



Folorunsho Mojeed Gbemileke

A WEB-BASED SOLUTION FOR OPEN DATA

Technology and Communication

2017

ACKNOWLEDGEMENT

All glory is to the almighty God, who made this journey easy for me through the difficult times. My sincere gratitude goes to my wonderful family for believing in me and supporting through their prayers and motivational support. I am grateful to all my friends, most especially to Ibrahim Olanigan, Akanni Timothy, Gbenga, Julius, Ilke and Hauwa , for providing their constant assistance when I needed them most.

To my project supervisor and mentor, Dr. Chao Gao, who guided me through the working process of this project and tried to get the best out of me through his feedback during the development of the project. This journey wouldn't have been successful without the help of all my teachers, most especially Dr. Ghodrat Moghadampour, Jarmo Makelä and Seppo Makinen for impacting their knowledge on me. I am grateful for the wonderful and challenging times you gave me.

Finally, my sincere gratitude goes to all the teaching and non-teaching staff of this citadel of learning. I appreciate all your efforts in creating a friendly and warm environment in which to learn.

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Information Technology

ABSTRACT

Author	Folorunsho Mojeed Gbemileke
Title	Accessibility of Open Data Files by Developers and Others Using API's
Year	2016
Language	English
Pages	52
Name of Supervisor	Dr Chao Gao

Open data is increasingly gaining popularity in the IT world today where information can be accessed easily through several electronic devices. Research has shown that there are numerous value and vast business opportunities that can be obtained through information extracted from open data files. However, extracting the needed information has not been easy for open data users in recent years due to the availability of appropriate tools.

The main objective of this thesis was to provide a viable solution for developers and users of open data files in gaining easy accessibility to the information stored in the files for efficient use. Open data files CSV and Excel formats were used as a case study. An API in a web-based application format was created as a solution for these files to be easily accessed by its users. MEAN stack and other web technologies were used in developing both the front-end and back-end of the application.

The results showed that the aims and objectives stated for the project were achieved through the solution developed. However, there are still possible ways to improve the project.

Keywords: Open Data, CSV, Excel, Back-end, Front-end, API.

LIST OF FIGURES

Figure 1.URL for the workspace.....	16
Figure 2.My Thesis workspace on c9	17
Figure 3.c9 development environment for the thesis	17
Figure 4.Overview of the project structure in c9	18
Figure 5.Navigation tab on c9	19
Figure 6.c9 compiler environment	20
Figure 7.MongoDB dependencies.....	21
Figure 8.mongo Database collections	22
Figure 9.ExpressJS dependency and version used in this project.....	23
Figure 10.Module declaration	24
Figure 11.Controller function.....	24
Figure 12.Bower Package with Dependencies	27
Figure 13.Architecture of the Application	28
Figure 14.Use Case Diagram for the Application.....	29
Figure 15. Flowchart for back-end program	31
Figure 16.Flowchart for front-end program	32
Figure 17.List of indoor facilities in JSON format	33
Figure 18.List of golf facilities in JSON format	34
Figure 19.API result showing only the names of the facilities	35
Figure 20. Getting accurate information of a facility by name	35
Figure 21.API result showing all the facility types.....	36
Figure 22.List of facilities with a gym facility	37
Figure 23.List of municipalities where facilities are located	37
Figure 24.Home page of the application	38
Figure 25.Indoor facilities presented on a table	38
Figure 26.Using search box to filter dataset with user input.....	39
Figure 27.Golf facilities presented on a table	39
Figure 28.Filtering the dataset with checkbox options	40
 Code Snippet 1.Creating MongoDB connection to localhost server	 30

LIST OF ABBREVIATION

API	Application programming Interface
CSS	Cascading Style Sheet
CSV	Comma-separated Values
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
JS	JavaScript
JSON	JavaScript Object Notation
MEAN	MongoDB, Express JS, Angular JS, Node JS
UI	User Interface
XML	Extensible Markup Language

Table of Contents

ABSTRACT.....	iii
1 INTRODUCTION.....	8
1.1 Background	8
1.2 Objectives.....	9
1.3 Overall Structure of the Thesis	10
2 LITERATURE REVIEW OF THE TOPIC.....	11
2.1 WHAT IS OPEN DATA?.....	11
2.1.1 WHY OPEN DATA?.....	13
2.2 What is API?	14
3 RELEVANT TECHNOLOGIES AND IMPLEMENTATION	16
3.1 Relevant Technologies Used	16
3.1.1 Cloud9 IDE.....	16
3.1.2 Mongo DB	20
3.1.3 Express JS	22
3.1.4 Angular JS.....	23
3.1.5 Node JS.....	25
3.1.6 Hyper Text Mark-up Language (HTML), Bower, and Bootstrap.....	26
3.2 Application Architecture	28
3.3 Use Case Diagram for the Application	28
3.4 Implementation of the API and GUI.....	29
3.4.1 Application Programming Interface Implementation (Back-End).....	29
3.5 Flowchart of the Program	31
3.5.1 Back-end Program.....	31
3.5.2 Front-end Program.....	32
4 TEST AND RESULT.....	33
4.1 Result from Back-end Program	33
4.2 Result from Graphical User Interface (Front-end) Program	38
5 CONCLUSION.....	41
5.1 Challenges	41
5.2 Possible Improvements.....	41

References	43
------------------	----

1 INTRODUCTION

Imagine having access to an enormous business resource. This resource can then be modified through mobile applications and the web in order to create more business opportunities. The concept of open data is to make some data freely available to everyone without any form of restriction, patents or another mechanism of control. This data can be reformed or republished by users for personal or business purpose. /1/

Open Data is becoming a secret to success for smart business leaders around the world. Investors use it to analyze the risks and rewards of different companies in their search for the best opportunities. Company owners use it to understand subtle clues to their brand's reputation and to develop data-driven marketing strategies. And entrepreneurs are using Open Data on weather, housing, transportation, and more to build businesses that provide new services and benefit the public. /1/

In October 2013, McKinsey & Company released a new report on the value of Open Data worldwide. The report calculated that "an estimated \$3 trillion in annual economic potential could be unlocked" through Open Data in sectors like education, energy, and health. This is the largest value put on Open Data so far, and may seem too inclusive; for example, it includes the value of using Open Data to increase the population's earning power by improving education. Regardless, this latest McKinsey report demonstrates the growing business interest in Open Data and excitement about its potential. /3/

1.1 Background

Information Technology (IT) is moving towards an era where almost all information can be accessed from mobile phones through various mobile applications. Nowadays web applications and mobile applications communicate

with each other through APIs but are currently facing some overload problems when handling large data. Some of the problems faced by users working with open data in CSV and excel format are as follows; the exact information needed from the files are difficult to extract as it contains an enormous set of data, and accessing these data through mobile phone devices tends to be strenuous at the moment for users and developers.

This is why the thesis is aiming at exploring the advantages and benefits in making the accessibility to open data easy for users through Application Programming Interface (API) on a web-based platform. The API to be built for this project will work on some specific set of Comma Separated Value (CSV) and Microsoft Excel files.

1.2 Objectives

The objective of this project is to create an Application Programming Interface (API) in a web-based application format in order to make the accessibility of open data (CSV and excel files) more convenient for users of the files to navigate through without any complications. The aim of this project is to also explore the difficulties in accessing these files without API's and also to create a solution through the API created in showing the benefits that are available in the files.

The advancement made by engineering and technology in recent years has made it possible for information to be easily accessed by almost everyone in all part of the world, through their various electronic devices such as laptops, tablets, and mobile phones. Therefore, making the route to these sets of data or information feasible for users is the priority of the developer. Open Data is *“going to help launch more start-ups. It's going to help launch more businesses. . . . It's going to help more entrepreneurs come up with products and services that we haven't even imagined yet”* —(President Barack Obama, May 9, 2013). Since open data is foreseen to be the future for more business opportunities, research and information distribution for people, the difficulties which are currently faced in accessing the files in this two different formats (CSV and Excel).

In summary, the following are the main objectives of what is expected of this project:

- Easy accessibility to open data files with mobile phones or computers.
- Converting the CSV and Excel files into user-friendly files i.e files will be ready for easy use without users worrying about converting the files or using special devices in accessing the files.
- To help developers and users in creating new business ideas with this data(s).
- To reduce the difficulty faced by developers in converting the files from their original formats (CSV and Excel).

1.3 Overall Structure of the Thesis

The basic structure of the thesis is described as follows:

Chapter 1 describes the motivation and objectives of the thesis. Chapter 2 gives an in-depth explanation of the related topics and background knowledge for developing the project. Chapter 3 explains the reasons and benefits of the technologies used in developing the project. Chapter 4 focuses on the implementation of the project, while chapter 5 shows the test procedures and results. Conclusions on the project are made in chapter 6.

2 LITERATURE REVIEW OF THE TOPIC

In this chapter an in-depth description of the main topic will be discussed in detail, i.e the concept of open data and Application Programming Interface (API).

2.1 WHAT IS OPEN DATA?

Open data can simply be described as some sets of data which is freely available to everyone to use and republish to their best usage, without government restrictions, copyrights diminution, patents or other mechanisms of restriction. The doctrine behind open data has been longer established (for example in the Mertonian tradition of science), but the term “open data” itself is recent, gaining popularity with the advancement of internet and World Wide Web and especially, with the launch of open-data government's initiatives such as Data.gov and so on. In 2009, open data started to become more visible in the mainstream, with various governments (such as the USA, UK, Canada and New Zealand) announcing new ideas towards opening up their public information. /2/

The Open Data development started with majority rule objectives, powered by the idea that governments ought to make the information or data they gather accessible to the citizens who have paid to gather it. However, notwithstanding its social benefits, Open Data has made tremendous openings for new business ideas, which is one of the areas this research is about to uncover. It is worth recalling that the internet itself started as an administration supported activity, the ARPANET, made by the Advanced Research Projects Agency.

Open Data incorporates government, state, and local information; scientific information publicized by scientists, information that organizations release about their own operations, user reviews composed by conventional individuals, and any sort of information that can be found on Google or other websites. By utilizing these numerous sorts of Open Data:

- Government and private organizations are giving new, centralized information assets for business improvements. For example, Hri.fi is a web service that provides fast and easy access to open data sources and files in XML, CSV and XSL formats between the cities of Helsinki, Espoo, Vantaa, and Kauniainen. These datasets are open and accessible for anybody to use for free.
- Companies are becoming more open about their activities, by disclosing both government-required and un compelled datasets to the public freely. This initiative has helped companies to attract new investors, and improve the company's brand image.
- Open data is aiming to help consumers through the decision-making process by providing them with varieties of information for all kind of products and services available online. It allows the consumers to have access to detailed information about their choice and choose from the best options related to those choice(s).

The openness of data basically means that data has been made as easy as possible for anyone to use. The level of openness can vary in different aspects. The most important areas are technical accessibility, free access and reuse permitting licensing. The more the use of data is restricted, on purpose or accidentally, the less open it is./5/

- **Public domain:** The data has to be communal information in order for it to be opened for public use. Sensitive or otherwise compromising information cannot be published due to the protection of individuals' rights to privacy and the general safety of the society.
- **Technical accessibility:** Data has been published in a format which computer programs can understand and which makes it easy to use in the development of new web services. Information in PDF documents or on most HTML web pages is often hard to use in derivative works. In a sense, the information stays locked inside the document. Publishing data in CSV or XLS formats or giving access to the original data source through an application programming interface (API) are better solutions.

- **Free access:** Data can be accessed free of charge. Free access is especially important for people exploring a data source for the first time. Free access enables experimentation and innovation with data without the need for budget bureaucracy.
- **Reuse permitting licensing:** The publisher permits reuse and clearly expresses it in a license accompanying the data. Otherwise finding out the terms of use can be so strenuous that data is left unused.

2.1.1 WHY OPEN DATA?

Open data is a large resource which is largely still untapped. Many organizations and individuals gather different forms of information for their personal use. Therefore, due to the centrality and quantity of this data, making it open for the public will be of more use. /1/

There are several areas which one can expect open data to more useful, and examples of where and how it has been used exist. There are numerous groups of individuals and organizations who can profit from the availability of open data, including governments itself.

Some of the areas in which open government data is creating values are as follows:

- Self-empowerment
- Improved efficiency of government services
- Innovation
- Improved or new private products and services
- Transparency and democratic control

The importance of open data in the economic sector of many countries is on the increase. Several studies have shown the economic value added by open data to the EU countries is estimated around 10billions of Euros annually. New start-up companies and individuals are re-using open data to improve their businesses. For instance, Google translate uses the enormous volume of EU documents that

appear in all European languages to train the translation algorithm, thus improving its quality of service.

This numerous untapped potential can only be achieved if public data are made open without any legal, financial or technological restrictions.

2.2 What is API?

An Application Programming Interface (API) is a set of routine definitions, protocols, and tools for developing software and applications. It is a set of building blocks which can be assembled together by developer or programmer in order to perform a certain task which it is designed to execute. An API may be for a database system, operating system, computer hardware or a web-based system. It enumerates how the software components should interact with the User Interface (UI).

One of the purposes of application programming interface is that it makes the work of programmer much easier by allowing the developer to use some set of technologies in developing applications. For example, an API for copying files from one location to the other will only require the developer to call the function name into his program without putting more effort into building the function itself.

There are different areas in which APIs can be used by a developer; some of these areas are through;

- **Libraries and frameworks:** These sets of application programming interfaces exist in form libraries that share the same programming interface. It allows programs written in one language to gain access to another program with a different programming language. For example, a program written in Scala can use any JAVA API because both are compiling into compatible bytecode.

- **Remote APIs:** Remote APIs gives a programmer the possibility of communicating with a remote database or server. This communication is established through some specific protocols and standards in order to make the technologies work together. For examples, there is Java database connection in Java which allows a developer to query and modify any part of the datasets in a database through a java application that requires a database connection. Some part of this project falls into this category APIs.
- **Web APIs:** Web APIs are defined interfaces through which communication occur between a server and an application that uses it properties. Using a web API usually involves declaring a set of Hypertext Transfer Protocol (HTTP) request message, along with its response counterpart, which is usually in JavaScript Object Notation (JSON) or in an Extensible Markup Language (XML). The response format used in building this project API application is JSON.

3 RELEVANT TECHNOLOGIES AND IMPLEMENTATION

In this chapter the technologies used in developing the application will be discussed in details as well as the architecture of the application in order to understand how the application works.

3.1 Relevant Technologies Used

This project was developed with cloud9 IDE, Mongo DB, Express JS, Angular JS, and Node JS technology. However, other web development technologies such as Hyper Text Mark-up Language (HTML), Bower, Bootstrap, and LiveReload were also used.

3.1.1 Cloud9 IDE

Cloud9 IDE (Integrated Development Environment) is an online programming and a LINUX development environment which is published as an open source. It provides a coding environment with a pre-configured workspace which is stored in the cloud and allows developers to push their codes to GitHub. This IDE is entirely written in JavaScript and uses Node JS on the back-end. Cloud9 IDE supports different programming languages including, Python, JavaScript, Ruby, Node.js, Angular.js, C++, PHP and so on.

In order to use the editor, a workspace needs to be created. A sample of the workspace created for this project is shown in Figure 1;

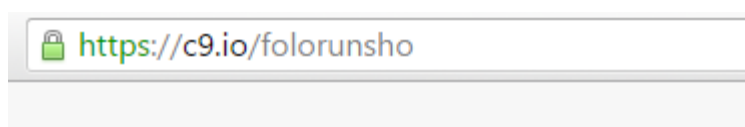


Figure 1.URL for the workspace

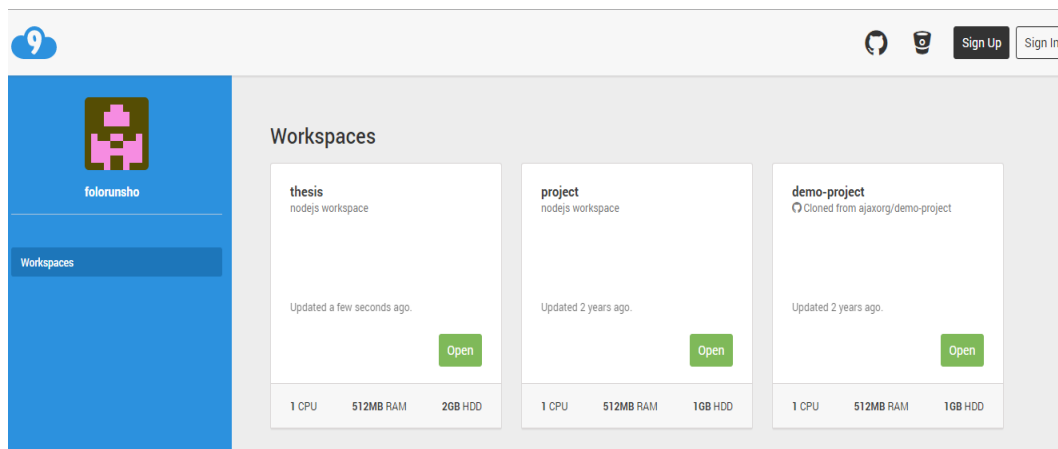


Figure 2.My Thesis workspace on c9

In order to get into the development environment, the open tab in the thesis workspace which is a Node.js environment is to be clicked. Figure 3 shows the programming environment;

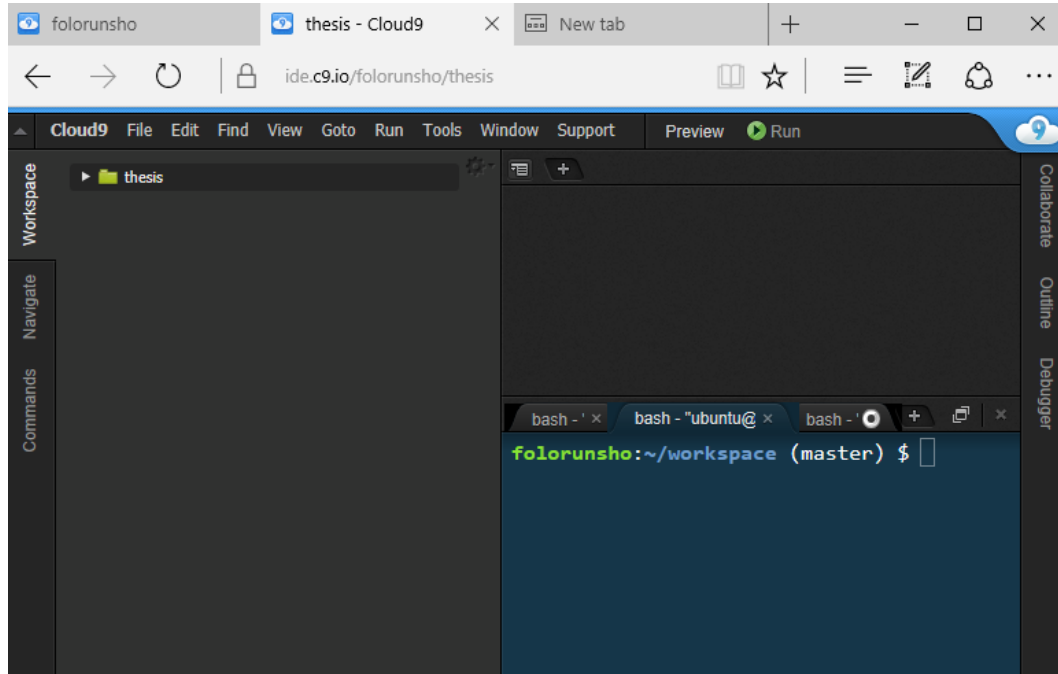


Figure 3.c9 development environment for the thesis

The overview of the project structure is shown in Figure 4.

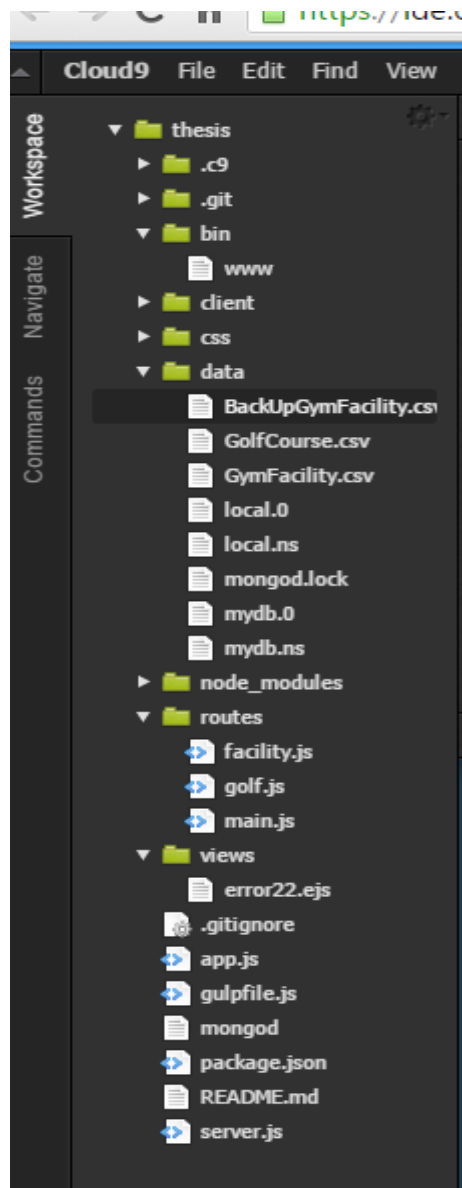


Figure 4. Overview of the project structure in c9

The Navigation tab on the workspace shows the root structure of all the files used in the development of the project. Figure 5 shows a sample of it.

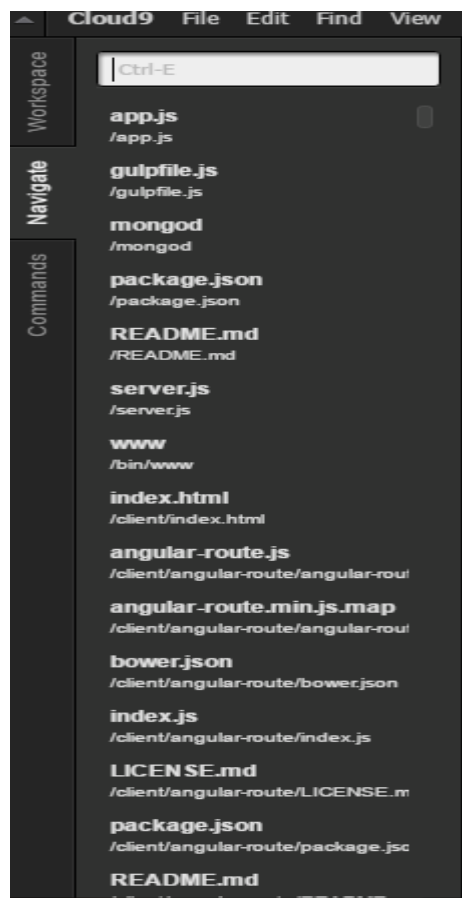


Figure 5. Navigation tab on c9

Cloud9 IDE uses a LINUX based compiler for its files. Only LINUX commands work in this environment as shown in Figure 6.

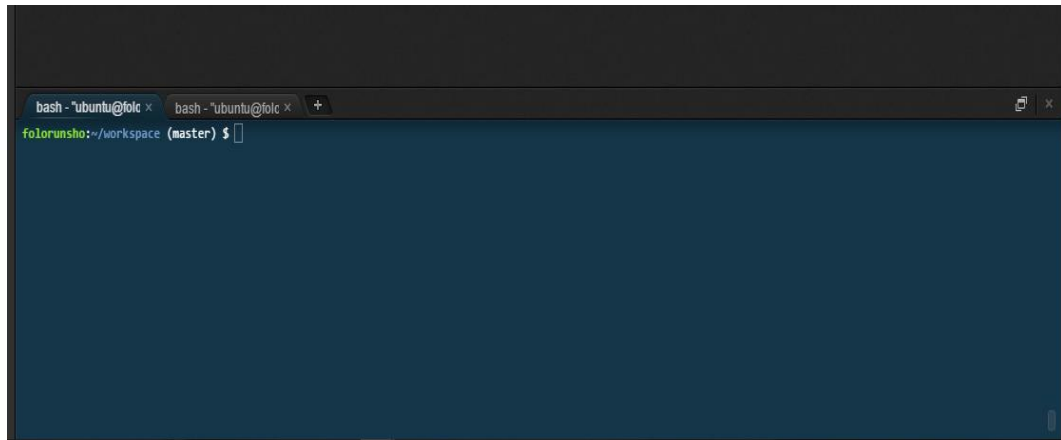


Figure 6.c9 compiler environment

MEAN technology was used in developing this project. It is a technology built on JavaScript framework and it is used for developing both server-side and web applications. It is basically used for building fast, maintainable and durable web applications and helps to keep the application in a well organized form.

The development of this project was segmented into two different parts. Those of Back-End and Front-End. Both of these segments can actually be developed on the c9 environment. However, due to some difficulties in installing some of the MEAN technology dependency files needed for the front-end development, only the back-end was successfully developed on the c9 IDE. A further explanation on MEAN technology will be given in the following paragraphs.

3.1.2 Mongo DB

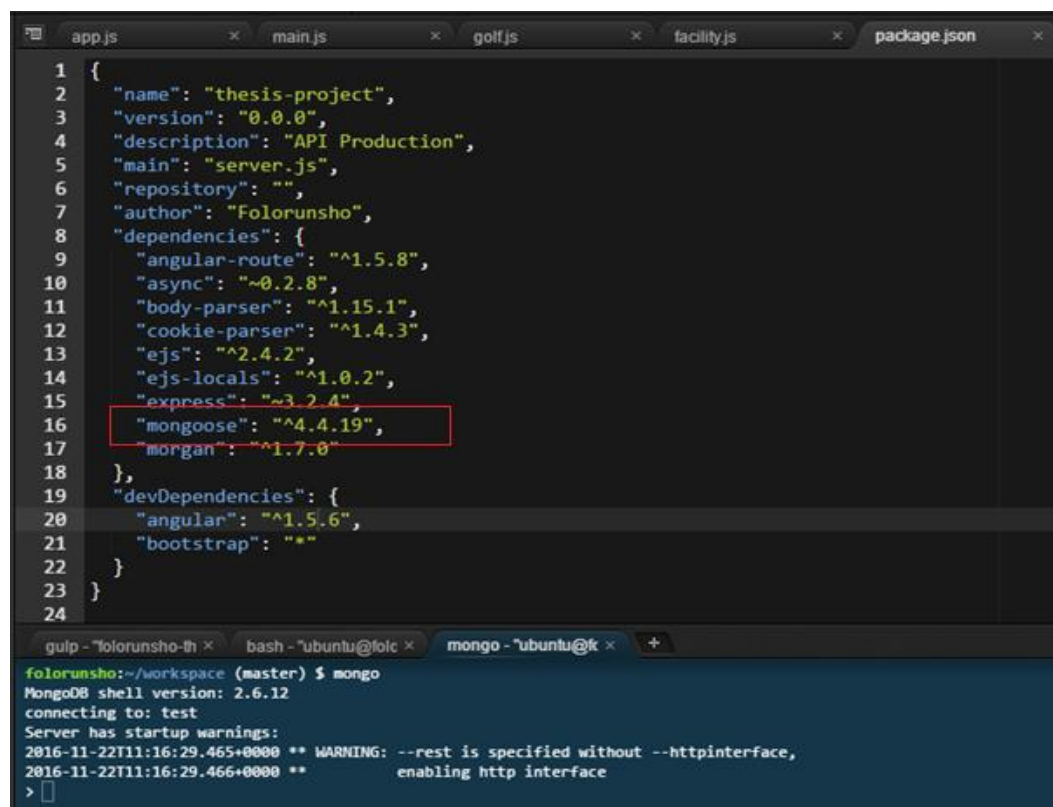
Mongo DB is an open source database management scheme that uses a document-oriented database program written in C++. It is a NoSQL type of database management which means its data are not stored using tables and rows; rather, they are stored using an architectural system of collections and documents. These documents comprise of some sets of key-value pairs and are the basic units of data

in MongoDB. Collections contain sets of documents and functions as the equivalent of relational database tables.

MongoDB is known for its dynamic design, allowing the documents stored in its collections to have different structures. Documents are stored in the database in a BSON (Binary JSON) format, which provides a binary format of JSON-like documents. MongoDB is known for its high-performance quality in storing an extensive amount of data, which is why it has been used for this research due to the amount of data involved.

Cloud9 IDE comes with a MongoDB dependency which only requires installing the latest version of mongo on the development environment for ready usage. These dependency files are stored in the *package.JSON* file on the thesis folder.

Figure 7 shows the version of Mongo used for this project.



```

1 {
2   "name": "thesis-project",
3   "version": "0.0.0",
4   "description": "API Production",
5   "main": "server.js",
6   "repository": "",
7   "author": "Folorunsho",
8   "dependencies": {
9     "angular-route": "^1.5.8",
10    "async": "~0.2.8",
11    "body-parser": "^1.15.1",
12    "cookie-parser": "^1.4.3",
13    "ejs": "^2.4.2",
14    "ejs-locals": "^1.0.2",
15    "express": "~3.2.4",
16    "mongoose": "^4.4.19",
17    "morgan": "^1.7.0"
18  },
19  "devDependencies": {
20    "angular": "^1.5.6",
21    "bootstrap": "*"
22  }
23 }
24

```

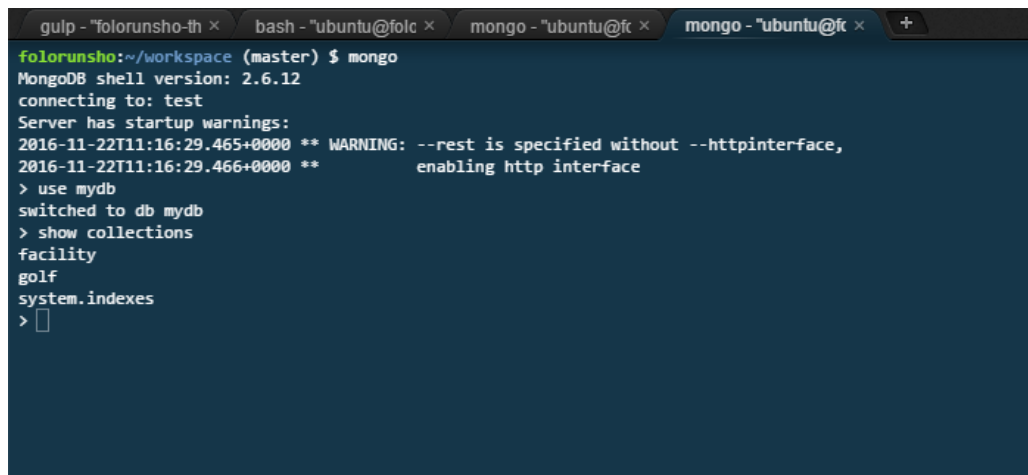
```

gulp - "folorunsho-th" x bash - "ubuntu@foic" x mongo - "ubuntu@k" x +
folorunsho:~/workspace (master) $ mongo
MongoDB shell version: 2.6.12
connecting to: test
Server has startup warnings:
2016-11-22T11:16:29.465+0000 ** WARNING: --rest is specified without --httpinterface,
2016-11-22T11:16:29.466+0000 ** enabling http interface
>

```

Figure 7. MongoDB dependencies

Figure 8 shows the collections of data stored the mongo database.



```

gulp - "folorunsho-th x  bash - "ubuntu@folc x  mongo - "ubuntu@fc x  mongo - "ubuntu@fc x  +
folorunsho:~/workspace (master) $ mongo
MongoDB shell version: 2.6.12
connecting to: test
Server has startup warnings:
2016-11-22T11:16:29.465+0000 ** WARNING: --rest is specified without --httpinterface,
2016-11-22T11:16:29.466+0000 **          enabling http interface
> use mydb
switched to db mydb
> show collections
facility
golf
system.indexes
> 

```

Figure 8.mongo Database collections

The following commands were used cloud9 IDE command terminal to access the files stored in the mongo database.

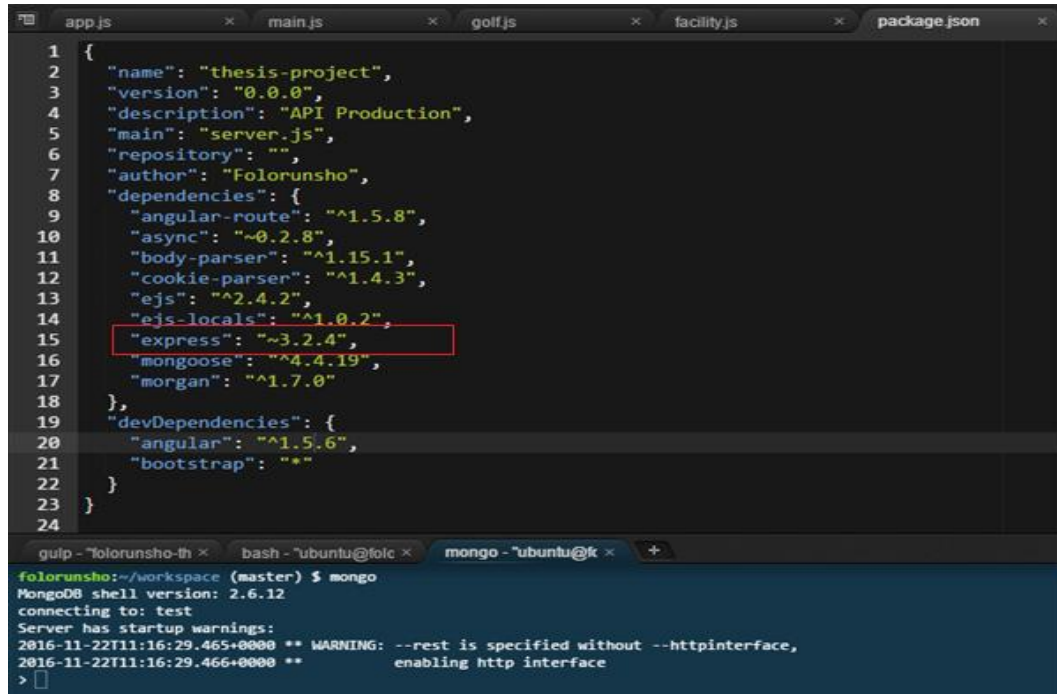
- **\$mongo** : used for changing the terminal directory to mongo.
- **Use mydb** : This command changes the directory to “mydb” which is a name given to the database used for this project.
- **Show collections:** it shows the list of all the documents stored in mydb.

More of the commands used will be discussed comprehensively in the next chapter.

3.1.3 Express JS

Express.js is a web-application framework for Node.js that allows JavaScript to be used outside the web browsers, for creating network and web applications. Express Js is used as the backend part of this project, together with MongoDB

database. It helps in organizing the application better through its plug-in and routing systems. The version of Express JS used in this project is highlighted as shown in Figure 9.



```

1  {
2    "name": "thesis-project",
3    "version": "0.0.0",
4    "description": "API Production",
5    "main": "server.js",
6    "repository": "",
7    "author": "Folorunsho",
8    "dependencies": {
9      "angular-route": "^1.5.8",
10     "async": "~0.2.8",
11     "body-parser": "^1.15.1",
12     "cookie-parser": "^1.4.3",
13     "ejs": "^2.4.2",
14     "ejs-locals": "^1.0.2",
15     "express": "~3.2.4",
16     "mongoose": "^4.4.19",
17     "morgan": "^1.7.0"
18   },
19   "devDependencies": {
20     "angular": "^1.5.6",
21     "bootstrap": "*"
22   }
23 }
24

```

```

folorunsho:~/workspace (master) $ mongo
MongoDB shell version: 2.6.12
connecting to: test
Server has startup warnings:
2016-11-22T11:16:29.465+0000 ** WARNING: --rest is specified without --httpinterface,
2016-11-22T11:16:29.466+0000 ** enabling http interface
>

```

Figure 9.ExpressJS dependency and version used in this project.

3.1.4 Angular JS

Angular.js is a front-end JavaScript framework that is used for creating dynamic and responsive single-page web application. The framework improves the traditional way of presenting data with HTML through two-way data-binding synchronization of models and views. Angular.js uses the MVC (Model, View, and Controller) pattern to control and present data in an application.

Modules and controllers are very important elements of any angular application. A module is a like a container or the main method for different parts of an application which consist of the controller, services, directives filters and so on. For example, Figure 10 shows how a module is defined for the front-end part of this project.

```
var app = angular.module('dapp', ['ngRoute']);
```

Figure 10. Module declaration

Notice that the expression contains two arguments; the first argument **'dapp'** is the name of the module while the latter **['ngRoute']** usually is an array of the dependencies to be used in building the application.

A controller in angular.js is a JavaScript function. The job of a controller is to build a Model for the View to display. This controller can be called through its name in any part of the program whenever its functions are needed. An Example of a controller function used in the project is shown in figure 11.

```
app.controller('FacilitiesController', function($scope, $http) {
    $scope.showID = true;
    $scope.showName = true;
    $scope.showMun = true;
    $scope.showPro = true;
    $scope.showYear = true;

    //function to fetch JSON FILE from facility url
    $scope.fetchFacilities = function() {
        $http.get("https://thesis-folorunsho.c9users.io/facility")
            .then(function(response) {
                console.info('Facilities:', response.data);
                $scope.facilities = response.data;
            });
    };
});
```

Figure 11. Controller function

Directives are also some of the key components of an angular.js application. It allows the developer to reuse custom HTML-like attributes or tags that define data bindings and components behaviour.

Some of the most commonly used directives are;

- **ng-app**: initialization of the root element of an angular application.

- **ng-model:** used for two-way data binding between the view and the scope.
- **ng-repeat:** used for initializing an element per item from a group of collections.
- **ng-view:** it is the directive responsible for routing in the application.
- **ng-controller:** it specifies the JavaScript controller class that evaluates HTML expressions.
- **ng-class:** it allows the class attributes to be loaded dynamically on the HTML page.
- **ng-show & ng-hide:** it shows or hides the value of an element depending on the value of a Boolean expression.

Benefits of using angular to develop an application are;

- It's ability for dependency injection.
- Two- way data binding process, i.e changing the model and view automatically or through updates.
- It is easy to maintain and test the application through unit testing.

Angular JS technology was used to develop the User Interface of the project. This technology helps in developing a responsive single page application, that is, each page of the application has its own content (a JSON file) whenever a link or page is loaded. The technology has several plug-in packages which make it easier to design and develop the Graphical User Interface. It also allows the incorporation of other coding technologies such as JavaScript, jquery, and bootstrap.

The response time required in loading up an Angular JS web page on a web browser is significantly faster than other web development technologies. This project involves a series of heavy data processing which cannot be easily processed with other web development technologies.

3.1.5 Node JS

Node.js is an open-source JavaScript runtime environment for developing different types of applications. It is a server-side platform that uses Google chrome JavaScript engine (V8 Engine). The introduction of node.js was as a result of the difficulties faced by developers in developing server-side programs.

Node.js is basically used for developing fast and efficient mobile or web applications such as Data Streaming Application, Single Page Application, and JSON APIs Applications. One of the difficulties with applications running heavy information or data is buffering. However, the key feature for the introduction of node.js was to eliminate the buffering aspect of data and load them as fast as possible.

3.1.6 Hyper Text Mark-up Language (HTML), Bower, and Bootstrap

Hyper Text Mark-up Language (HTML) is a standard mark-up language for developing web pages and web applications. It uses a combination of cascading style sheet (CSS) and JavaScript codes to change the behaviour of web pages and render them into multimedia web pages.

Bower is a package management system for web components that comprises of JavaScript, HTML, CSS and so on. It uses a package registry mechanism to publish new packages. These packages are free to download from bower server with git repository without authentication or user management on the bower registry. Bower gives more flexibility in installing and managing all kinds of web components needed and handles the dependencies properly. Figure 12 shows the contents in the bower component as well as their respective dependencies.

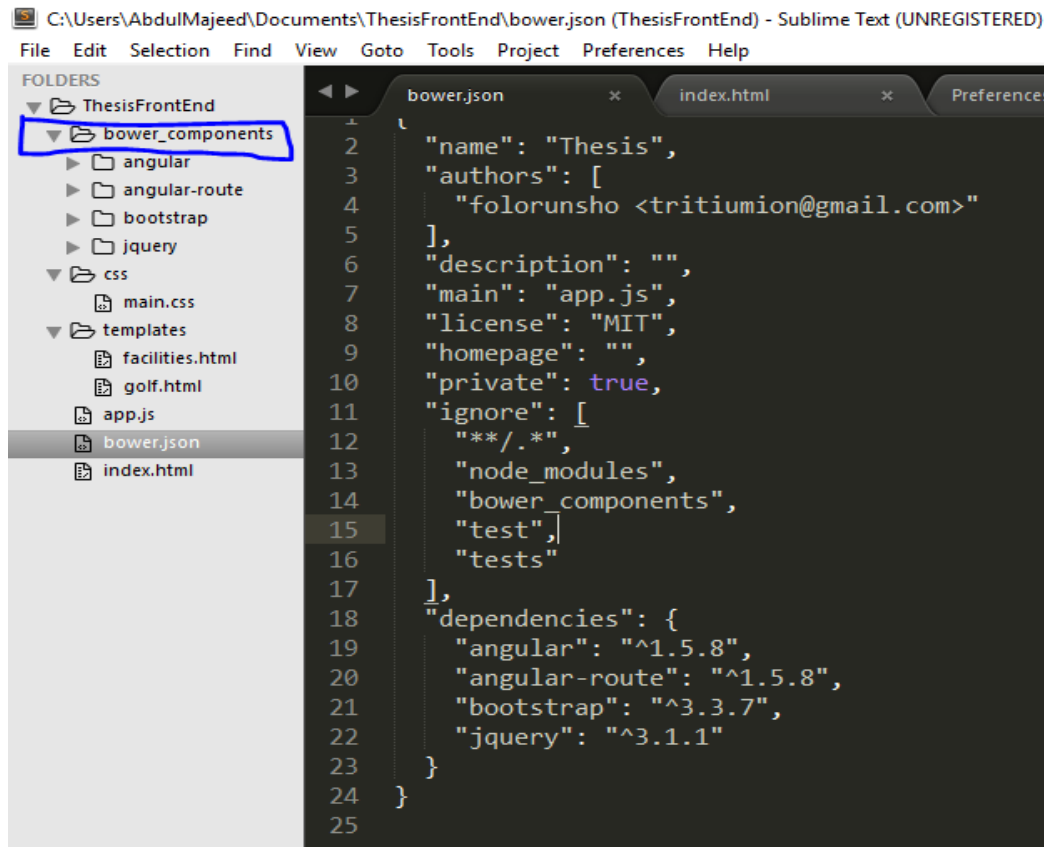


Figure 12. Bower Package with Dependencies

Bootstrap is basically a front-end JavaScript framework for developing mobile applications and responsive web applications. Bootstrap makes the front-end easier to develop without caring so much about the details of CSS and JavaScript.

In summary, the development of this project was divided into two parts; the Back-end and the Front-end. The API and web application were developed using MEAN stack development and other web-application technology.

3.2 Application Architecture

The architecture of the project (Figure 13) shows the process involved in getting the data from the API to the Graphical User Interface (GUI).

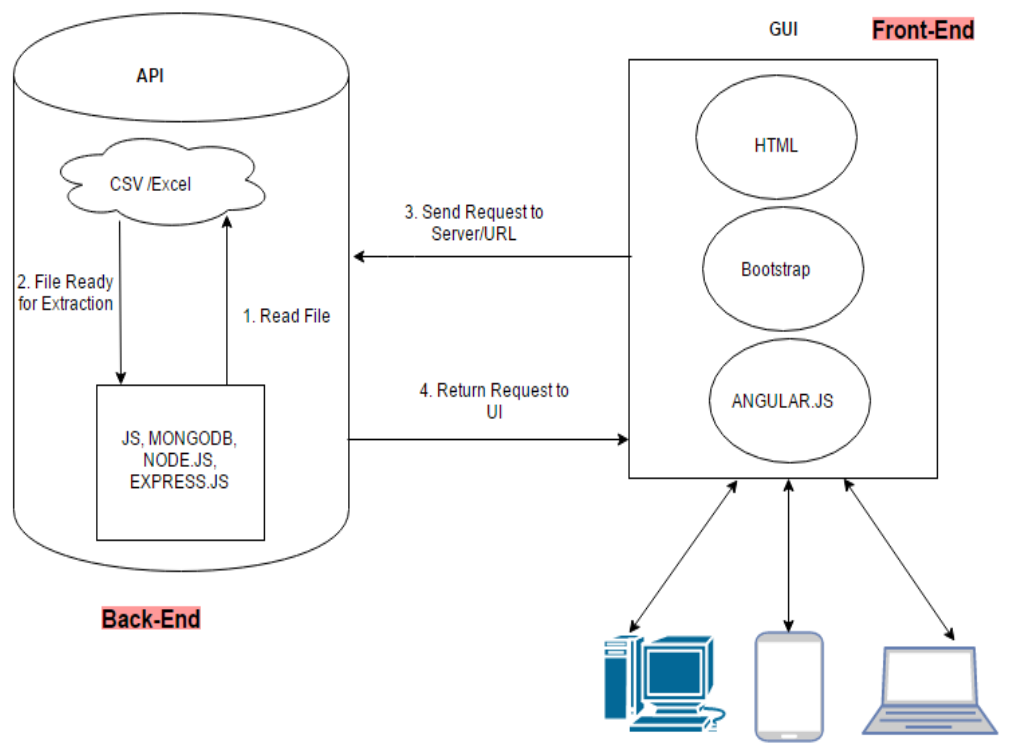


Figure 13. Architecture of the Application

The open data files are extracted and processed through the API program from the back-end. Each API result is connected to a Uniform Resource Locator (URL) or route, which is used as a server for the front-end application. Once the server connection is established from the front-end program, the API result is displayed on the web page whenever a certain request is made by the user.

3.3 Use Case Diagram for the Application

The use case diagram in figure 14 shows the functions of the application. Also, it explains the interaction between the actors or users of the application as well as the relationship between different functions of the application.

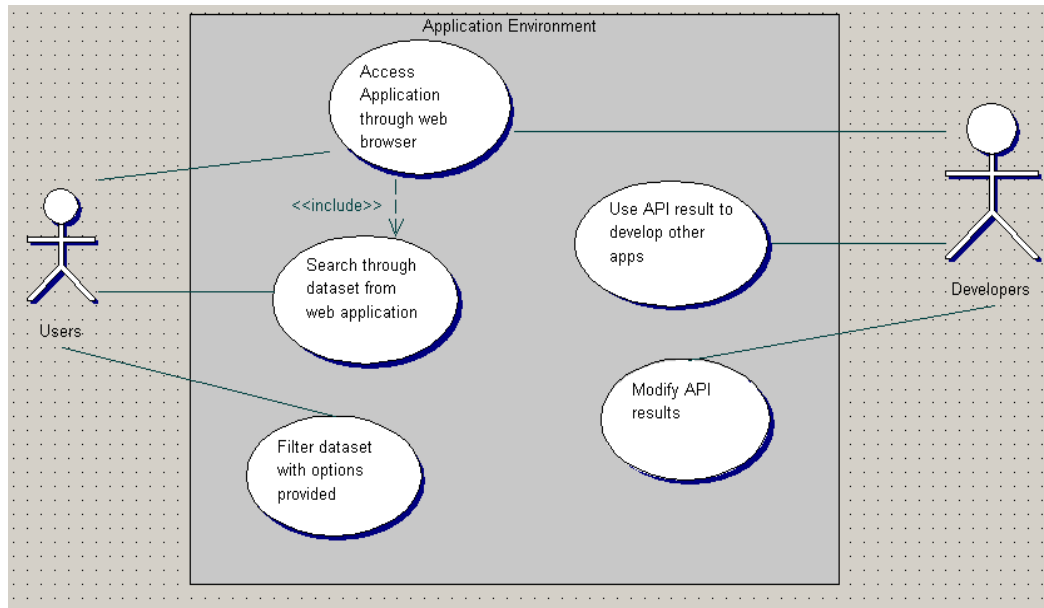


Figure 14. Use Case Diagram for the Application

3.4 Implementation of the API and GUI

In this section the process involved in the development of the Application Programming Interface (Back-end) and Graphical User Interface (Front-end) will be illustrated.

3.4.1 Application Programming Interface Implementation (Back-End).

The technology used in the development of this part of the project (ExpressJS) was discussed in detail in the previous chapter. However, the integration of these technologies in the development of this API application will be shown in the following figures with more explanation.

Firstly, MongoDB database and HTTP connection were established in order to retrieve information needed from the files through the API created. This database connection will ensure that the GUI has a valid HTTP connection to fetch data from. In code snippet 1 Mongoose and Express functions were used to create a connection between the database files and API program as shown below.

```

/* This file include the connection to the data base */

/* Show an event in the log if successful or not*/

var express = require('express'),

    mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/mydb', function(error) {

    if (error) {

        console.log('Cannot connect due to %s', error);

        return;

    }

    console.log('Mongo DB connected');

});

```

Code Snippet 1. Creating MongoDB connection to localhost server

The next step was to import and convert the CSV files simultaneously into the database. The following script (snippet 2) shows how the CSV files were converted into a JSON file from a command prompt window.

```

mongoimport --db mydb --collection facility --type csv --file data/GymFacility.csv --headerline
mongoimport --db mydb --collection golf --type csv --file data/GolfFacility.csv --headerline

```

Code Snippet 2. Importing CSV files to MongoDB and converting to JSON

The “mydb” highlighted in the above snippets is the database name where the files are been stored, while “facility” and “golf” are the collection names because MongoDB stores its data as a set of collections. “data/GymFacility” represents the file name. The database structure in Figure 8 shows that the “facility” and “golf” files are stored successfully into Mongo database.

3.5 Flowchart of the Program

In this project there is a back-end and front-end program. The back-end program generates an API result which is transferred to the front-end or User Interface of the application.

3.5.1 Back-end Program

Figure 16 shows a flow chart diagram of how the back-end program works.

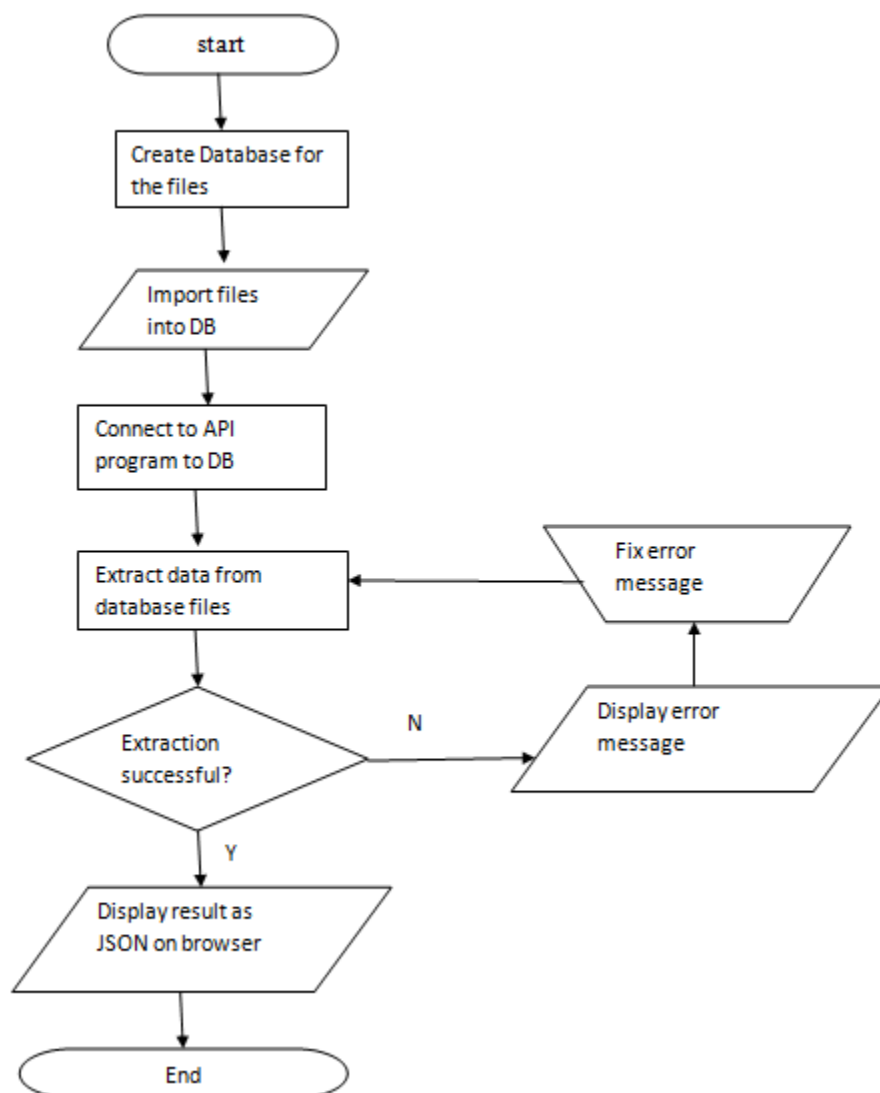


Figure 15. Flowchart for back-end program

3.5.2 Front-end Program

Figure 16 shows a flow chart diagram of how the front-end program works

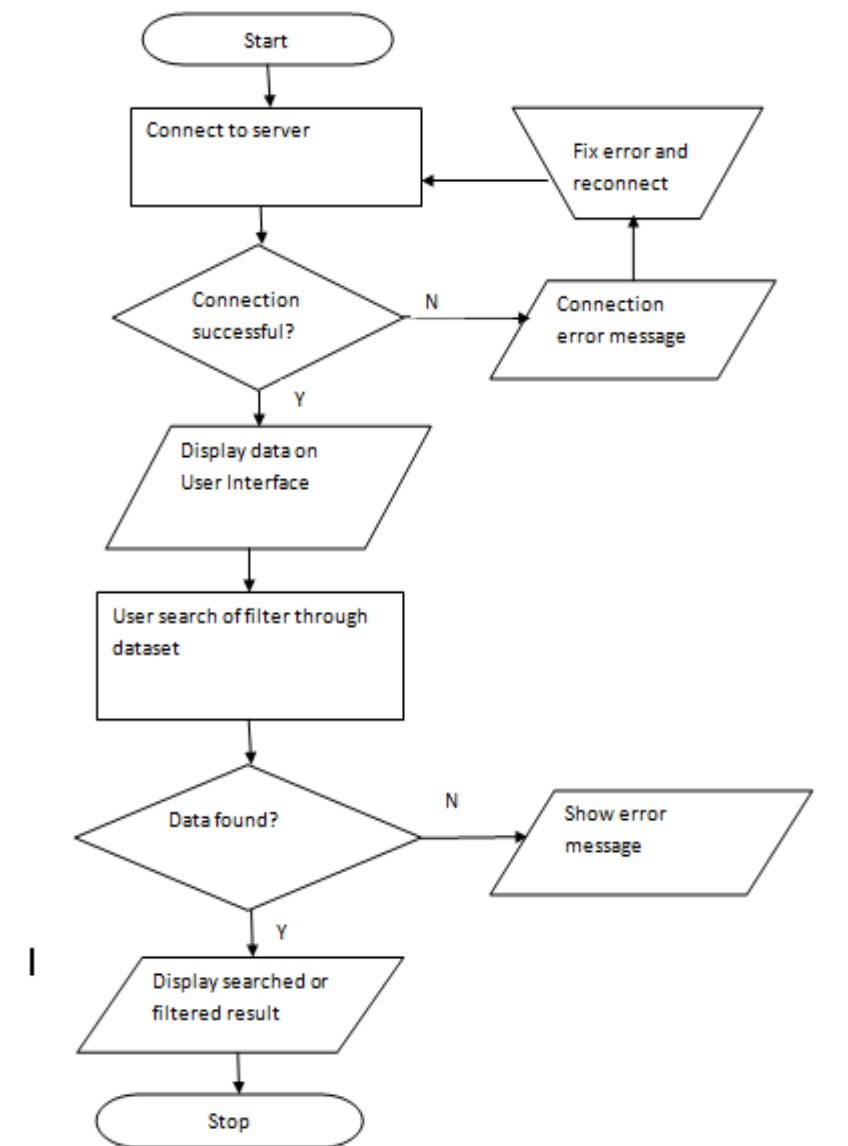


Figure 16.Flowchart for front-end program

4 TEST AND RESULT

The results achieved through the implementation phase of the project will be shown and discussed in this chapter. The results achieved from the back-end program will be shown first, followed by the front-end results.

4.1 Result from Back-end Program

The following figures (17-23) show the results achieved from the facility and golf API program.

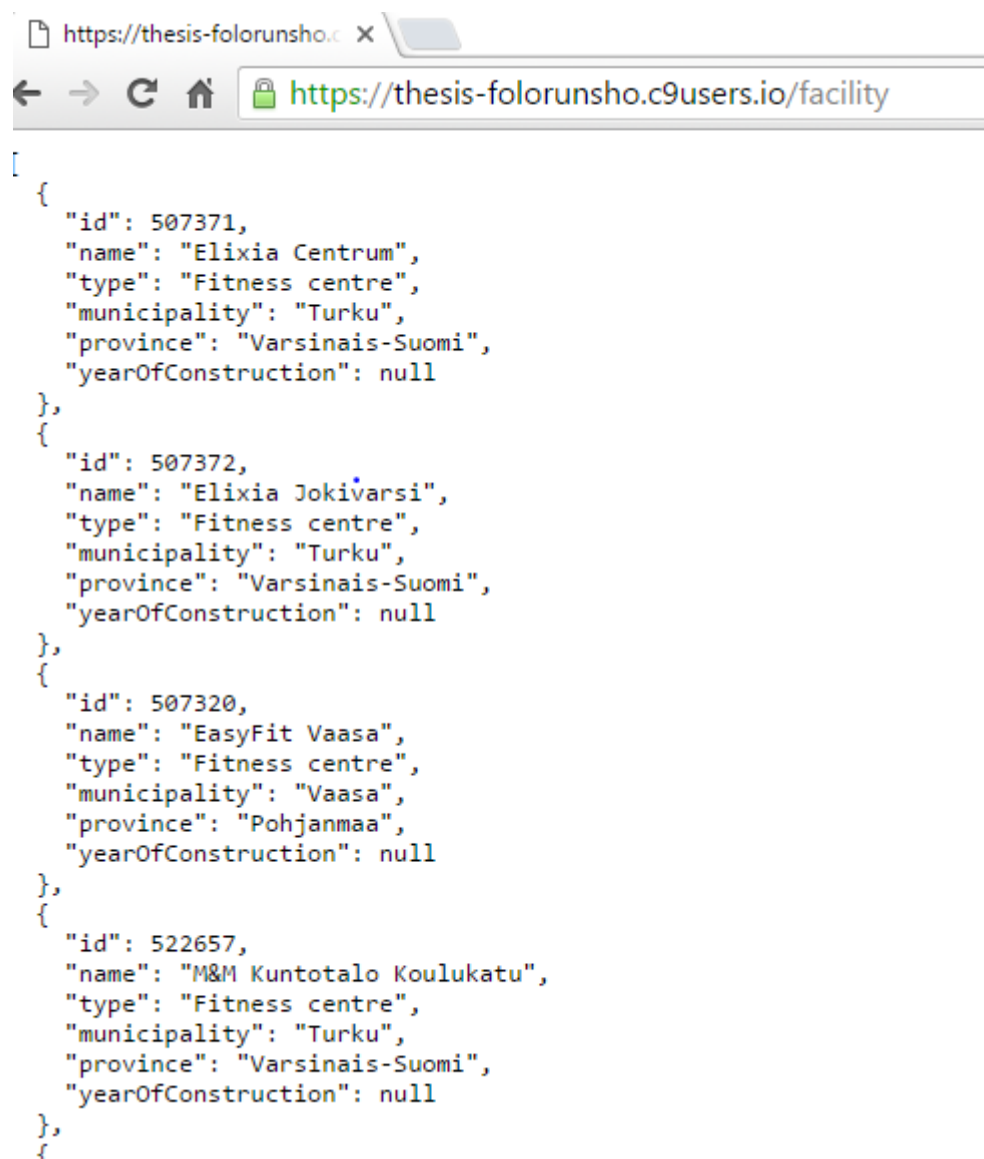
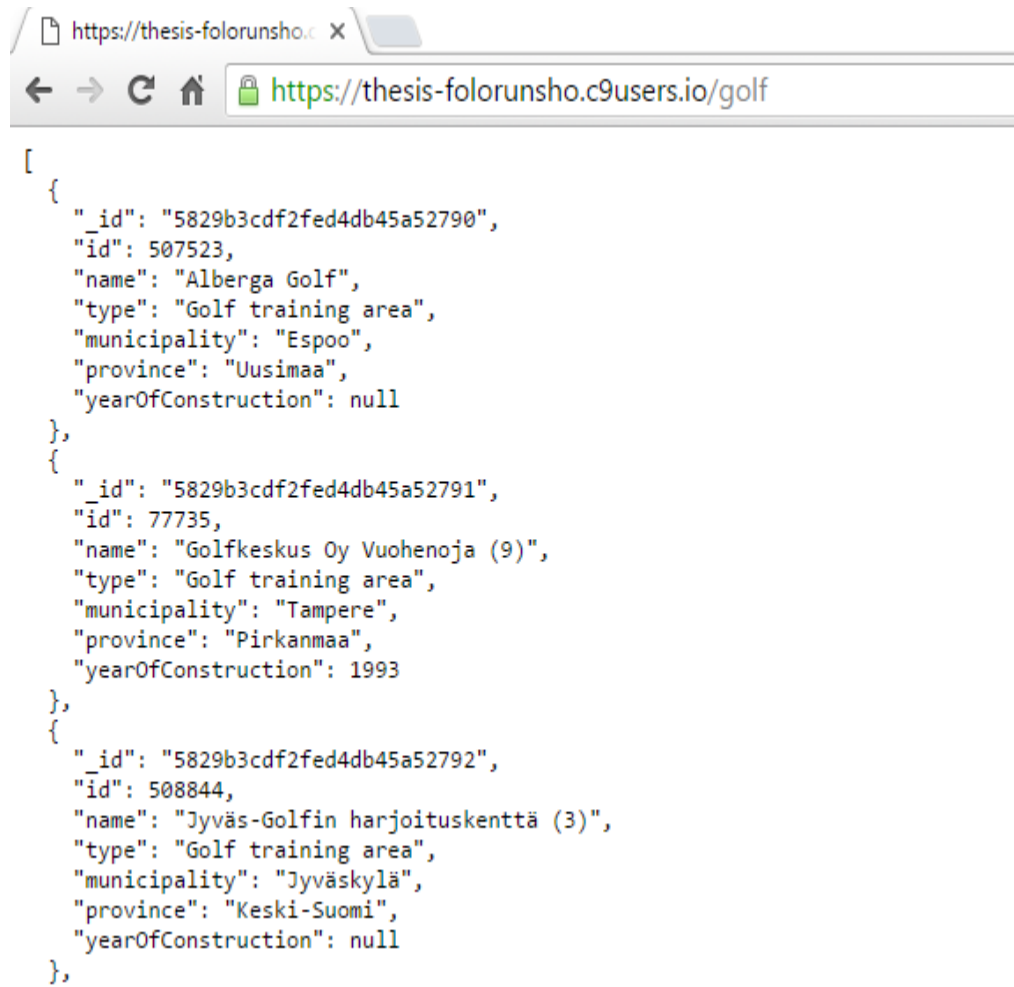


Figure 17.List of indoor facilities in JSON format



The image shows a web browser window with the address bar displaying `https://thesis-folorunsho.c9users.io/golf`. Below the browser window, a JSON array is displayed, containing three objects representing golf facilities. Each object has fields for `_id`, `id`, `name`, `type`, `municipality`, `province`, and `yearOfConstruction`.

```
[
  {
    "_id": "5829b3cdf2fed4db45a52790",
    "id": 507523,
    "name": "Alberga Golf",
    "type": "Golf training area",
    "municipality": "Espoo",
    "province": "Uusimaa",
    "yearOfConstruction": null
  },
  {
    "_id": "5829b3cdf2fed4db45a52791",
    "id": 77735,
    "name": "Golfkeskus Oy Vuohenoja (9)",
    "type": "Golf training area",
    "municipality": "Tampere",
    "province": "Pirkanmaa",
    "yearOfConstruction": 1993
  },
  {
    "_id": "5829b3cdf2fed4db45a52792",
    "id": 508844,
    "name": "Jyvä-Golfin harjoituskenttä (3)",
    "type": "Golf training area",
    "municipality": "Jyväskylä",
    "province": "Keski-Suomi",
    "yearOfConstruction": null
  }
]
```

Figure 18.List of golf facilities in JSON format



Figure 19. API result showing only the names of the facilities

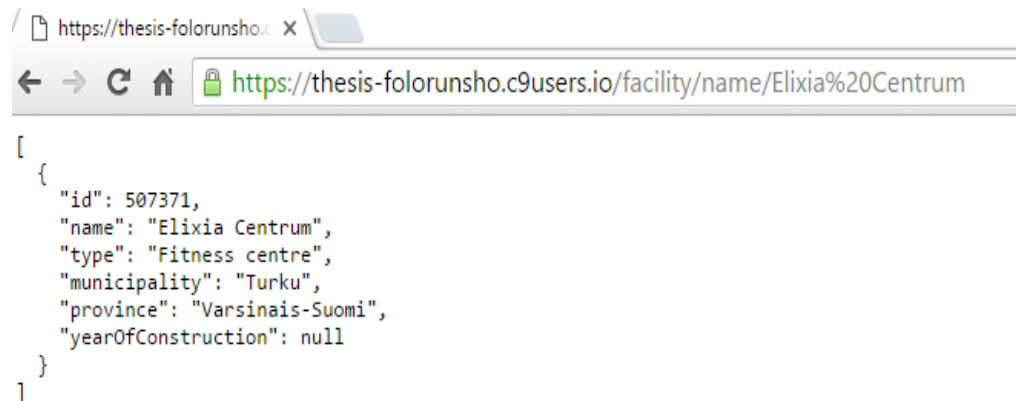


Figure 20. Getting accurate information of a facility by name

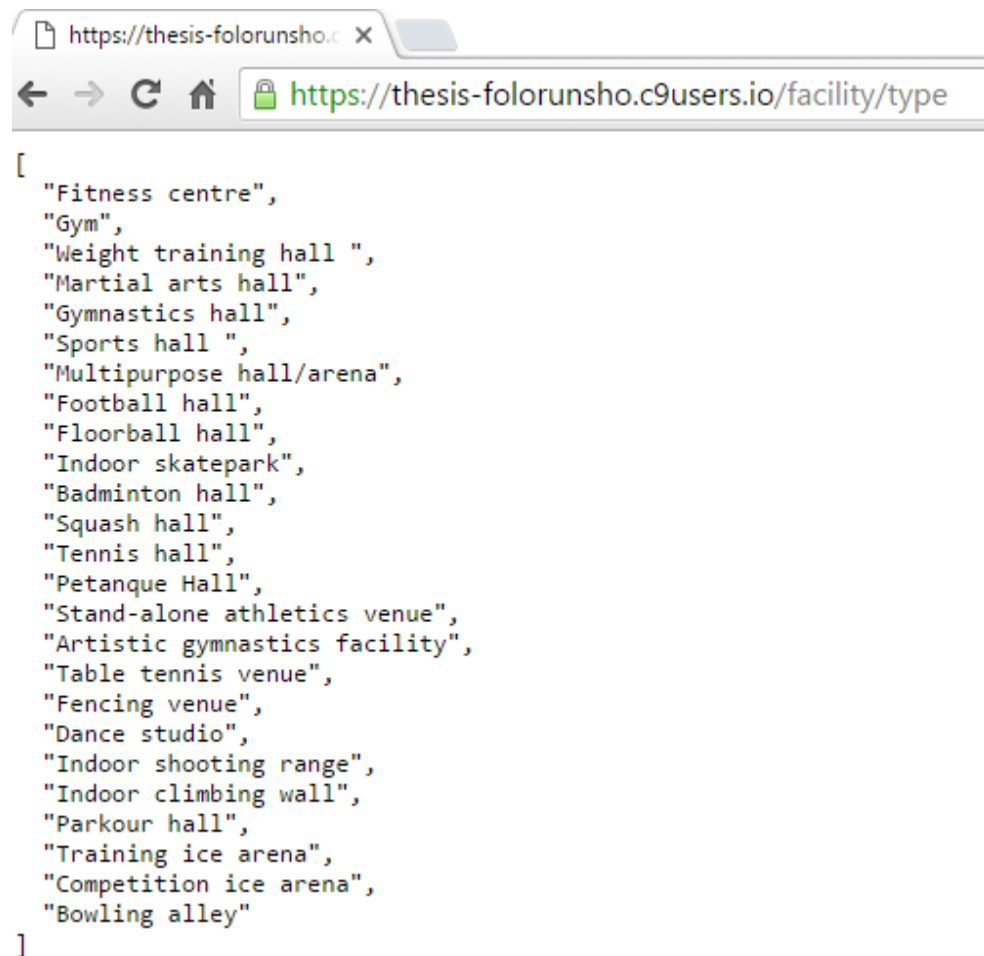


Figure 21.API result showing all the facility types



Figure 22.List of facilities with a gym facility

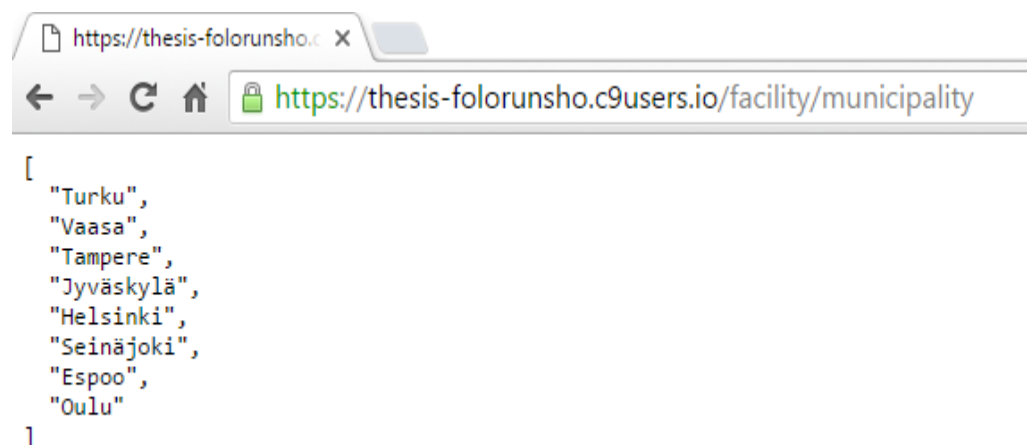


Figure 23.List of municipalities where facilities are located

4.2 Result from Graphical User Interface (Front-end) Program

The following figures (24-28) show results from the user interface.

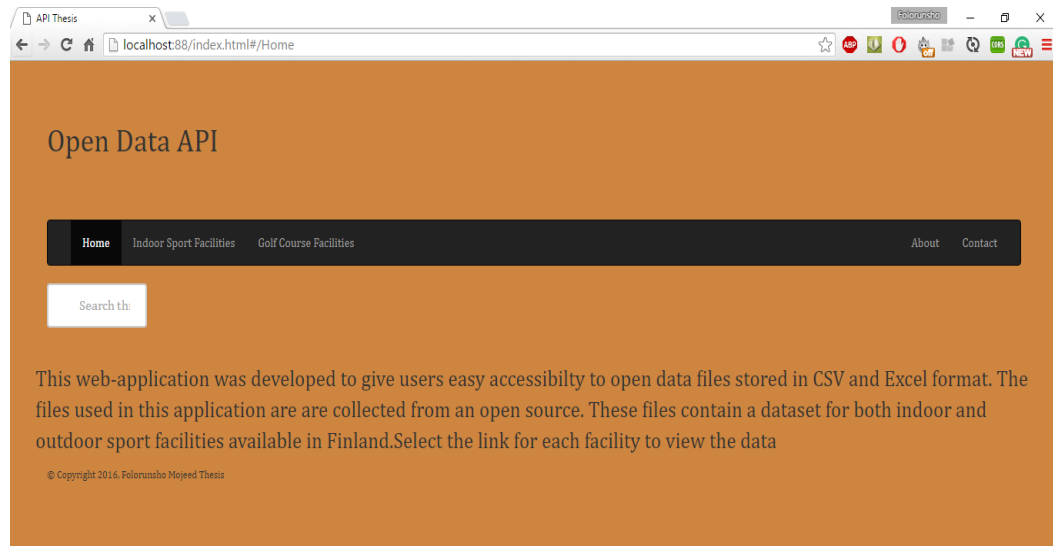


Figure 24.Home page of the appliccation

Search through the datasets..

Filter the Datasets By:

☒ Id ☒ Name ☒ Municipality ☒ Province ☒ Year

Facility Id	Name of Facility	Municipality	Province	Year of Construction
507371	Elixia Centrum	Turku	Varsinais-Suomi	
507372	Elixia Jokivarsi	Turku	Varsinais-Suomi	
507320	EasyFit Vaasa	Vaasa	Pohjanmaa	
522657	M&M Kintotalo Koulukatu	Turku	Varsinais-Suomi	
507322	LadyLine Vaasa	Vaasa	Pohjanmaa	

Figure 25.Indoor facilities presented on a table

[Home](#)
[Indoor Sport Facilities](#)
[Golf Course Facilities](#)

[About](#)
[Contact](#)

Filter the Datasets By:

☒ Id
☒ Name
☒ Municipality
☒ Province
☒ Year

Facility Id	Name of Facility	Municipality	Province	Year of Construction
507320	EasyFit Vaasa	Vaasa	Pohjanmaa	
507322	LadyLine Vaasa	Vaasa	Pohjanmaa	
507324	Kuntokeskus Wasamove	Vaasa	Pohjanmaa	
522575	Kuntokeskus Ladies Club & Spa	Vaasa	Pohjanmaa	

Figure 26.Using search box to filter dataset with user input

[Home](#)
[Indoor Sport Facilities](#)
[Golf Course Facilities](#)

[About](#)
[Contact](#)

Filter the Datasets By:

☒ Id
☒ Name
☒ Municipality
☒ Province
☒ Year

Golf centre Id	Name of Golf course Arena	Municipality	Province	Year of Construction
507523	Alberga Golf	Espoo	Uusimaa	
77735	Golfkeskus Oy Vuohenoja (9)	Tampere	Pirkanmaa	1993
508844	Jyväskylän harjoituskenttä (3)	Jyväskylä	Keski-Suomi	
508871	Laajasalon Golfjokamieskenttä (6)	Helsinki	Uusimaa	1959

Figure 27.Golf facilities presented on a table

[Home](#) [Indoor Sport Facilities](#) [Golf Course Facilities](#) [About](#) [Contact](#)

Search th:

Filter the Datasets By:

☐ Id ☒ Name ☒ Municipality ☒ Province ☐ Year

Name of Golf course Arena	Municipality	Province
Alberga Golf	Espoo	Uusimaa
Golfkeskus Oy Vuohenoja (9)	Tampere	Pirkanmaa
Jyväskylän Golfin harjoituskenttä (3)	Jyväskylä	Keski-Suomi
Laajasalon Golfjokamieskenttä (6)	Helsinki	Uusimaa

Figure 28.Filtering the dataset with checkbox options

5 CONCLUSION

5.1 Challenges

One of the major problems faced by developers in developing applications like this is how to effectively manage the datasets from the server side of the application to the user interface. Proper synchronization between the files and the server program can be difficult if the data are not properly structured in the open data files.

The development of this application with MEAN stack technology requires a very good understanding of programming in general and more specifically in JavaScript programming. This is due to the fact that MEAN stack is a programming style which is built on JavaScript framework. Before developing this application, I had to learn how to develop an application using MEAN technology. The learning process took some months, which eventually hindered my progress on this project.

The back-end and front-end of the application was planned to be developed on cloud9 IDE, which is an online development environment. However, due to some difficulties in downloading some of the MEAN dependencies, only the back-end was successfully developed on the cloud9 platform. Therefore, creating the front-end on another platform require proper importation of the necessary files from the back-end. Several difficulties were encountered during the importation of the files but were later solved.

5.2 Possible Improvements

The solution presented in this project has met the stated requirements and objectives of the thesis work. However, there are possibilities that developers and

users of the application would likely have their own open data files stored in CSV or Excel format that data of which they would like to access easily. Therefore, improvements can be made on the application such that users and developers can upload their files to the application, while the application program helps analyze and present the data accordingly on the graphical user interface.

Another possible improvement would be to build a mobile application (Android and IOS) that communicates with the sports facilities API. Perhaps this could be considered as a thesis topic for another student of this university.

References

- /1/ Open Knowledge International. 2009. Open Data Handbook. Accessed 11.10.2016. <http://opendatahandbook.org/guide/en/introduction/>
- /2/ Joel Gurin. 2014. Open Data Now. Accessed 11.10.2016. <http://www.opendatanow.com/wp-content/uploads/2013/12/Open-Data-Now-Advanced-Sample-Chapter.pdf>, p1-3.
- /3/ Tim Davies. Open Data Impacts – the research blog of @timdavies. Accessed 12.10.2016. <http://www.opendataimpacts.net/>
- /4/ Finnish Transport Agency. Open Data .Accessed 13.10.2016. <http://www.liikennevirasto.fi/web/en/open-data#.WNK77knS201>
- /5/ Helsinki Region Infoshare.About Open Data .Accessed 13.10.2016. <http://www.hri.fi/en/open-data/>
- /5/ Open Data and Interoperability Tools.Open Datasets .Accessed 20.10.2016. <https://www.avoindata.fi/data/en/dataset>