

Project Report:

Mongo App – A NoSQL project

|  |  |
| --- | --- |
| Course | Advanced Topics on NoSQL databases  Problem to solve |
| Team | 4 |
| Major | IBO-2 |
| Team members | TAZI Maxime  RAMOS Miguel  GOUJET Guillaume |

# SUMMARY

Introduction 2

summary of the project and queries 2

software and frameworks used 2

design of project 3

nosql database : mongo db 3

implementation 4

queries 5

conclusion 10

appendix #

# Introduction

###### summary of project and queries

On this project, we had to developed an application based on the language of our choice, with one of the NoSQL database that we had studied in class.

We then decided to create a Web application in JS, this application is used as an user interface for NoSQL database MongoDB

QUERIES :

* Search of town with parameters (name, country, population, timeZone)
* Search of closest town based on coordinates (longitude, latitude)
* Add a town to the database
* Filter the querries with the name of the town
* Sorting town by population
* Display of towns as a table list

###### Software and frameworks used

* Front-end : ReactJS
* Back-end : NodeJS
* Design : Bootstrap, CSS
* Database: Mongo DB, GUI: Studio 3T

# Design of project

##### NOSQL DATABASE: MONGO DB

Since we had some experience with MongoDB, we decided to use it as our database. We could have used CassandraDB, however MongoDB was more to our liking and more object oriented. We could also have used ElasticSearch, but our project is more into queries than analyzing data.

##### Implementation

Server side:

We used Node JS with Express module to host our WebServer with some endpoints.

Examples :



Figure 1: Search all cities endpoint (name, country … parameters)



Figure 2: Search near cities endpoint (lng and lat parameters)

Each query is handle by an endpoint with parameters.

The goal was to get the parameters from the user, so that mongoose module (enable the use of MongoDB with JS) could process the queries.



Figure 3: Requete near géré par mongoose

In doing so, we managed to get the result via an API

Client side:

To display the queries results and the parameters as forms, we used the ReactJS framework.

In ReactJS, every dynamic data has a state and that state is changed when an event function is called. For example, we created a state for the towns. When the user hit the search button, it creates an event, that call a function.

This function then creates an http request on the endpoints we created.

Here is how the function that handle the event of town search look alike.

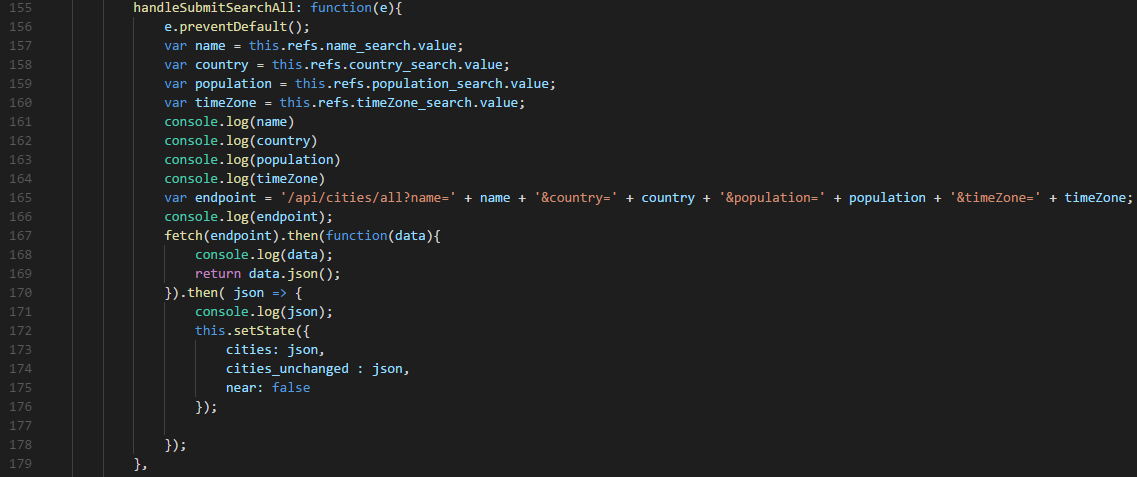


Figure 4: Event function fetching data from the api/cities/all endpoint

Once those data collected with the API, we display them as table list in HTML.



Figure 5: HTML code to display data in the form of an array

Finally, for the design, we used Bootstrap and CSS.

Here is the final rendering of the application:

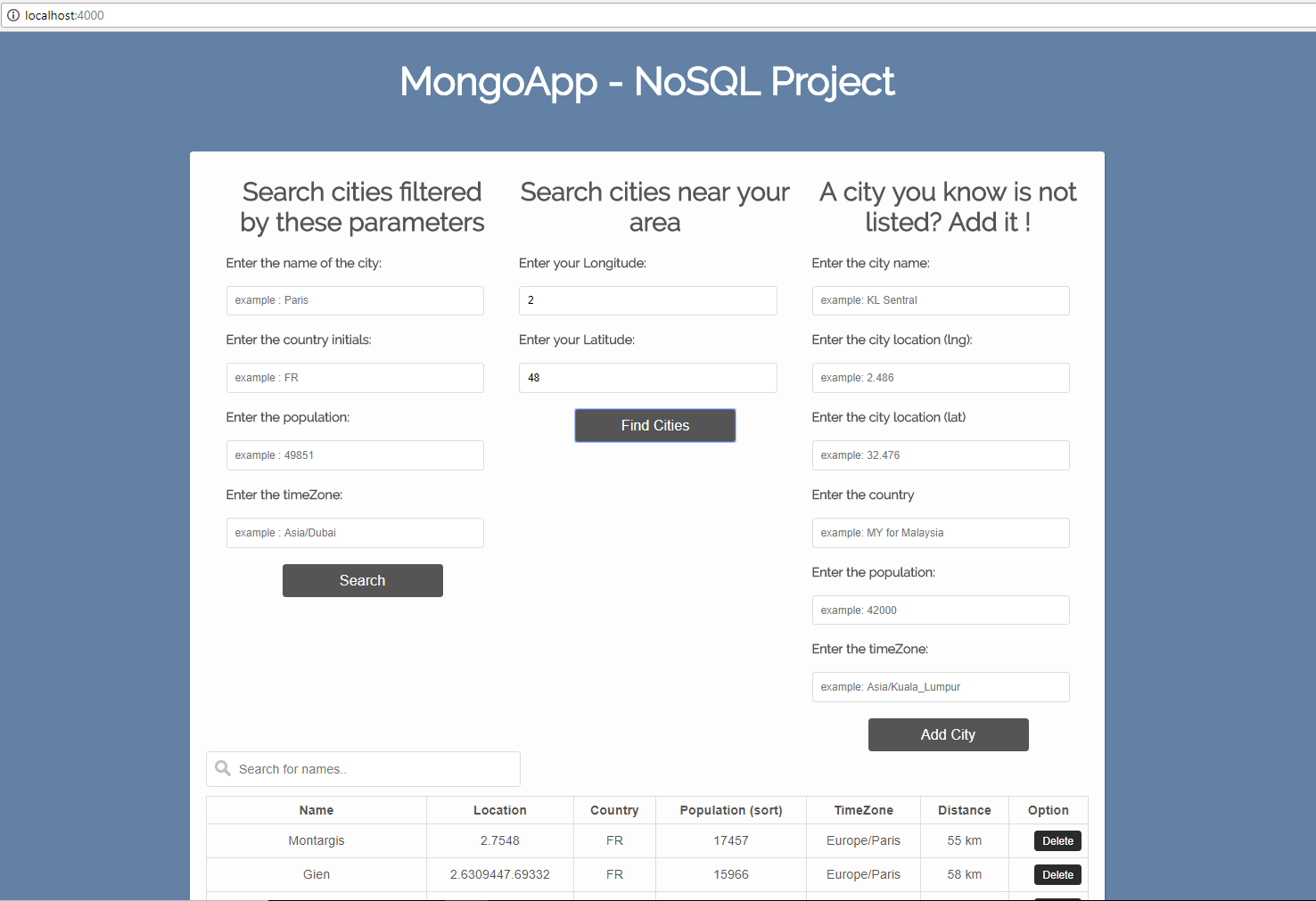


Figure 6: Final result of the web application

# ****Conclusion****

This project led us to work with NoSQL databases, to understand how to use them and to deploy them in a real application. It really helped us to practise what we have studied in Advanced Topics on NoSQL.

To improve this project, we could make the Web application more open to the user, let him more freedom with the querries. We could also change the database software, use Elasticsearch for example. With it we could simplify the querries and even get graphical display of data.

GIT HUB OF THE PROJECT : https://github.com/MiguelRamosF/AppMongo

