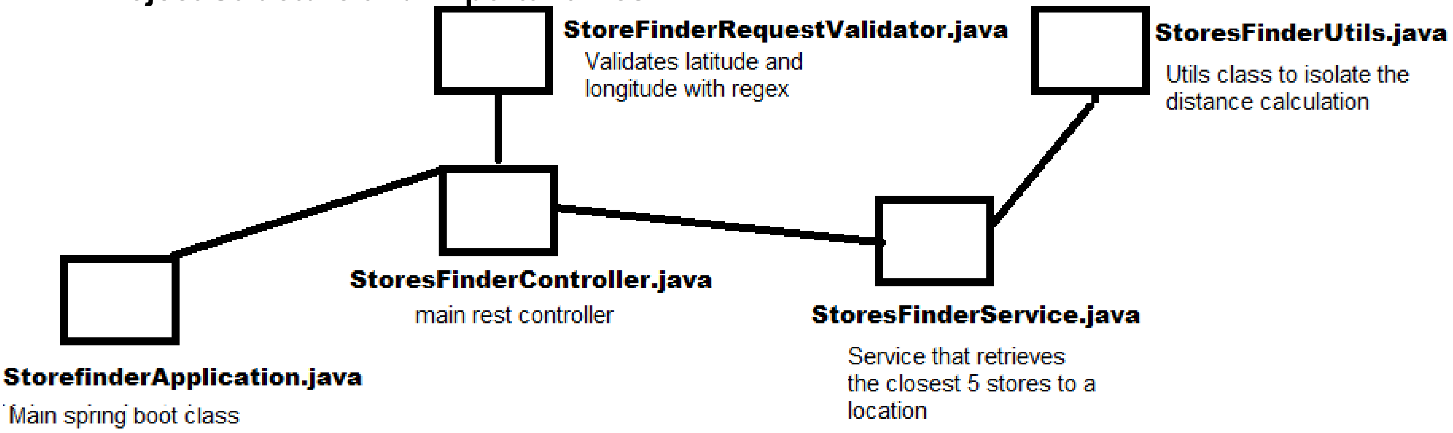
Close: 20:00

Jumbo Sittard Lemmens

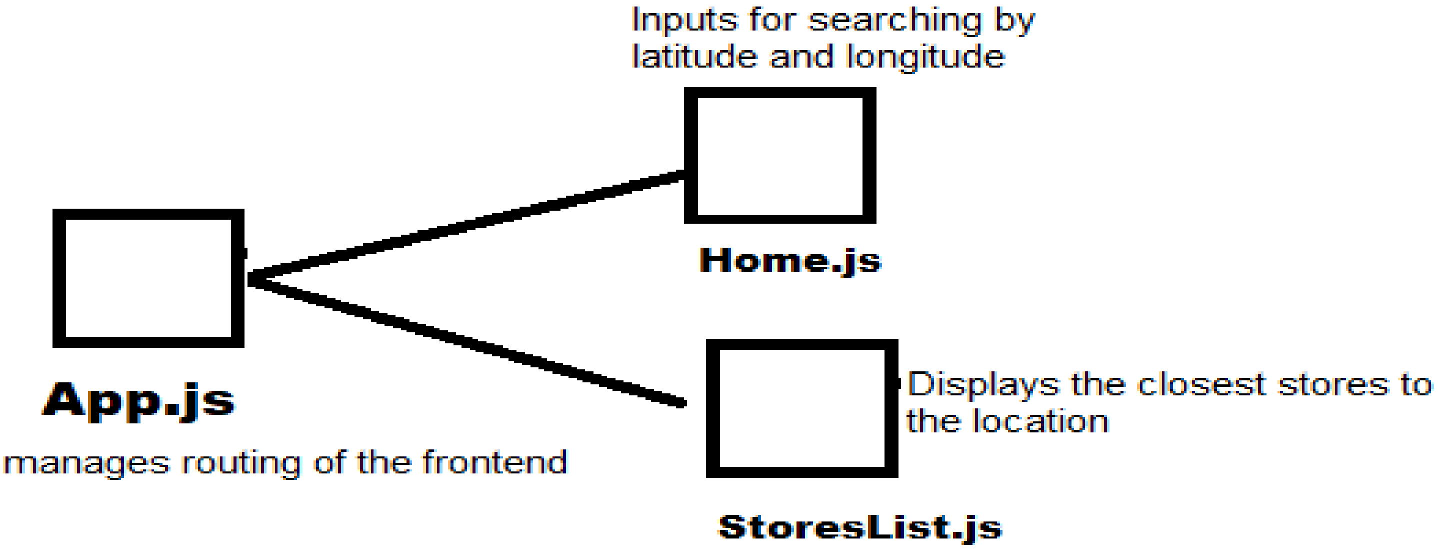
Address: Bremstraat 14-16, 6134 TZ, Sittard, Netherlands

Which contains information about the addres, distance, and open hours which I thought where the most relevant things to know

Project Structure and important files:


StoreFinderRequestValidator.java
Validates latitude and longitude with regex
StoresFinderUtils.java
Utils class to isolate the distance calculation
StoresFinderController.java
main rest controller
StoresFinderService.java
Service that retrieves the closest 5 stores to a location
StorefinderApplication.java
Main spring boot class

API Structure Summary


Inputs for searching by latitude and longitude
Home.js
App.js
manages routing of the frontend
Displays the closest stores to the location
StoresList.js

Frontend Architecture summary

Other files and classes:

AppNavBar.js: Top bar on the frontend which buttons home, a link to jumbo site and jumbo tech blog
StorefinderApplicationTests.java: Unit tests for the API
StoreFinderConstants.java: Constants used in the API
Stores.json: Data provided with the information of the stores
log4j2.xml: logs configuration