1. Project overview

Creating a project in which we can view a list of schools, search/filter these schools and add a new school.

1.1. Objective

For each new school, at least the following attributes should be captured:

● Name

● Address (street, suburb, postcode and state)

● Number of students registered

The list-view should display all schools and allow for a search on name and address. Any school I add, should be stored in some sort of persistence.

1.2. Business case

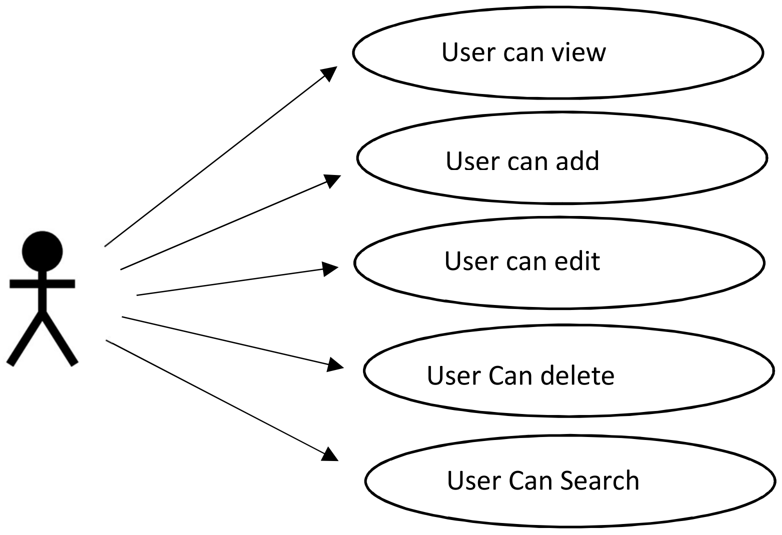
Access details about schools in a given area and store data related to them at a one central location. Also the system should provide opportunity to find and edit details very easily without any delay.

1.3. Out of scope

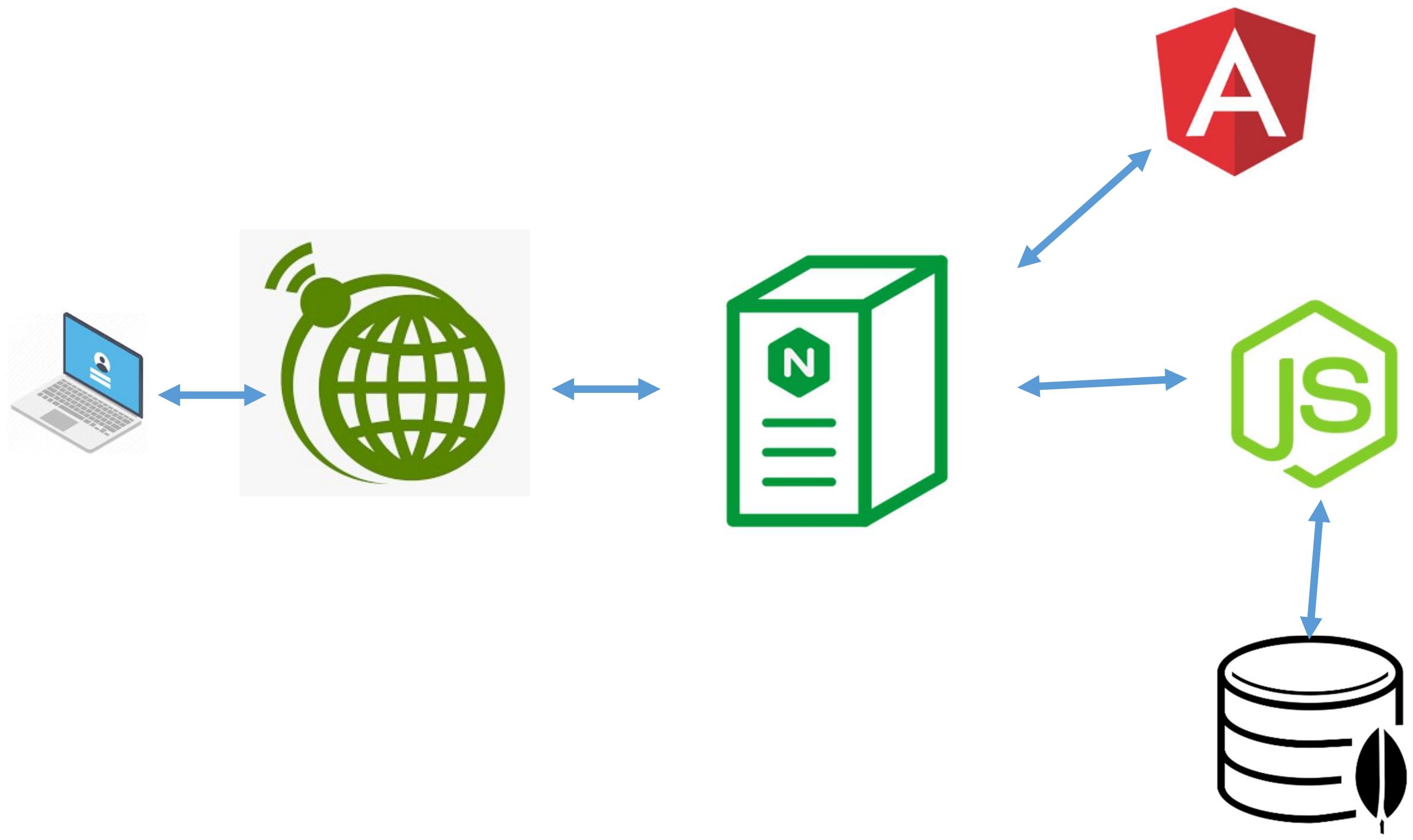
Authentication and authorization

2. Technical design diagram

2.1. Use case Diagram



2.2 Application infrastructure diagram



3. Technical specifications

Front end is implemented with the Angular 7 frame work. Angular Material and Bootstrap frameworks were used to build up the UI. Latest version of all the technologies being used and the application build as a single page mobile responsive app.

Backend API were developed with Nodejs with support of Expressjs Framework. Also few other nodjs modules being used such as mongoose, nodemon. And data persistence done with mogosDB as collections.

3.1.Web servers

Nginx is used as web server and front end is deployed in it. Also this server being used as revers proxy to make the necessary API calls to get data from backend.

4. Project setup

4.1.Font-end

Unzip the content in **Skoolbag_FrontEnd** in to local folder

Install nodejs latest version from following link https://nodejs.org/en/

Move to the project folder at command prompt and type npm install, press enter (This process will take little time to install all the necessary NPM packages.)

To run the project without deploying any sever use the following command **npm run start**

To build the front end in order to deploy in in a server run the following command **npm run build/ ng build**

4.2.Installing nginx (windows)

Follow the instruction in link given bellow

http://nginx.org/en/docs/windows.html

Nginx configurations and Deploying the front end Angular app

Go to nginx server folder and create new folder as skoolbag.

Move to conf folder and add new .conf file as skoolbag.conf

Add following code block in to that newly created .conf file and include that file into nginx.conf at bottom of http block.

```
server {

    listen       8081;

    server_name  localhost;

    #charset koi8-r;

    #access_log  logs/host.access.log  main;

              root skoolbag;

              index index.html;
location /api/ {
        proxy_pass      http://localhost:8080/api/;}}
```

Copy files inside dist folder of angular app and paste them inside newly created skoolbag folder inside nginx

Move to nginx folder in command prompt and start the nginx server using following command **start nginx**

4.3.Back-end

Unzip the **Skoolbag_BackEnd** in to a folder in your machine.

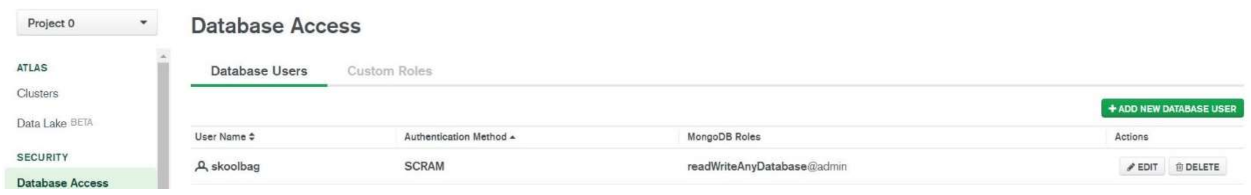
Move to the above created folder at command prompt and run **npm run Dev**

4.4.Mongo Db configurations

Create a new free account in mongo cloud at following link

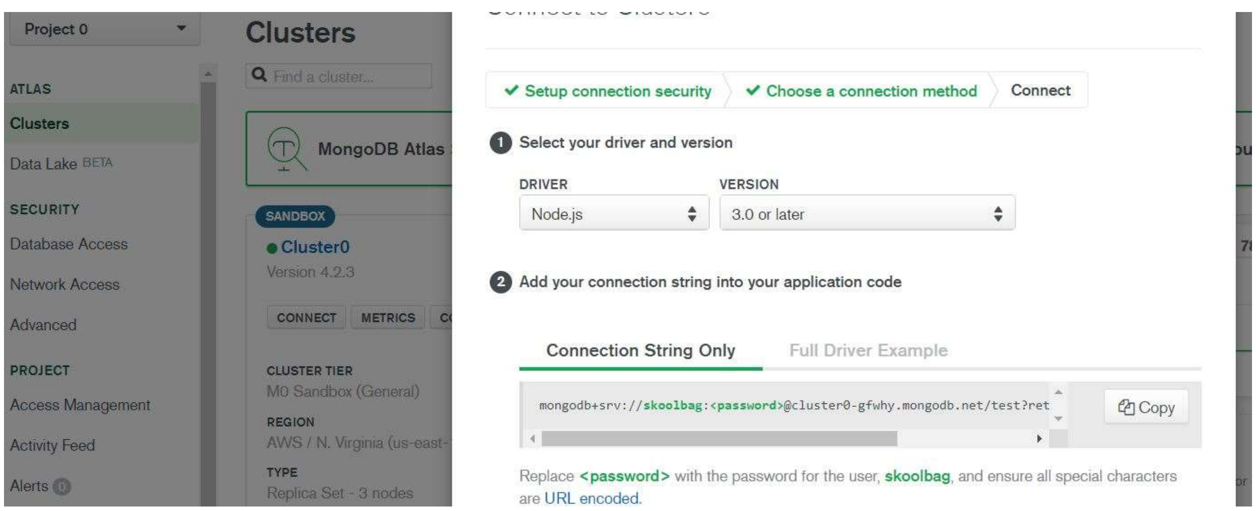
https://account.mongodb.com/account/login

Then move to Database access using left hand menu and add new user with relevant access permissions as in the image given bellow



User Name	Authentication Method	MongoDB Roles	Actions
skoolbag	SCRAM	readWriteAnyDatabase@admin	EDIT DELETE

Next navigate to clusters and get the db connection string and add the password in to the string which you used at the time when you create the user.



1 Select your driver and version

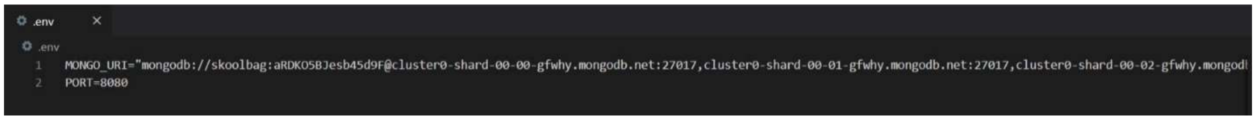
DRIVER: Node.js
VERSION: 3.0 or later

2 Add your connection string into your application code

```
mongodb+srv://skoolbag:<password>@cluster0-gfwhy.mongodb.net/test?ret
```

Replace <password> with the password for the user, skoolbag, and ensure all special characters are URL encoded.

Next update the connection string in .env file inside the backend API folder



```
MONGO_URI="mongodb://skoolbag:aRDKO5BJesb45d9F@cluster0-shard-00-00-gfwhy.mongodb.net:27017,cluster0-shard-00-01-gfwhy.mongodb.net:27017,cluster0-shard-00-02-gfwhy.mongod
PORT=8080
```

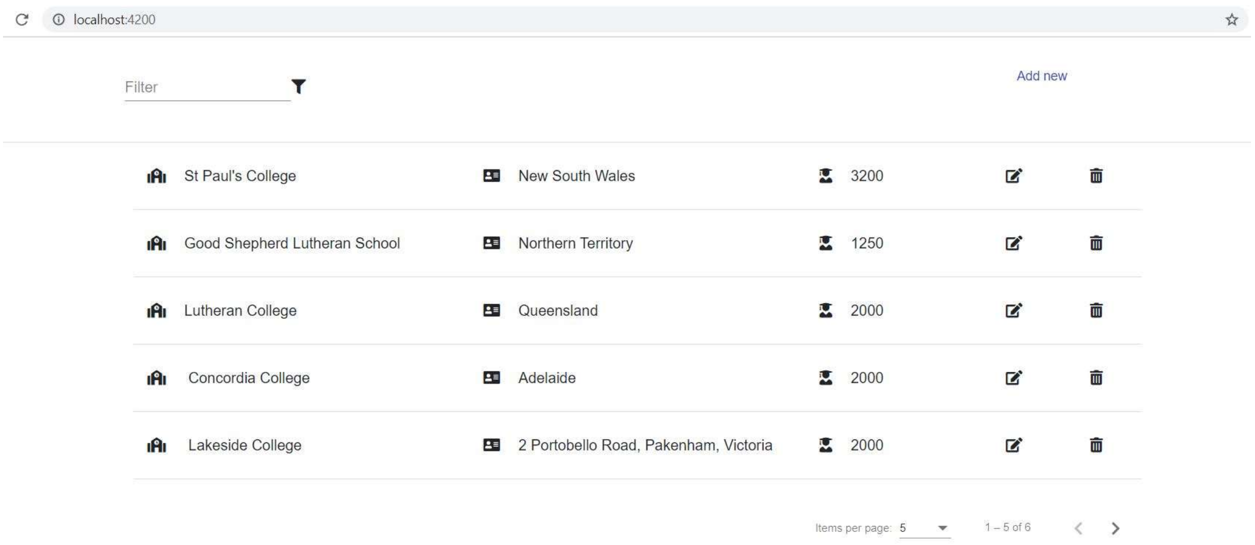
Note: You can download the community version from following link and set up the DB locally in your machine as well. https://www.mongodb.com/download-center/community

Finally move to the backend project folder using command promt and run the following command

npm run dev

(*Make sure you have done the Mogo configuration before you start the node serevr)

now open new browser window and type http://localhost:8081/ hit ente. If you have done all the configuration properly you will see similar screen like bellow.



St Paul's College	New South Wales	3200
Good Shepherd Lutheran School	Northern Territory	1250
Lutheran College	Queensland	2000
Concordia College	Adelaide	2000
Lakeside College	2 Portobello Road, Pakenham, Victoria	2000

5. API Description

/api/getSchools:

GET:

summary: find and return all the schools

description: return all the collection in db

produces:

- application/json

responses:

200:

```
        description: successful operation
        type: array


        schema:
                _id:String
                name:String
                address:String
                count:Number
```

/api/addSchools:

```
  GET:
   summary: find and return all the schools
   description: return all the collection in db
   parameters:
   - in: body
     name: body
     description: School object that needs to be added
     required: true
     produces:
        - application/json
        schema:
                name:{
                        type:String,
                        required:true,
                        maxlength:100,
                        minlength:5
                        },
                address:{
                        type:String,

                        required:"Address is required",

                        minlength:5,

                        maxlength:400

                        },

                count:{

                        type:Number,

                        required:true,

                        min:5,

                        max:10000

                        }

        responses:

        200:

                data: last added school object,

                message: "New school successfuly added"


        400:

        description: Invalid parameters
```