

Nama : Steven Tjayadi

NIM : 535180085

Kelas : C

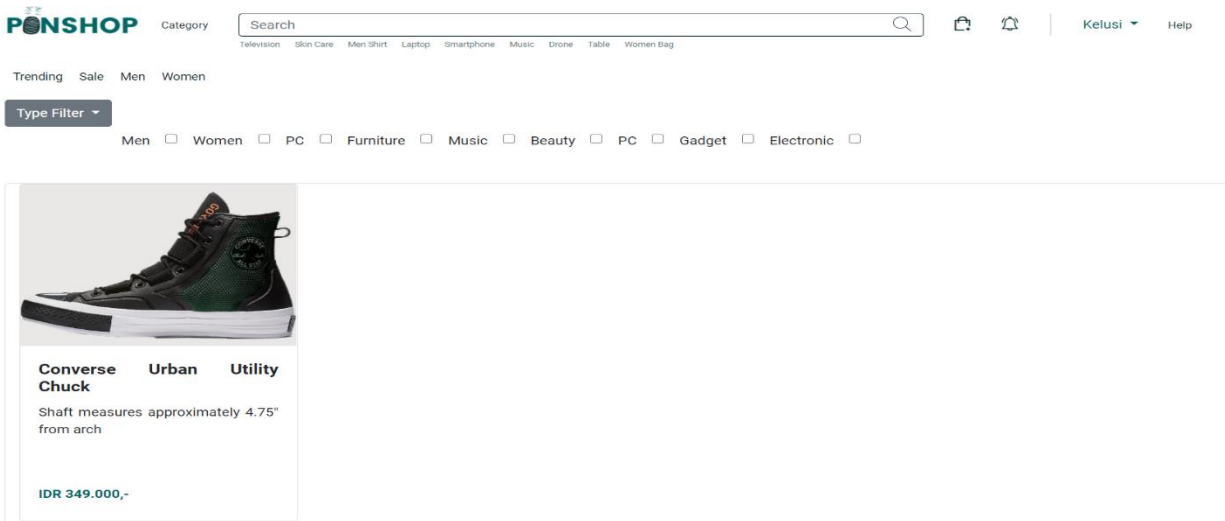
Mata Kuliah : Web Programming

Dokumentasi Progress Hasil Akhir Web Programming Project

1. Dalam pembuatan Website ini, dalam membuat aplikasi web termasuk dalam kelompok 8 di bagian online shop yang berjudul ponshop yang sudah dilakukan saat UTS dan UAS. Terdiri dari 2 bagian yang akan jelaskan yaitu :
 - Bagian FrontEnd
 - Bagian BackEnd

A. FrontEnd

Dalam suatu pembuatan FrontEnd atau juga merupakan bagian dari UTS Web Programming , bagian saya adalah bagian Searching. Maka, akan dijelaskan layout, gambaran serta source code fungsinya untuk menjelaskan cara kerja setiap fungsi dan output HTML, CSS , dan JS nya.



Gambar 1. Layout Searching

Dalam pembuatan layout searching ini dibuat berdasarkan kemiripan sedikit dari layout Home serta Category agar dapat disesuaikan dengan database dan format dengan jQuery nya untuk setiap layout HTML nya. Lalu dalam bagian Layout searching ini akan dijelaskan satu per satu per bagian yaitu :

- Filter
- Checkbox Category , seperti Men, Women , dll
- Hasil Layout Gambarnya
- Search box
- Investigate Searching

Pertama kali yang akan dijelaskan adalah bagian Struktur layout nya yaitu Search box dan investigate Searching.



Gambar 2. Layout Search Box dan Investigate Searching Button

Dalam pembuatan layout searching ini akan sangat penting pada saat akan melakukan pencarian suatu barang terutama dalam bagian homepage dan Category. Pembuatan source code HTML nya dilakukan seperti format di bawah ini :

```
<div id="search" class="p-2 bd-highlight" style="margin-top: 20px; padding-right: 100px;">
  <form action="/Search" method="post">
    <div class="input-group mb-3">
      <input type="text" name="search_user" class="form-control" placeholder="Search" aria-label="Search" ar:
      <div class="input-group-append">
        <button class="btn btn-outline-secondary" type="button" style="height: 32px; border-color: #0A292B;
        </div>
      </div>
    </div>
  </form>
```

Gambar 3. Source Code HTML Search Box dan Searching Button

Dalam pembuatan HTML ini disertai dengan div class yaitu untuk mengelompokkan bagian searching dan button nya. Lalu, untuk form ini akan dijelaskan lebih lanjut melalui bagian BackEnd. Placeholder akan digunakan untuk melakukan penulisan contoh format agar dapat dipahami user bahwa textbox tersebut dilakukan untuk searching barang.



Gambar 4. Layout Dropdown Filter

Penjelasan Design HTML selanjutnya adalah bagian Type Filter. Yaitu dimana setiap harga barang akan selalu dicategorikan sebagai Trending, Sales, dan Newest yang bergantung pada situasi user dan produksi itu sendiri. Untuk pembuatan design ini menggunakan menu DropDown. Untuk penjelasan setiap barang merupakan trending, Sales, maupun Newest akan

dijelaskan di bagian BackEnd . Berikut ini merupakan format source code HTML untuk bagian DropDown nya :

```
<div class="dropdown">
  <button class="btn btn-secondary dropdown-toggle" type="button" id="dropdownMenuButton" data-toggle="dropdown">
    Type Filter
  </button>
  <div class="dropdown-menu" aria-labelledby="dropdownMenuButton">
    <a class="dropdown-item" href="#">Trending</a>
    <a class="dropdown-item" href="#">Sales</a>
    <a class="dropdown-item" href="#">Newest</a>
  </div>
</div>
```

Gambar 5. Source Code HTML DropDown Filter

Dalam penjelasannya, yaitu menggunakan bootstrap dalam div class dropdown-menu dan bagian Button class = btn btn-secondary dropdown-toggle. Dalam dokumentasinya dapat dilihat pada link berikut ini :

<https://getbootstrap.com/docs/4.3/components/dropdowns/>

Kemudian Design selanjutnya adalah checkbox category. Checkbox Category merupakan cara untuk dapat mengklik tombol checkbox sesuai dalam category per barang searching masing-masing. Berikut di bawah ini merupakan hasil layout checkbox category nya.

Men ☐ Women ☐ PC ☐ Furniture ☐ Music ☐ Beauty ☐ PC ☐ Gadget ☐ Electronic ☐

Gambar 6. Layout Checkbox Category

Untuk Source Code HTML dapat dilihat seperti di bawah berikut :

```

<div class="container">
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">Men</button>
    <input type="checkbox" class="hidden"/>
  </span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">Women</button>
    <input type="checkbox" class="hidden"/>
  </span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">PC</button>
    <input type="checkbox" class="hidden"/>
  </span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">Furniture</button>
    <input type="checkbox" class="hidden"/>
  </span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">Music</button>
    <input type="checkbox" class="hidden"/>
  </span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">Beauty</button>
    <input type="checkbox" class="hidden"/>
  </span>
</div>

</span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">Beauty</button>
    <input type="checkbox" class="hidden"/>
  </span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">PC</button>
    <input type="checkbox" class="hidden"/>
  </span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">Gadget</button>
    <input type="checkbox" class="hidden"/>
  </span>
  <span class="button-checkbox">
    <button type="button" class="btn" data-color="default">Electronic</button>
    <input type="checkbox" class="hidden"/>
  </span>
</div>
</div>

```

Gambar 7. Hasil Source Code HTML Checkbox Category

Untuk pembuatannya dapat dilakukan dengan menggunakan span Button Checkbox (berfungsi sebagai tambahan div) biasa dengan tambahan input dan button type dalam checkbox tersebut. Data-color default adalah menggunakan warna checkbox yang formal, yaitu putih.

Lalu, kemudian terakhir dari penjelasan frontend akan dijelaskan bagian dalam searching yaitu card. Card ini merupakan salah satu bentuk yang dilakukan untuk mendesign layout setiap barang yang akan digunakan untuk diklik dan dibeli dalam website. Berikut ini merupakan gambaran layout untuk hasil card salah satu barang.



Gambar 8. Layout Card

Dalam pembuatannya adalah menggunakan JQuery dan memanggil JSON agar bisa dilakukan print HTML nya secara looping. Berikut ini merupakan tampilan Javascript dan koneksi menuju HTML nya :

```
$.getJSON('public/json/kumpulanDataHome/homeMen.json', function(data){
    let men = data.men;

    $.each(men, function(data){
        $('#homeMen').append('<div class="mid card h-100"> type="button" style="margin-top: 60px; margin-left: 90px; margin-right: 90px; ma
    });
});
```

Gambar 9. Format Javascript menggunakan JQuery

```

</div>
<div class="mid card h-100" style="margin-top: 40px; margin-left: 90px; margin-right: 90px; margin-bottom: 40px">
  <div id="homeMen">
  </div>
</div>
<script src="https://code.jquery.com/jquery-3.5.1.js" integrity="sha256-QWo7LDvxbWT2tbbQ97B53yJnYU3WhH/C8ycbRAKjPDc" ></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B+>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8" ></script>
<script src="js/home.js"></script>
<script src="js/home2.js"></script>
<script src="js/SearchFilter.js"></script>
</body>

```

Gambar 10. Koneksi id Javascript menuju HTML

Dalam penjelasan diatas ini, terlebih dahulu menggunakan `$(#homeMen)` untuk dapat bisa memanggil id ke HTML serta dengan source code :

```
<script src="js/SearchFilter.js"></script>
```

Demikian untuk penjelasan bagian FrontEnd tampilan Searching. Selanjutnya akan dijelaskan bagian berikutnya yaitu bagian untuk BackEnd.

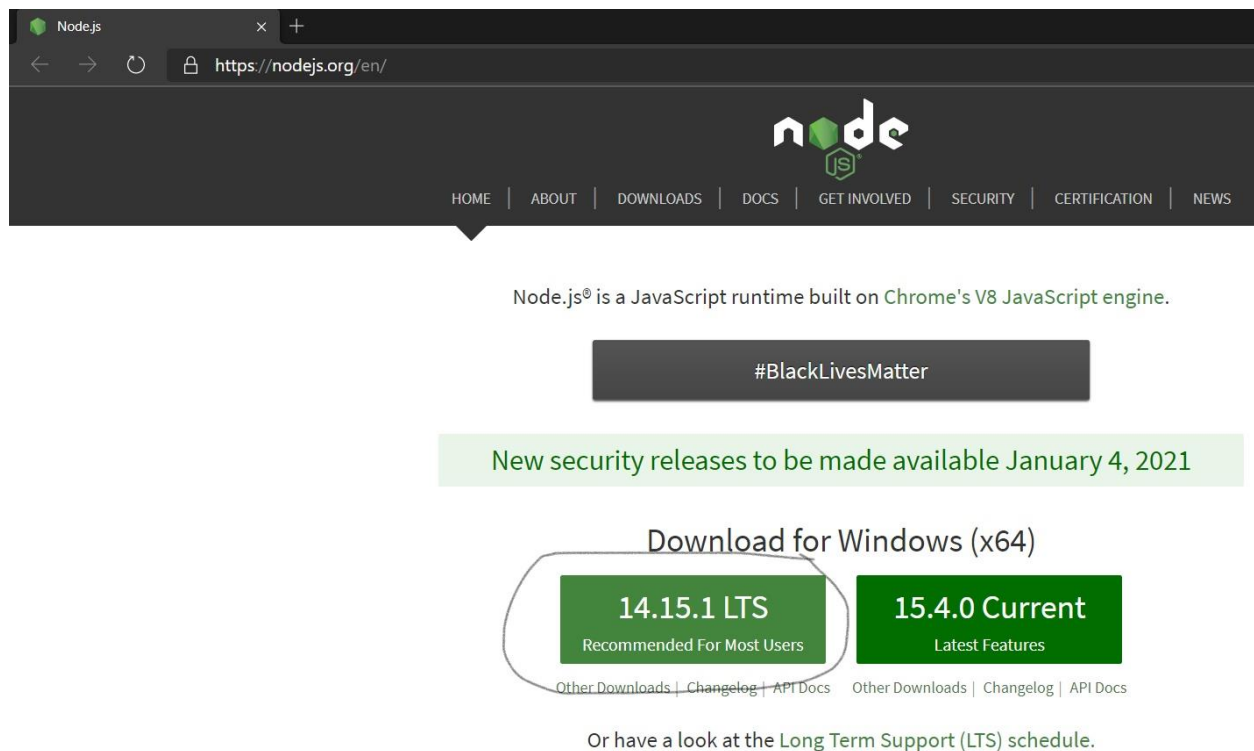
B. BackEnd

BackEnd merupakan bagian dalam server web yang akan melakukan service untuk bisa mengakses hasil frontend menuju ke public. Dalam penjelasan BackEnd ini akan lebih dijelaskan melalui routing, EJS, Mongoose, dll. Untuk penjelasan BackEnd di bawah ini akan lebih difokuskan pada bagian Searching. Dalam bagian BackEnd Searching ini , akan dibagi 3 cara untuk melakukan nya , yaitu :

1. Routing
2. EJS
3. MongoDB dan Mongoose

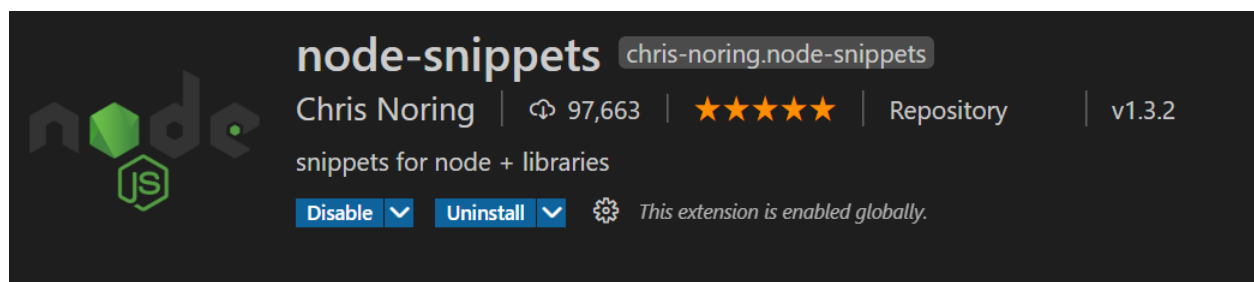
1. Routing

Dalam melakukan Routing di Searching, sebelum itu perlu melakukan instalasi npm terlebih dahulu terutama di dalam Node JS dan Express. Untuk Instalasi Node JS dapat melakukan Instalasi terlebih dahulu di dalam link : <https://nodejs.org>. Seperti format di bawah ini.



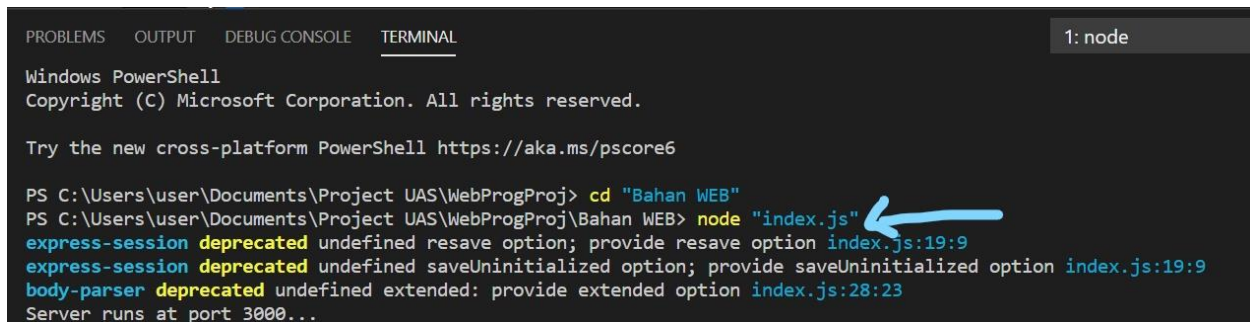
Gambar 11. Format Instalasi Node JS di website

Setelah melakukan Instalasi Node JS, kemudian melakukan instalasi Extension di dalam Visual Studio Code (VS Code). Yaitu berada di bagian node-snippet seperti gambar di bawah ini :



Gambar 12. Extension Node JS di Visual Studio Code

Kemudian setelah itu, dapat melakukan testing di terminal dengan menggunakan node seperti contoh di bawah :



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: node
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\user\Documents\Project UAS\WebProgProj> cd "Bahan WEB"
PS C:\Users\user\Documents\Project UAS\WebProgProj\Bahan WEB> node "index.js"
express-session deprecated undefined resave option; provide resave option index.js:19:9
express-session deprecated undefined saveUninitialized option; provide saveUninitialized option index.js:19:9
body-parser deprecated undefined extended: provide extended option index.js:28:23
Server runs at port 3000...
```

Gambar 13. Format output dengan menggunakan Node di Visual Studio Code

Selain itu, bisa juga melakukan node di cmd tergantung kebutuhan. Biasanya lebih sering menggunakan Visual Studio Code karena fitur aplikasinya sangat canggih dan mudah digunakan. Namun, jika melakukan project dari orang lain misalnya seperti fetch full hasil package json node js nya ini tidak akan bisa dilakukan sehingga perlu dilakukan instalasi terlebih dahulu sebelum dapat melanjutkan project.

Langkah selanjutnya yaitu melakukan instalasi menggunakan NPM atau disingkat sebagai node package manager. Ini akan melakukan instalasi NPM melalui cmd atau bisa saja menggunakan di terminal VS code itu sendiri. Yang akan diinstall berkaitan dengan project searching routing adalah instalasi Express. Instalasi Express akan dilakukan melalui format pertama :

npm init

NPM init berfungsi untuk membuat package baru yang akan dijalankan di File Express di js

npm install express

NPM Express akan digunakan untuk routing bagian searching

npm install nodemon

Nodemon akan digunakan untuk rerun program tanpa harus melakukan restart secara manual.

Namun, jika mengambil instalasi package dari sumber orang lain misalnya seperti git pull / fetch dari github desktop. Maka, dapat langsung melakukan instalasi menggunakan :

npm install

Maka akan muncul semua package yang ada di node-modules dan package-lock tambahan di file sendiri sehingga dapat melanjutkan project dari instalasi package sumber lain dengan sederhana tanpa harus instalasi satu per satu. Lalu kemudian dimasukkan import Express di dalam JavaScript sebagai berikut :

```
const express = require('express');  
const router = express.Router();
```

```
const app = express();
```

Lalu, setelah diinisialisasi variable app nya , Penjelasan Routing dalam bagian Searching ini dapat dilakukan melalui app.get dan app.post sesuai dengan format file yang berkaitan dengan searching itu sendiri. Gambaran untuk hasil express yang digunakan adalah seperti di bawah ini :

```

app.get('/Search' , (req,res) => {
  res.render('pages/Search' , {pageTitle : 'PONSHOP'});
});

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

  const search_name = req.body.Name;

  var searching = database_searching.find({Name : {$regex: search_name}})
  searching.exec((error, data) => {
    if (error) console.log("No Result for " + search_name);
    if (data) console.log("Got result in = " + JSON.stringify(data));
    res.redirect('/Search');
  });
});

```

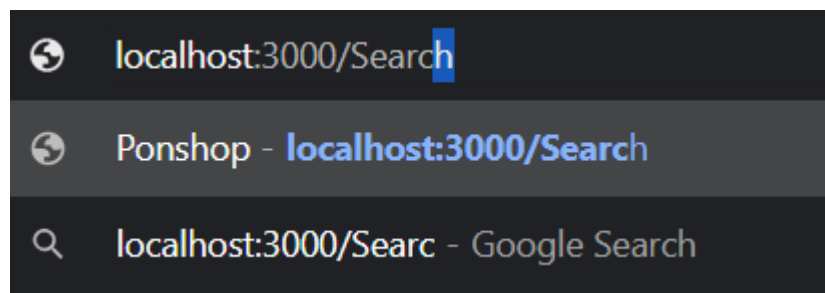
Gambar 14. Format Backend Searching di dalam Routing

Penjelasan :

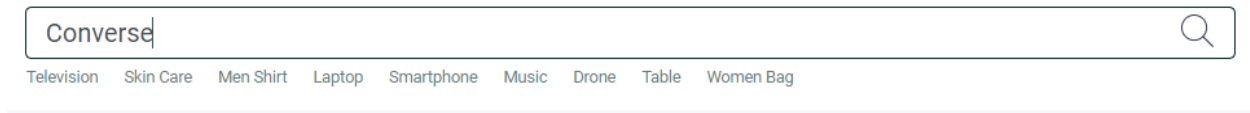
App.get : akan langsung masuk ke halaman search tanpa harus melakukan searching di front end

App.post : mesti masukkan terlebih dahulu condition (pencarian text di searching) kemudian melakukan redirect ke searching tersebut.

Format gambaran Perbedaan Searching menggunakan app.get dan app.post



Gambar 15. Format Searching menggunakan app.get



Gambar 16. Format Searching menggunakan app.post

Untuk Koneksi di dalam HTML nya adalah terletak di bagian gambar sesuai di bawah ini

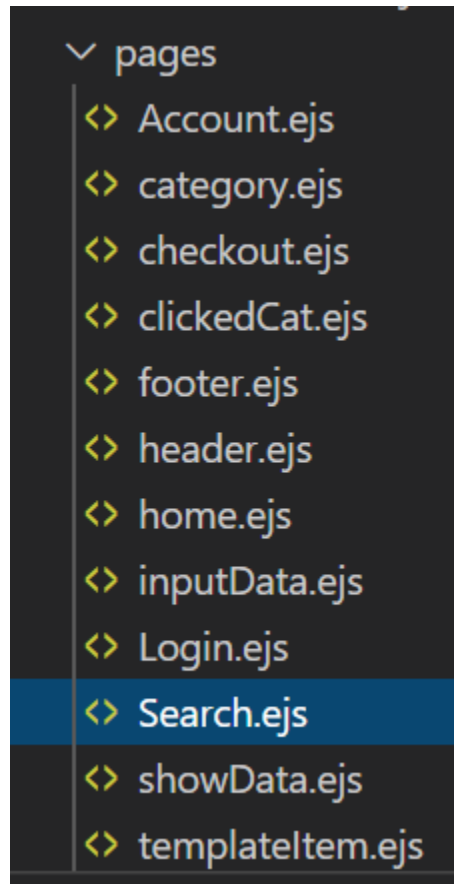
:

```
<div id="search" class="p-2 bd-highlight" style="margin-top: 20px; padding-right: 100px;">
  <form action="/Search" method="post">
    <div class="input-group">
      <input type="text" name="search_user" class="form-control" placeholder="Search" aria-label="Search" />
      <div class="input-group-append">
        <button class="btn btn-outline-secondary" type="button" style="height: 32px; border-color: #0A292B;">
        </div>
      </div>
    </div>
  </form>
```

Gambar 17. Letak Koneksi antara bagian post dari backend menuju frontend

2. EJS

Setelah melakukan proses Routing, yaitu menggunakan EJS untuk membuat template searching. Di bagian Searching saya. Hasil HTML nya sudah dijadikan dalam format EJS. Seperti di bawah berikut :



Gambar 18. Format Search dalam bentuk EJS diubah dari HTML

Walaupun tampilan EJS dan HTML tidak banyak berubah syntaxnya, namun itu akan berpengaruh dalam melakukan localhost routingnya.

Lalu, sebelum Masuk ke dalam format EJS, perlu menginstall NPM tambahan yaitu

```
npm install -save express nodemon body-parser ejs
```

```
npm install -save express-session
```

Untuk bagian body-parser ejs , dapat ditambahkan script dev nodemon sebagai berikut :

```
{
```

```
...  
"scripts": {  
  "dev": "nodemon",  
  ...  
}
```

Sehingga hasil dalam format package json nya akan tampil seperti di bawah ini :



```
    "devDependencies": {},  
    "scripts": {  
      "dev": "nodemon"  
    },  
    "repository": {  
      "type": "git",  
      "url": "WebProgProj"  
    },  
  },  
}
```

Gambar 19. Script format penulisan dev nodemon

Dalam penulisan format import EJS dapat dituliskan code berikut ini :

```
const bodyParser = require('body-parser');  
const session = require('express-session');
```

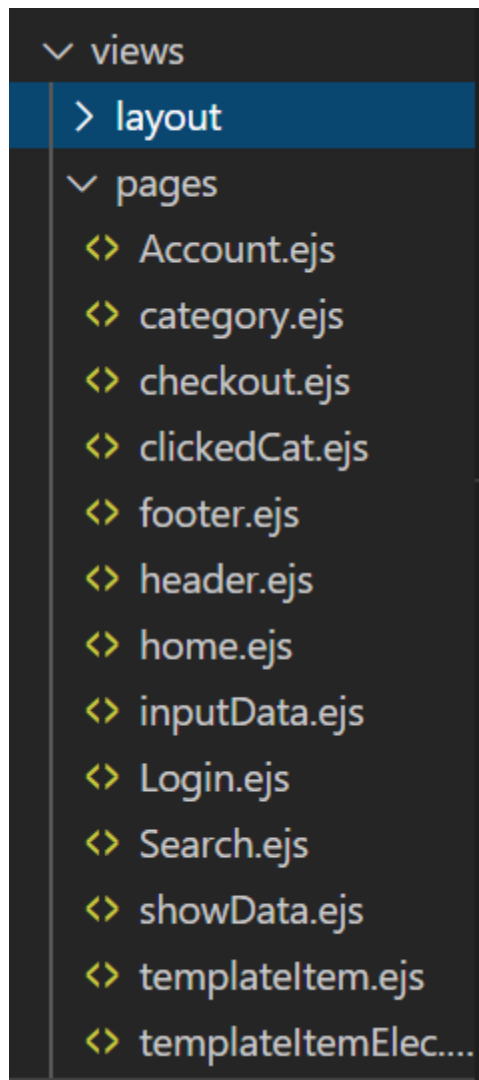
Kemudian dilakukan set engine EJS nya untuk dimasukkan format searching

```
// set the view engine to ejs

app.set('views', './views')
app.set('view engine', 'ejs');
app.use(session({
  secret: 'some_secret_key',
  cookie: {}
}));
```

Gambar 20. Hasil Source Code menampilkan hasil Views EJS menuju localhost

Sehingga dapat disimpulkan format untuk memasukkan views adalah sebagai format berikut ini :



Gambar 21. Hasil Lokasi Folder yang dipanggil dalam EJS

Kesimpulannya adalah format : `views/pages/Search`

Lalu, sebelum memasuki Langkah menuju MongoDB, dilakukan Source Code seperti di bawah ini untuk menyambungkan hasil JSON dari MongoDB menuju file localhost agar dapat dimasukkan ke dalam Express menuju EJS.


```
// static files

app.use("/public", express.static(__dirname + '/public'));
app.use(express.static('public'));
app.use(express.static("Database JSON"));
app.use('/kumpulanData', express.static(__dirname+ 'Database JSON/kumpulanData'));
app.use('/css', express.static(__dirname+ 'public/css'));
app.use('/js', express.static(__dirname+ 'public/js'));
app.use('/img', express.static(__dirname+ 'public/img'));
app.use('/json', express.static(__dirname+ 'public/json'));
app.use(bodyParser.urlencoded({ extended: false }))
app.use(bodyParser.json())
```

Gambar 22. Hasil File yang static

File yang digunakan untuk menggabungkan hasil MongoDB dan Express adalah :

- app.use(express.static('Database JSON'));
- app.use('/kumpulanData', express.static(__dirname+ 'Database JSON/kumpulanData'));
- app.use('/json', express.static(__dirname+ 'public/json'));

3. MongoDB atau Mongoose

MongoDB / Mongoose merupakan aplikasi database yang akan digunakan untuk menampung/ store data di bagian service-oriented dan cloud-service seperti di dalam dunia IoT (Internet of Things) untuk dunia kerja sekarang ini. Dalam Searching ini akan sangat berperan besar dalam melakukan aktivitas Searching tersebut di bawah ini. Sebelum masuk ke Mongoose dan jika belum menginstall , maka dapat dilakukan dengan menggunakan instalasi npm seperti di bawah ini :

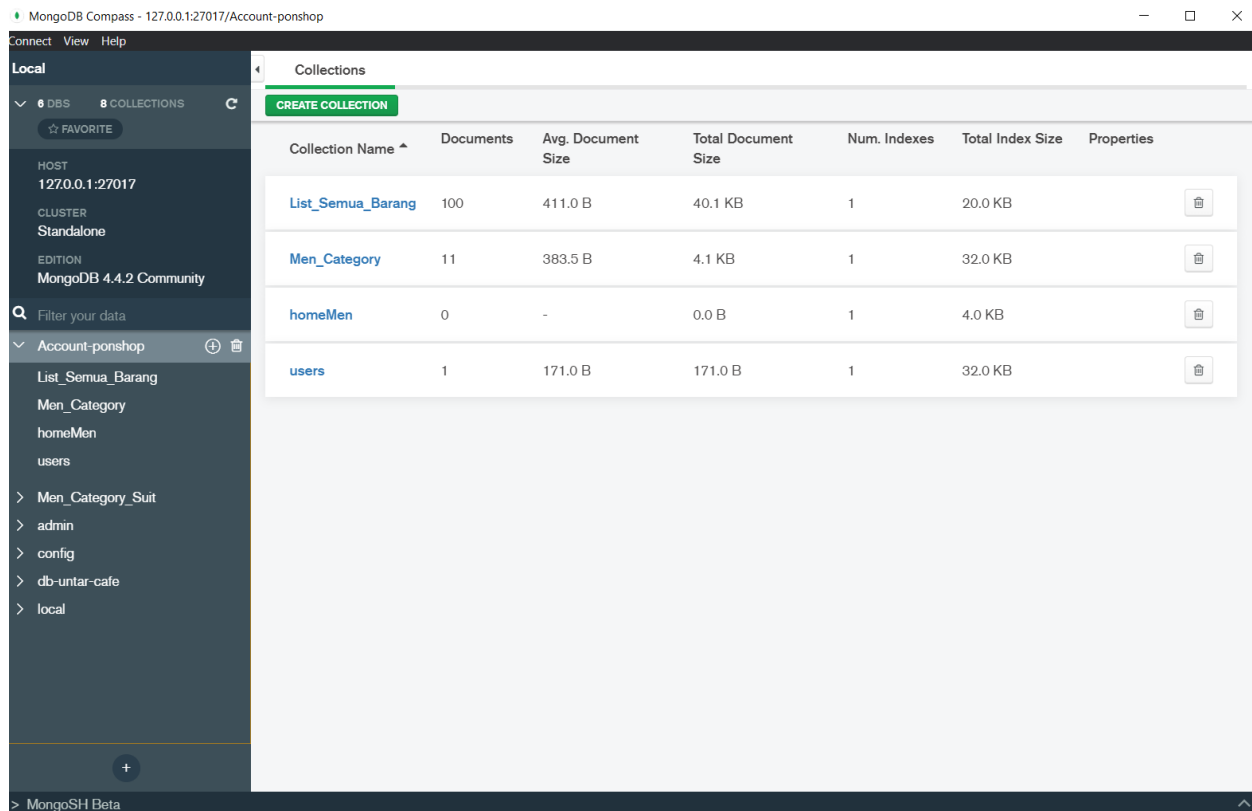
npm install mongoose

Jika di dalam fetch github , maka tinggal melakukan install npm install dan dijalankan dengan menggunakan npm run dev jika ingin melakukan testing npm nya. Berikutnya akan ditambahkan code di dalam script index.js yaitu sebagai berikut :

```
const mongoose = require("mongoose");
const user = require('./models/user');
const { static } = require('express');
const connectDB = require('./server/database/connection');
const database_searching = require('./server/model/item');
```

Gambar 23. Script Import MongoDB untuk memanggil Searching

Lalu, kemudian dilakukan Connection Mongoose nya melalui nama berikut ini :

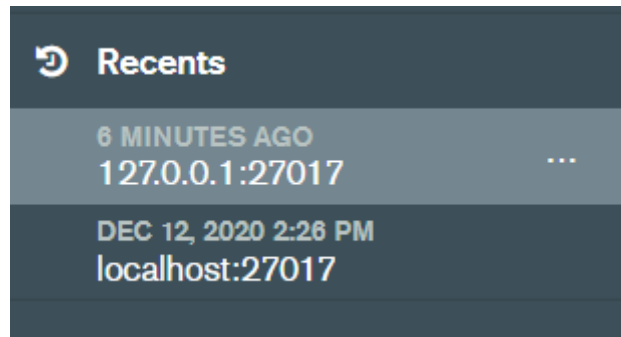


Gambar 24. Layout LocalHost MongoDB Compass

Untuk Nama connection dan hosting nya adalah :

Mongodb://127.0.0.1:27017/Account-ponshop

Atau keterangan lebih detail nya yaitu seperti gambar di bawah :



Gambar 25. LocalHost Connection

Sehingga jika digunakan dalam bentuk code Javascript , akan tampil seperti gambar di bawah :

```
const mongoose = require('mongoose');
const url = "mongodb://127.0.0.1:27017/Account-ponshop"
const connectDB = async()=>{
  try{
    const con = await mongoose.connect(url,
      {useNewUrlParser:true,
      useUnifiedTopology: true,
      useFindAndModify: false,
      useCreateIndex: true
    })
    console.log(`mongoose Connected: ${con.connection.host}`);
  }catch(err){
    console.log(err);
    process.exit(1);
  }
}
```

Gambar 26. Hasil Script Connection to MongoDB

Sehingga akan dikoneksi melalui index.js (main js nya) melalui fungsi code seperti di bawah berikut :

```
const mongoose = require("mongoose");
const user = require('./models/user');
const { static } = require('express');
const connectDB = require('./server/database/connection');
const database_searching = require('./server/models/item');
const app = express();
```

```
//MongoDB Connection
connectDB();
// body-parser to parse request body

router.use(bodyParser.urlencoded());
```

Keterangan : Butuh import './server/database/connection' dulu baru bisa dipanggil connectDB()

Selanjutnya adalah membuat Schema untuk bagian Database nya agar dapat diekspor dari mongoose menuju main js nya, yaitu :

```
const mongoose = require('mongoose');

var schema = new mongoose.Schema({
  Image : {
    type: String,
    required: true
  },
  Name : {
    type: String,
    required: true
  },
  Description : {
    type: String,
    required: true
  },
  Color : String,
  ListPrice : {
    type: Number,
    required: true
  },
  Category : {
    type: String,
    required: true
  },
  Filter : String,
  Discount : Number
});

const itemDB = mongoose.model('itemDB', schema);

module.exports = itemDB;
```

Gambar 27. Gambar Schema mongoose untuk dapat mengekspor dengan mudah

Sehingga pada saat dipanggil hasilnya adalah di bagian kode script berikut ini :

```
const mongoose = require("mongoose");
const user = require('./models/user');
const { static } = require('express');
const connectDB = require('./server/database/connection');
const database_searching = require('./server/model/item');
const app = express();
```

Gambar 28. Pada saat Schema ItemDB dipanggil

Lalu, dibuat controller dengan source code berikut ini :

```
var itemDB = require('./model/item');
const express = require("express")
const app = express();

exports.create = (req, res) => {
  if(!req.body){
    res.status(400).send({message:"Content can not be empty!"});
    return;
  }

  const item = new itemDB({
    Image: req.body.Image,
    Name: req.body.Name,
    Description: req.body.Description,
    Color: req.body.Color,
    ListPrice: req.body.ListPrice,
    Category: req.body.Category,
    Filter: req.body.Filter,
    Discount: req.body.Discount
  });

  //save to DB
  item
    .save(item)
    .then(data =>{
      // res.send(data)
      res.redirect('/input-data')
    })
}
```

Gambar 29. Controller untuk koneksi melalui EJS dari Schema

Kemudian, kita dapat melihat database list semua barang yang ada contohnya seperti gambaran di bawah ini :

```
_id: ObjectId("5fd6468ec1687f12445c38d1")
Image: "kategori/Men/Pants.jpg"
Name: "Adidas Men's Tiro 19 Training Pants"
Description: "100% Polyester, Imported, Drawstring closure, Unleash your inner strik..."
Color: "B"
List Price: 579000
Category: "Men"
Filter: "Normal"
Discount: 0
Price: 579000
Save: 0
```

Gambar 30. Hasil Gambaran Schema Database JSON dalam Mongoose

Dalam list tersebut, terdapat list data berikut :

- Image
- Name
- Description
- Color
- List Price
- Category
- Filter
- Discount
- Price
- Save

Dalam list diatas ini, yang kita gunakan dalam project untuk bagian Searching adalah Image, Name, Description, Category, and Price. Lalu data yang dapat digunakan biasanya adalah format JSON.

Untuk pembuatan Database banyak ini, Langkah saya adalah menggunakan excel terlebih dahulu untuk memasukkan semua barang. Kemudian ditransfer excel format dimasukan ke dalam csv untuk dibuat normalisasi data di dalam python karena terdapat library yang bisa

menormalisasi data dengan cepat yaitu dengan menggunakan library pandas (Javascript juga ada library lain yaitu namanya danfo-js , namun saya lebih menggunakan python karena fleksibel penggunaannya). Lalu setelah itu disorting huruf namanya sesuai alphabet dengan menggunakan method sort(). Akhirnya dimasukkan ke dalam format JSON dengan format JSON biasa. Sehingga akhirnya bisa dimasukkan ke dalam MongoDB Compass melalui import file.

Untuk penerapan caranya adalah sebagai berikut :

1.Import Library pandas dan JSON sertakan tampilkan melalui read_csv (Python)

```
import pandas as pd
import json

database_ponshop = pd.read_csv("semua data yang digabungkan ditambah gambar revisi untuk dipindah ke pandas.csv")
database_ponshop
```

	Image	Name	Description	Color	List Price	Category	Filter	Discount	Price	Save
0	kategori/Electronic/TV.jpg	TCL 32" 3 SMART TV	Smart functionality delivers all your favorite...	B	1850000	Electronic	Normal	0.0	1850000	0
1	kategori/Electronic/HTJ.jpg	BESTISAN Home Theater System Wired and Wireles...	100 watt 40 Inch Deep bass sound bar trasfer 1...	B	1490000	Electronic	Normal	0.0	1490000	0
2	kategori/Electronic/AWM.jpg	COMFEE' 1.6 Cuft Portable Washing Machine	This portable washing machine has 6 most commo...	B	3790000	Electronic	Normal	0.0	3790000	0
3	kategori/Gadget/SM.jpg	Samsung Galaxy Watch 3	Galaxy Watch3 combines style--two sizes, two fi...	B	4499000	Gadget	Normal	0.0	4499000	0
4	kategori/Gadget/SP.jpg	Apple iPhone 8 Plus	Apple iPhone 8 Plus 64GB Rose Gold Fully Unloc...	B	6699000	Gadget	Normal	0.0	6699000	0
...
95	HomePage/totebag/men,dll/totebag/Left Hard Pur...	Left Hard Purple	REUSABLE COTTON CANVAS BAGS. Canvas tote bags ...	None	40000	Totebags	Newest	0.0	40000	0
96	HomePage/Trending/beanie.png	White Beanie	Premium Material, Classic Unisex Solid Color W...	None	69000	White	Trending	0.0	69000	0
97	HomePage/Trending/bag.png	Duffle Bag	Large rectangular-shaped duffel bag with roomy...	None	239000	Bag	Trending	0.0	239000	0
98	HomePage/Trending/blanket.png	Blanket	Soft blanket utilizes 100% microfiber fabric a...	None	99000	Pillow	Trending	0.0	99000	0
99	HomePage/Trending/pillow.png	Square Pillow	The pillow cover is made of 100% cotton fabric...	None	39000	Pillow	Trending	0.0	39000	0

100 rows x 10 columns

Gambar 31. Cara menimport dataset dengan Python

2. Lakukan dengan method sort_values() sesuai dengan “Name” nya

```
sortedponshop = database_ponshop.sort_values(by = ["Name"])
sortedponshop
```

	Image	Name	Description	Color	List Price	Category	Filter	Discount	Price	Save
57	HomePage/sale/Electronic/kamera.png	4K Camera Digital	4K digital camera, with 4K/30 fps smooth video...	B	1650000	Electronic	Sale	26.0	1215000	435000
63	HomePage/sale/Electronic/tv.png	55-Inch Class Crystal UHD	Crystal processor 4K- This ultra-fast process...	B	8200000	Electronic	Sale	8.0	7500000	700000
19	kategori/PC/Monitor.jpg	AORUS F127Q 27" 165Hz 1440P FreeSync IPS Gamin...	AORUS F127Q 27" 165Hz 1440P FreeSync IPS Gamin...	B	6999000	PC	Normal	0.0	6999000	0
18	kategori/PC/Laptop.jpg	Acer Predator Helios 300 Gaming Laptop	Acer Predator Helios 300 Gaming Laptop, Intel ...	B	15999000	PC	Normal	0.0	15999000	0
13	kategori/Men/Pants.jpg	Adidas Men's Tiro 19 Training Pants	100% Polyester, Imported, Drawstring closure, ...	B	579000	Men	Normal	0.0	579000	0
...
58	HomePage/sale/Electronic/earbuds.png	Wireless Earbuds	Installed with Qualcomm Apt-X Tech, provides p...	B	670000	Electronic	Sale	13.0	580000	90000
79	HomePage/sale/Furniture/7.png	Wood Dining Table	Multiple functions - can be used as coffee tab...	None	3290000	Furniture	Sale	19.0	2672190	617810
6	kategori/Music/Guitar.jpg	Yamaha GL1 Guitalele	A unique mini 6-string nylon guitar that is sl...	B	1350000	Music	Normal	0.0	1350000	0
65	HomePage/sale/Fashion/beanie.png	Yellow Beanie	Soft, comfortable, friendly to skin, will give...	B	69000	Fashion	Sale	20.0	55200	13800
45	HomePage/totebag/men,dll/hoodie/Yellow Hoodie.png	Yellow Hoodie	Premium Material, Classic Unisex Solid Color W...	None	150000	Hoodie	Trending	0.0	150000	0

100 rows x 10 columns




Gambar 32. Melakukan sorting agar nama databasenya mudah dibaca

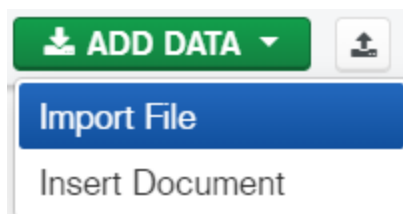
3. Kemudian dilakukan transfer ke JSON untuk dimasukkan ke Mongoose. Untuk JSON yang biasa digunakan termasuk bagian Searching ini adalah orient = "records"

```
convert_tojson = sortedponshop.to_json(orient = "records")
parsed = json.loads(convert_tojson)
json.dumps(parsed, indent = 4)
with open('D:/Database dipake untuk buat JSON MongoDB/Semua_Category.json', 'w') as f:
    json.dump(parsed, f)
```

Gambar 33. Cara membuat dataset menjadi format JSON

4. Pada saat datanya muncul secara otomatis , maka bisa mengimport data dengan format JSON di dalam MongoDB Compass

 Sale_Category	11/26/2020 9:59 AM	JSON File	9 KB
 Semua_Category	12/15/2020 7:10 AM	JSON File	40 KB
 Tech news electronic Category	11/22/2020 7:03 PM	Microsoft Excel Co...	4 KB



Untuk dapat melakukan Searching bisa menggunakan format coding sesuai di bawah ini :

```
const search_name = req.body.Name;

var searching = database_searching.find({Name : {$regex: search_name}})
searching.exec((error, data) => {
  if (error) console.log("No Result for " + search_name);
  if (data) console.log("Got result in = " + JSON.stringify(data));
  res.redirect('/Search');
});
```

Gambar 34. Source Code untuk lakukan Searching

Keterangan : Perintah Regex adalah text yang akan melakukan titik huruf pada saat searching akan muncul partial text bukan berbentuk full text.

Kesimpulan : Dalam Dokumentasi ini, masih bergantung pada bagian Searching saya sendiri dan masih ada ketergantungan antara bagian homepage dan Category sehingga dalam bagian Searching ini semoga bisa dikembangkan lebih baik lagi.