ADMS Week 5

- create a folder give it a name anywhere in your computer best is the desktop ex: adms
- open cmd,powershell,terminal and go to the folder ex:cd/desktop/adms
- run the following command

```
npm init -y
```

- drag and drop the folder you have created to the visual code or notepad++
- install the following packages using the smd/shell

```
npm install express mongoose mongodb dotenv
```

- open the package.json and look under dependencise
- create a file and name as index.js
- write the following code inside the index.js

```
const mongoose = require('mongoose')
const express = require('express');
const res = require('express/lib/response');
const app = express()
app.get('/', (req,res)=> res.send ('Hello World'));
var port=3000
app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

- save the file and run the following command

```
node index.js
```

- this will run the server on port 3000 go to the webbrowser localhost:3000 and you will see hello world written there
- with the above command you have to start and stop the server everytime you make a change to the index file or any file in the project we dont want to do that for that we will install a new package as development dependency the package name is ***nodemon***.
- install the package with the below command

```
    npm i -D nodemon
```

- add following in the script section of package.json

```
"start": "nodemon index"
```

setting up the database :

- create .env file in vscode same way you created an index file
- this is to save our critical information in a safe place (webserver or any other environment server)
- write the following in the .env file

```
DATABASE_URI=mongodb://localhost:27017/adms
PORT=3000
```

- if you are using local mongodb run the mongodb server
- write the following in the index.js file

```
require('dotenv').config()// at this to the top of the index
```

add the following code to the index.js

```
mongoose.connect(process.env.DATABASE_URI,{
  useUnifiedTopology:true,
  useNewUrlParser:true
})
.then(() =>{console.log('database connected')})
.catch ((err)=>{
  console.log(`database not connected ->${err.message}`)
})
```

your index file should look like this

```
require('dotenv').config()
const mongoose = require('mongoose')
const express = require('express');
const res = require('express/lib/response');
const app = express()
app.get('/', (req,res)=> res.send ('Welcome Home from index2'));
var port=3000

mongoose.connect(process.env.DATABASE_URI,{
  useUnifiedTopology:true,
```

```
  useNewUrlParser:true
})
.then(() =>{console.log('database connected')})
.catch ((err)=>{
  console.log(`database not connected ->${err.message}`)
})


app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

- start the server if it is not running npm start it will show database connected message on the console

create a schema for the collection

add the following to the index file after ***const app = express()***

```
const { Schema } = mongoose;
```

add the following code to the index file

```
//Schema
const studentSchema = new Schema({
  name:String,
  age:Number
 });
```

crate a model for student

```
//Modle
const Student=mongoose.model('student', studentSchema)
```

- Add the data to the student collection using mongo db compass we create a route for students and get the student data from the student collection using the following code

```
app.route('/students').get(function(req, res) {
  Student.find({},(err, result)=>{
    if (!err){
      res.send(result);
    }else{
      console.log(err)
      res.send('Some Error')
    }
```

```
  })
})
```

- go to the web browser and write the following in the address bar ***http://localhost:3000/students***
- you will see the result in json format