Lappeenrannan teknillinen yliopisto

School of Business and Management

Software Development Skills

Muhammad Shahrukh Naveed, x080006

LEARNING DIARY CT70A9130\_15.04.2020 Software Development Skills: Back-End Anytime-course MODULE

**LEARNING DIARY**

15/July/2020

I started the course early, but due to the summer break, I started getting in the course a bit late. In the starting I didn’t know much about how to use Git. Hence I watched couple of videos on how to use git. From the basic commands of how to initialize, to how to set the origin and how to push and pull.

I created my own account.

17/July/2020

Got VS Code. Pretty easy to use Git with it. I personally think, it is a better editor than spider etc. Hence the rest of couple of hours were spend on getting the right editor. And VS code is the winner.

19/July/2020

I already knew about APIs. Hence I started watching the crash courses for Node.js from Travesy Media. The guy is pretty good, and help us understand the concepts in a easy way. I took the short course of Node.js, that basically explains everything needed to know how to start Node.js.

20/July/2020

Took course about Node.js express. Express is far easier and handy to use. Travesy Media explains it quiet well. Glad that we have this guy for free on Youtube. The rest of the couple of hours were send on watching and copying the code about Node.js express on my VS Code.

21/July/2020

Today I took the course about Mongo DB. The crash course by Travesy Media was all I needed to know to get started. Very informative course. Really learned about the unrelational database, since I have only used SQL. It will be very important to look at the GraphSQL too.

22/July/2020

Took a crash course on HTML 5 and CSS from some Youtub video. It was a good recap since I have worked with HTML 5 before. CSS and frontend stuff was new to me.

23/July/2020

Started this course from Travesy Media about a complete app about articles. And I think that will be for my application for this assignment.

24/July/2020

First part about easy. Initialise the Node. js

* Learned about Nodemon
* Learned about HTML a bit more
* How to connect Mongo DB in the application
* How to create different routes

Some Notes I took:

**INITIAL SETUP**

* Bring in Express
* app.get and app.listen
* Bring in pug
* Set view with app.set
* Bring in path
* App.set view engines
* Create view
* Res.render
* Create a layout template for the rest of the website
  + extends layout
  + block content
* In the index.pug get the article object with a loop, this will later be connected to the mongodb

25/July/2020

Some notes I took:

**Setting up MongoDB**

* Cmd to Mongobd portal
* Create a db with
  + use " "
  + db.createCollection(' article')
  + db.article.insert('')
* Install Mongoose
* Require it
* Get the connection key from online
* Create your mongoose **MODEL**
  + **It will be in a separate folder and file called articles.js**
  + \*This is very important to understand\*
  + This structures our database since it is no sql server.
  + Gives some infrastructure to our db
  + This lets us structure it on the application level
  + You first make the schema with
    - Let articleSchema = mongoose.Schema({……..})
    - Export the module
* Bring in your MODEL
  + Then use the article variable with
    - Require where the model is with articles.js
  + Article.find({}, (err,articles)=> {

res.render('index'.{…….}

}

26/July/2020

Started adding Articles. Some notes I took:

Learned about bootstraps a bit more

Added the form

**Add articles with form**

form(method='POST', action='/articles/add')

#form-group <=== bootstrap class

-In the app file app.post('/articles/add' ……….

let article =new Article() <=== This is from the model..

-Need to read it with body parser!!! So you need to bring it in with 2 lines of code from the internet, which is the required middleware

-Rest of the code is like:

article.title = req.body.title…..

……………………

article.body =article.body.body

article.save(function(err){

………………}

else{

res.redirect})

28 July 2020

Added the Nav bar and some customization to the front end, although not needed for the course.

**Do the bootstrap and css**

* We need a public static folder in which we can upload all our css files etc. pictures etc. Express needs to know this, so we will write this as:

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

* Then you dowload bootstrap in a specific folder.. Need to check what is the new way of downloading bootstraps
* Then link the folder in the outline file like this:

Machine generated alternative text:
Rom 
—elp 
app-Js 
doc type html 
html 
href= ' /bower_components/boostrap/di 
2017 
Articles 
• Article One 
Article Two 
• Article Three 
dede 
• 
• Article Four 
CopyTight 2017 
6 
e.css.map 
e. m ln. css.m 
11 
head 
title Knowledgebase 
link( rel= ' stylesheet ' 
body 
block content 
br 
hr 
footer 
p Copyright &copy; 

-Add a nav bar.. Chech it online

-in the index.pug add this do it can redirect to that article

Machine generated alternative text:
Rom 
'ages Help 
app-Js 
extends layout 
block content 
8 
hl #{title} 
ul. list-group 
each article, i in articles 
li. list-group-item 
/ article/ "+article. 
article. title 

-then in the app.js write this get route so that that it can route to that page when the link is clicked

-TIP: /:id is a place holder

Machine generated alternative text:
S? 
SS 
56 
// Get Single Article 
app. get(' /article/:id', function(req, res){ 
Article.findById(req.params. id, function(err, 
res . render( ' article ' , 
article : article 
article) { 

-Then just create the view in the view folder

30 July 2020

Got the edit button working and working. Some notes are:

**Edit button for the page**

**-**Add the edit button in the article.pug file that will route it to the edit link

Machine generated alternative text:
8 
article. pug 
extends layout 
block content 
hl= article. title 
h5 Written by #{article.author} 
p= article. body 
hr 
a . btn. btn-default (href= ' / article/edit/ ' +article ._id) 

**-Get this app.get so it can take you to the edit form**

Machine generated alternative text:
84 
85 
86 
87 
88 
89 
91 
14 
// Load Edit Form 
app. get( ' /article/edit/:id', function(req, res){ 
Article. findById(req . params. id, function(err, article){ 
res . render( ' edit _ article ' , 
title: 'Edit Article' , 
article : article 

3 August:

Added the rest of the stuff to edit button:

**-Make the form**

Machine generated alternative text:
article_pug 
extends layout 
block content 
hl #{title} 
form(method= ' POST ' , 
#form-group 
label Title: 
edit_article.pug 
app. js 
action= ' / articles/edit/ ' +article ._id) 
8 
11 
1) 
input. form-control (name= ' title ' 
, type= 'text' , 
#form-group 
label Author: 
input. form-control (name= ' author ' 
type= •text' , 
#form-group 
label Body: 
value-article. title) 
value—article . author) 
textarea. form-control (name= 'body ' )= article. body 
br 
input . btn . btn-primary(type= ' submit ' , value= ' Submit ' ) 

**-Then you need app.post, so it can make the changes needed for the application**

* **For this you need to just create an object, which in this case is the let article = {}**
* **And have to make a querry which is like a dict in python {\_id:req.params.id}**
* **Then you just have to use the model to update the model with the id with Article.update(query, article….)**

Machine generated alternative text:
95 
96 
97 
100 
101 
102 
103 
104 
105 
106 
107 
108 
109 
L 10 
[11 
1- 
// Update Submit POST Route 
app. post(' /articles/edit/ :id', function(req, res){ 
let article 
article. title 
= req. body . title; 
article. author = 
req . body. author; 
article. body 
= req. body . body; 
let query = {_id: req. params .id} 
Article. update(query, article, 
if (err) { 
console . log(err) ; 
return; 
} else { 
res. redirect( 
Start Server 
app. listen(3Ø00, function(){ 
console. log( 'Server started on port 3000. ' ) • 

5th August:

In this I got the more complicated things figured out like

-The delete button

-Messages, Validation and sessions

Hence now I think the application is at stage that it could be submitted to the course grader

Machine generated alternative text:
:icle.pug — — Atom 
View Selection Find Packages 
P@ect 
nod ekb 
models 
node_modules 
public 
v bower_components 
bootstrap 
core.js 
jquery.js 
jquery.min.js 
jquery.min.map 
jquery.slim,js 
jquery.slim,min.map 
external 
app.js 
extends layout 
block content 
hl= article. title 
article. pug 
6 
h5 Written by #{article.author} 
p= article. body 