MongooseJS

**Install mongoose dependency in your app**

*. npm install mongoose –save*

**Requiring mongoose in model**

*var mongoose = require('mongoose');*

**Making Schema.**

*Var Schema = mongoose.Schema;*

**Defining Schema:**

*var bookSchema = new schema({*

*title: String, //of data type string*

*keywords: Array,*

*published: Boolean*

*});*

***Another method:***

var bookSchema = new Schema({

//there are two methods of datatypes in schema

//1.

title: string,

//2: passing object

title: {

type: string,

required: true/false, //it means that this field is required or not when saving the data

unique: true/flase, //whether the field should be unique or not

default: Date.now // for saving date

},

keywords: Array,

published: Boolean,

author: {

type: Schema.Type.ObjectId // this and the below is the same thing

type: Schema.ObjectId, //it is a kind of relation refering

ref: 'User' //to another model. so it will make relation

},

//embedded sub-document

// we will access this sub document as: data.detail.modelNumber

detail: {

modelNumber: Number,

hardcover: Boolean,

reviews: Number,

rank: Number

}

});

**Now exporting schema that it can be used where CRUD operation on data is performed.**

*module.exports = mongoose.model(‘Book’,bookSchema);*

CONNECTING TO MONGODB

*Var myDb = “mongodb://<dbuser>:<dbpassword>@ds117605.mlab.com:17605/irfistory”;*

*var db = "mongodb://root:irfan001@ds117605.mlab.com:17605/irfistory";*

*mongoose.connect(db,function(err){*

*if(err)*

*{*

*console.log(err);*

*}*

*else*

*{*

*console.log("db connected");*

*}*

*});*

USING MONGOOSEJS with MEAN STACK

**Npm install express mongoose body-parser --save**

*var express = require('express');*

*var app = express();*

*var bodyParser = require('body-parser'); //body parser will allow us to capture elements from the front end as well as from url*

*var mongoose= require('mongoose');*

*//create book.model.js and define book schema in it.*

*var Book = require('./Book.model');// book.model.js*

//routes

app.use(bodyParser.json()); //the format in which bodyParser will be used.

app.use(bodyParser.urlencoded({

extended: true

}));

app.get('/',function(req,res){

res.send("working");

});

//fetching all the records in the json format.

app.get('/books',function(req,res){

Book.find({})

.exec(function(err, books){

if(err)

{

res.send(err);

}

else

{

res.json(books);

}

});

});

//saving record to database

app.get('/save',function(req,res){

var b = new Book();

b.title = "another book";

b.author = "ifi";

b.category = "hardcover";

if(b.save())

{

res.send("saved");

}

});

//finding by id and passing id in url

app.get('/books/:id',function(req,res){

Book.findOne({

//getting parameter from url

\_id: req.params.id

}).exec(function(err,book){

if(err)

{

res.send(err);

}

else

{

res.json(book);

}

});

});

//middleware

app.get('/mid',function(req,res,next){

var x = 1;

if(x == 0)

{

//next();

}

else

{

console.log('not next');

}

});

//saving records through post method.

app.post('/create',function(req,res){

//one method for saving record can be the following

Book.create("arg1","arg2","arg3",function(err,book){

});

//// second method

b = new Book();

b.title = req.body.title;

b.author = req.body.author;

b.category = req.body.cat;

b.save(function(err,book){

if(err)

{

res.send(err);

}

else

{

res.send(book)

}

});

});

var port = 3000;

///connecting server

app.listen(port,function(){

console.log('app is listening on port '+port);

});

Book.model.js file

var mongoose = require('mongoose');

var Schema = mongoose.Schema;

var BookSchema = new Schema({

title: String,

author: String,

category: String

});

module.exports = mongoose.model('Book',BookSchema);

json node

const express = require('express');

const path = require('path');

const bodyParser = require('body-parser');

const app = express();

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({extended: false}));

// Set Static path

app.use(express.static(path.join(\_\_dirname, 'public')));

app.get('/users', (req, res) => {

let users = [

{

first\_name: "John",

last\_name: "Doe",

age: 34,

gender: "male"

},

{

first\_name: "Tom",

last\_name: "Jackson",

age: 23,

gender: "male"

},

{

first\_name: "Tracy",

last\_name: "Smith",

age: 38,

gender: "female"

}

];

res.json(users);

});

app.get('/download', (req, res) => {

res.download(path.join(\_\_dirname, '/downloads/pdf.pdf'));

});

app.get('/about', (req, res) => {

res.redirect('/about.html');

});

app.post('/subscribe', (req, res) => {

let name = req.body.name;

let email = req.body.email;

console.log(name+' has subscribed with '+email);

});

app.listen(3000, () => {

console.log('Server started on port 3000...');

});

PUG Template Engine

Main layout: layout.pug

doctype html

html

head

title= title

link(rel='stylesheet', href='/stylesheets/style.css')

link(rel='stylesheet', href='/stylesheets/bootstrap.css')

body

nav.navbar.navbar-inverse.navbar-fixed-top

.container

.navbar-header

button.navbar-toggle.collapsed(type='button', data-toggle='collapse', data-target='#navbar', aria-expanded='false', aria-controls='navbar')

span.sr-only Toggle navigation

span.icon-bar

span.icon-bar

span.icon-bar

a.navbar-brand(href='#') PC Repair

#navbar.collapse.navbar-collapse

ul.nav.navbar-nav

li(class=title=='Home' ? 'active' : undefined)

a(href='/') Home

li(class=title=='About' ? 'active' : undefined)

a(href='/about') About

li(class=title=='Services' ? 'active' : undefined)

a(href='/services') Services

li(class=title=='Contact' ? 'active' : undefined)

a(href='/contact') Contact

.container

block content

hr

footer

p PC Repair &copy; 2017

///inherited layout: index.pug

extends layout

block content

.jumbotron

.container

.row

.col-md-7

h1 Computer Not Working?

p

| Is your computer running slow? or maybe not at all? Call us for a quick and easy quote

p

a.btn.btn-primary.btn-lg(href='#') Read More

.col-md-5

img(src='images/mouse.png')

.row

.col-md-4

.well

h4 Hardware Problems?

p

| Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui.

.col-md-4

.well

h4 Software Problems?

p

| Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui.

.col-md-4

.well

h4 PC Tune Up

p

| Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui.

Reading JS FILE

var fs = require('fs');

var results;

fs.readFile('json/services.json', 'utf8', function(err, data){

if(err){

throw err;

} else {

results = JSON.parse(data);

}

});

router.get('/', function(req, res, next) {

res.render('services', {

title: 'Services',

services: results

});

});

Moule.exports = route;

Embedded js (.ejs)

/// file: header.ejs

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>TodoApp</title>

</head>

<body>

///Including header and footer in index.ejs file

<% include ../views/partials/header %>

HELLO FROM INDEX

<% include ../views/partials/footer %>

Fetching data from mongo db

app.get('/', (req, res, next) => {

Todos.find({}).toArray((err, todos) => {

if(err){

return console.log(err);

}

res.render('index',{

todos: todos

});

});

});

///////// printing the todos sent from the route.

<% include ../views/partials/header %>

<ul>

<% todos.forEach(function(todo){ %>

<li><%= todo.text %> - <%= todo.body %></li>

<% }) %>

</ul>

<% include ../views/partials/footer %>

MONGODB : ADD : DELETE :UPDAE

app.post('/todo/add', (req, res, next) => {

// Create todo

const todo = {

text: req.body.text,

body: req.body.body

}

// Insert todo

Todos.insert(todo, (err, result) => {

if(err){

return console.log(err);

}

console.log('Todo Added...');

res.redirect('/');

});

});

app.delete('/todo/delete/:id', (req, res, next) => {

const query = {\_id: ObjectID(req.params.id)}

Todos.deleteOne(query, (err, response) => {

if(err){

return console.log(err);

}

console.log('Todo Removed');

res.send(200);

});

});

// Update todo

Todos.updateOne(query, {$set:todo}, (err, result) => {

if(err){

return console.log(err);

}

console.log('Todo Updated...');

res.redirect('/');

});

});