

Title and Cover Page

ADVANCED DATABASE SYSTEM (QH0541)

Solent University Manchester Branch

Student name: Ivan

Student number:10103365

Module title: Advanced Database System (QH0541)

Module leader: Dr Ajmal Gharib

Web application link: <http://localhost:3001/>

Git Repository: <https://github.com/Tervel10/Assessment->

Introduction

Creating of this project and the request for this module give us vision for what developers job is. Giving us a clue of structure and criteria for planning and create a full stack web application. Developing and design from zero it wasn't a easy task to do, creating a web pages, connected to each other with paths and routes with secured connection to database, installing scripts and dependencies gives a lot of questions and errors which had take time to understand and fix it. But satisfaction of achieving results and building a knowledge worth it.

System overview

As a full stack web application should be created front end related to the client, back end ralated to server software and both needs connected to data base

- As a data server for this project its used MongoDB with localhost:27017

[Documents](#) [Aggregations](#) [Schema](#) [Explain Plan](#) [Indexes](#) [Validation](#)

FILTER

{ field: 'value' }

ADD DATA

VIEW

```
_id: ObjectId('62a0d0134978dc6b357b8e56')
userName: "Ivan Radulski"
email: "T3nd000@gmail.com"
password: "$2b$10$.xWb0WQf0w8bNEjfcj601OgSjDVfKTjLEGc43Yn0BgKf/2OSIXOS6"
date: 2022-06-08T16:36:35.864+00:00
__v: 0
```

In database used for this project have only one added member

[Home](#) [Login](#) [New Member](#) [About Us](#)

Member Registration

Full Name

Nicolas

Email address

Nicolas@yahoo.com

Password

...

Submit

Registering a new member in our database from the website by pressing a Submit button.

[Home](#) [Login](#) [New Member](#) [About Us](#)

Member Registration

Full Name

sss

Email address

sss

Please include an '@' in the email address. 'sss' is missing an '@'.

...

Submit

Requesting for a real Email address

Documents	Aggregations	Schema	Explain Plan	Indexes	Validation
<div><div><div>FILTER</div><div>{ field: 'value' }</div></div></div>					
<div><div><div>ADD DATA</div><div>VIEW</div><div></div><div></div><div></div></div></div>					
<div><div><div><div><div>_id: ObjectId('62a0d0134978dc6b357b8e56')</div><div>userName: "Ivan Radulski"</div><div>email: "T3nd000@gmail.com"</div><div>password: "\$2b\$10\$.xWb0WQf0w8bNEjfcj6010gSjDVfKTjLEGc43Yn0BgKf/2OSIXOS6"</div><div>date: 2022-06-08T16:36:35.864+00:00</div><div>__v: 0</div></div></div></div></div>					
<div><div><div><div><div>_id: ObjectId('62a118efd5ff1c085ca99c4a')</div><div>userName: "Nicolas"</div><div>email: "Nicolas@yahoo.com"</div><div>password: "\$2b\$10\$dWYH1ZUE1xV8pbOe0DIc3.VWwmd3TcgswOXvLM.5Aox7exKnmhAwq"</div><div>date: 2022-06-08T21:47:27.657+00:00</div><div>__v: 0</div></div></div></div></div>					

As can see in our data is added a new object with the same details submitted from the website with bcrypt password for secure purpose

Key Desing Decisions

for this project has been used .ejs

NodeJS backend framework has been created on Express.js as a standart framework which provide features to create a web app its easy to use and provides various feautures which make a developer work easier. Its very flexible and light framework to use.

Express.js gives advantage as easily use and light framework to others frameworks as Koa, Hapi.js, Meteor.js and other.

For DB we are using json export file type, Mongoose module is used to connect the backend and database comminication, Mongoose is a object database modeling traslating codes from NodeJS to MongoDB data base

```
controllers > JS carController.js > ...
1  require ('../models/database')
2  const { name } = require('ejs');
3  const req = require('express/lib/request');
4  const { append } = require('express/lib/response');
5  const res = require('express/lib/response');
6  const carModel = require('../models/carModel');
7  const import bcrypt ('../models/User')
8  const bcrypt = require('bcrypt');
9  const morgan = require('morgan');
```

```
models > JS database.js > ...
1  // Database connection
2  const mongoose = require('mongoose');
3  mongoose.connect(process.env.DATABASE_URI,{
4    useUnifiedTopology:true,
5    useNewUrlParser:true
6  })
7  .then(() =>{console.log('database connected')})
8  .catch ((err)=>{
9    console.log(`database not connected ->${err.message}`)
10 })
11
12 require('../carModel')
13
```

Security and scalability

Conclusion and Reflection
