JavaScript for exam

1.В папката където сме разархивирали скелета отваряме cmd/power shell.Пишем : npm install.Изчакваме докато се свалят пакетите.Стартираме start-mongod.bat за да се стартира MongoDB.Стартирам и RoboMongo.В cmd пиша node bin/www за да започне да слуша на порта или от WebStorm може да стартираме проекта.

2.Стартирам WebStorm.И отварям проекта за JS.В папката controllers ми е главният контролер.Там за да видя дали работи правилно,на метода index пиша: res.send("Hello") и стартирам наново приложението за да видя Hello на екрана(ако работи правилно).Ако проекта неиска да се стартира си отварям 1 Терминал и пиша : npm start

3.В WebStorm отивам в папка Models.Там ми е главната schema за проекта.В този файл трябва да добавя имената на колоните които ще ми се съдържат в таблицата(базата данни):

const mongoose = require('mongoose');  
  
let filmSchema = mongoose.Schema({  
 name: {type: 'string', required: 'true'},  
 genre: {type: 'string', required: 'true'},  
 director: {type: 'string', required: 'true'},  
 year: {type: 'number', required: 'true'},  
  
});  
  
let Film = mongoose.model('Film', filmSchema);  
  
module.exports = Film;

4.В главния контролер на метода Index добавям:

index: (req, res) => {  
 Film.find().then(films => {  
 res.render('film/index', {'films': films})  
 });

5.Отивам в папка config и файл config и виждам как ми се казва базата с данни:

connectionString: 'mongodb://localhost:27017/imdb'

6.Отивам в RoboMongo и създавам нова база данни със същото име като в config-a.

7.В папка models виждам как ще ми се казва колекцията,която създавам в RoboMongo

let Film = mongoose.model('Film', filmSchema);

Името трябва да е с малки букви и в множествено число.След това даваме insertDocument

И пишем някакъв пример : name : “alalba”,year : 1995. Даваме F5 и виждаме дали сме importnal-u нещо в базата.Винаги да поглеждам foreach-a в index,защото името може да е различно.

8.Метод за Create:отиваме в главния файл на контролера и пишем :

createGet: (req, res) => {  
 res.render('film/create');  
},  
createPost: (req, res) => {  
 let film = req.body;  
 Film.create(film).then(film => {  
 res.redirect("/");  
 }).catch(err => {  
 film.error = 'Cannot create film.';  
 res.render('film/create', film);  
 });

Като за данни пишем нашите данни-сменяме Film и film!Тестваме дали работи приложението.

9.Ако инпута е грешен,за да ни се запазват данните отиваме в view/create и добавяме на всяко поле value=“{{име на полето}}“

<input type="text" id="name" name="name" value="{{name}}">  
  
<label for="genre">Genre</label>  
<input type="text" id="genre" name="genre" value="{{genre}}">  
  
<label for="director">Director</label>  
<input type="text" id="director" name="director" value="{{director}}">  
  
<label for="year">Year</label>  
<input type="text" id="year" name="year" value="{{year}}">

10.Правим Delete метода и тестваме.

deleteGet: (req, res) => {  
 let filmId = req.params.id;  
 Film.findById(filmId).then(film => {  
 if (film) {  
 return res.render('film/delete', film);  
 }  
 else {  
 return res.redirect('/');  
 }  
 }).catch(err => res.redirect('/'));  
},  
deletePost: (req, res) => {  
 let filmId = req.params.id;  
 Film.findByIdAndRemove(filmId).then(film => {  
 res.redirect('/');  
 }).catch(err => res.redirect('/'));  
}

11.Правим Edit метода и тестваме.

editGet: (req, res) => {  
 let filmId = req.params.id;  
 Film.findById(filmId).then(film => {  
 if (film) {  
 res.render('film/edit', film );  
 }  
 else {  
 res.redirect('/');  
 }  
 }).catch(err => res.redirect('/'));  
},  
editPost: (req, res) => {  
 let filmId = req.params.id;  
 let film = req.body;  
  
 Film.findByIdAndUpdate(filmId, film, {runValidators: true}).then(films => {  
 res.redirect("/");  
 }).catch(err => {  
 film.id = filmId;  
 film.error = "Cannot edit film.";  
 return res.render("film/edit", film);  
 });

12.Когато сме готови с всичко,затваряме проекта(WebStorm) и базата данни.Стартираме start-mongodb,който се намора в папката на проекта.Стартираме Проекта още веднъж и пак тестваме.