Ahmad Al Saadi CIS 4339 05/03/2021

Code Snippets

Client

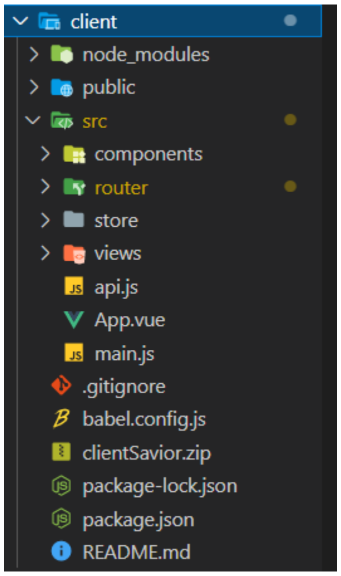

```
client
  node_modules
  public
  src
    components
    router
    store
    views
    api.js
    App.vue
    main.js
  .gitignore
  babel.config.js
  clientSavior.zip
  package-lock.json
  package.json
  README.md
```

Figure 1)Directory Hierarchy

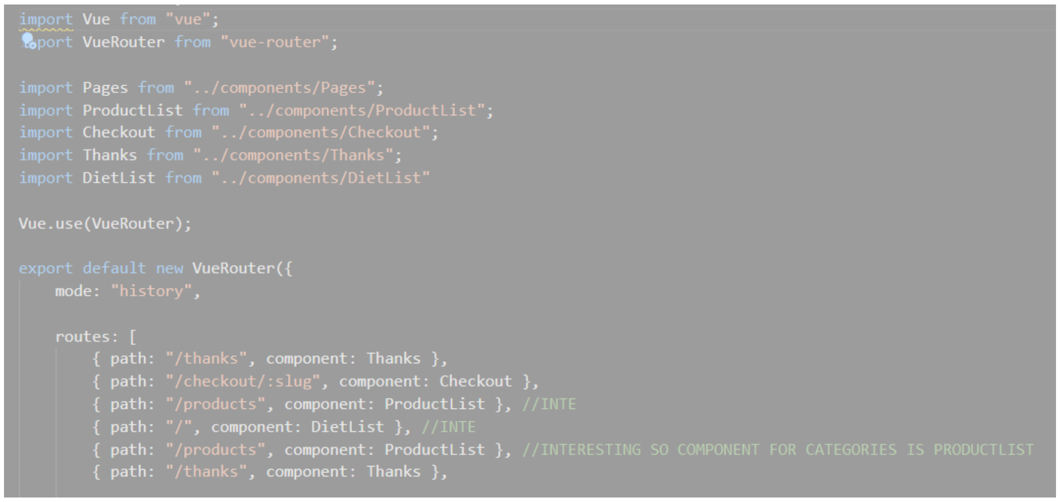
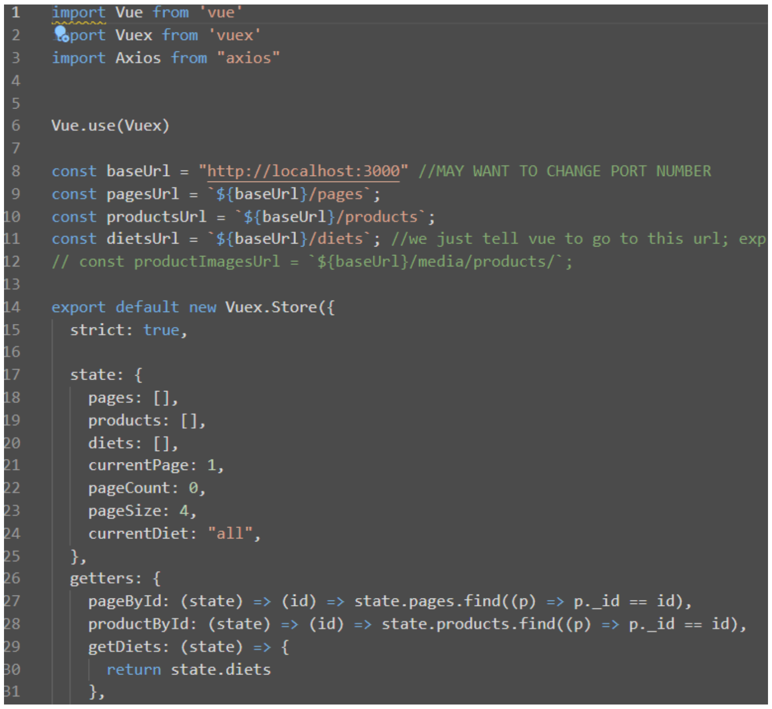
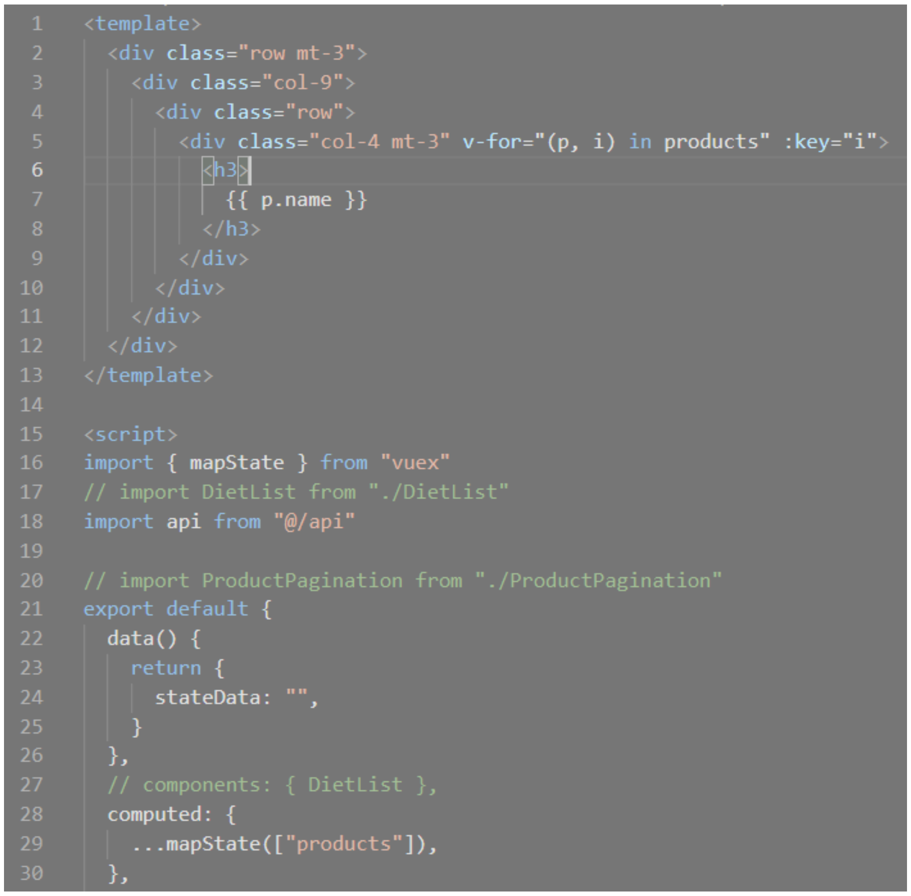

```
import Vue from "vue";
import VueRouter from "vue-router";

import Pages from "../components/Pages";
import ProductList from "../components/ProductList";
import Checkout from "../components/Checkout";
import Thanks from "../components/Thanks";
import DietList from "../components/DietList"

Vue.use(VueRouter);

export default new VueRouter({
    mode: "history",

    routes: [
        { path: "/thanks", component: Thanks },
        { path: "/checkout/:slug", component: Checkout },
        { path: "/products", component: ProductList }, //INTE
        { path: "/", component: DietList }, //INTE
        { path: "/products", component: ProductList }, //INTERESTING SO COMPONENT FOR CATEGORIES IS PRODUCTLIST
        { path: "/thanks", component: Thanks },
```

Figure 2)Routing code



```
import Vue from 'vue'
import Vuex from 'vuex'
import Axios from "axios"


Vue.use(Vuex)

const baseUrl = "http://localhost:3000" //MAY WANT TO CHANGE PORT NUMBER
const pagesUrl = `${baseUrl}/pages`;
const productsUrl = `${baseUrl}/products`;
const dietsUrl = `${baseUrl}/diets`; //we just tell vue to go to this url; exp
// const productImagesUrl = `${baseUrl}/media/products/`;

export default new Vuex.Store({
  strict: true,

  state: {
    pages: [],
    products: [],
    diets: [],
    currentPage: 1,
    pageCount: 0,
    pageSize: 4,
    currentDiet: "all",
  },
  getters: {
    pageById: (state) => (id) => state.pages.find((p) => p._id == id),
    productById: (state) => (id) => state.products.find((p) => p._id == id),
    getDiets: (state) => {
      return state.diets
    },
```

Figure 3)Store Code



```
<template>
  <div class="row mt-3">
    <div class="col-9">
      <div class="row">
        <div class="col-4 mt-3" v-for="(p, i) in products" :key="i">
          <h3>
            {{ p.name }}
          </h3>
        </div>
      </div>
    </div>
  </div>
</template>

<script>
import { mapState } from "vuex"
// import DietList from "./DietList"
import api from "@/api"

// import ProductPagination from "./ProductPagination"
export default {
  data() {
    return {
      stateData: "",
    }
  },
  // components: { DietList },
  computed: {
    ...mapState(["products"]),
  },
```

Figure 4)Sample Component Code

Server

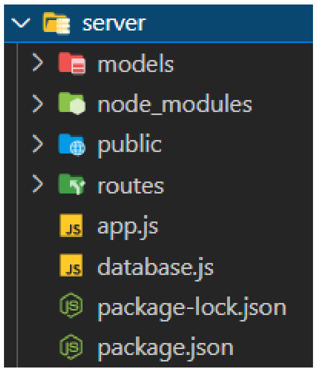
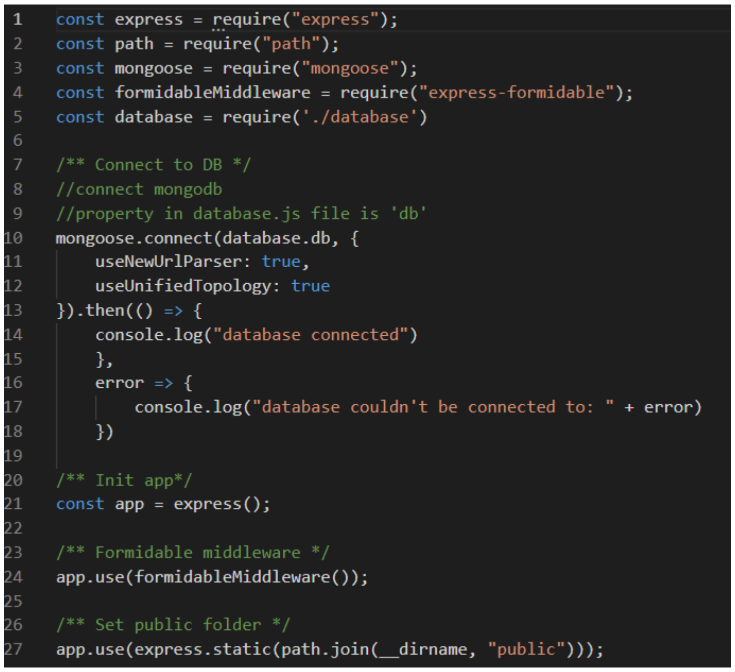
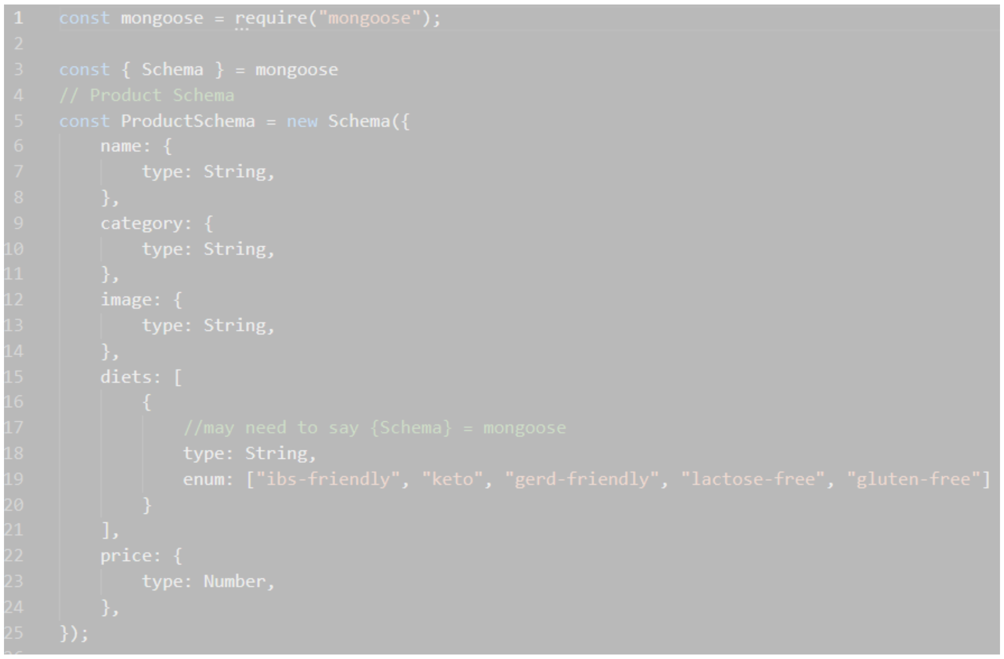


Figure 5)Server Directory Hierarchy



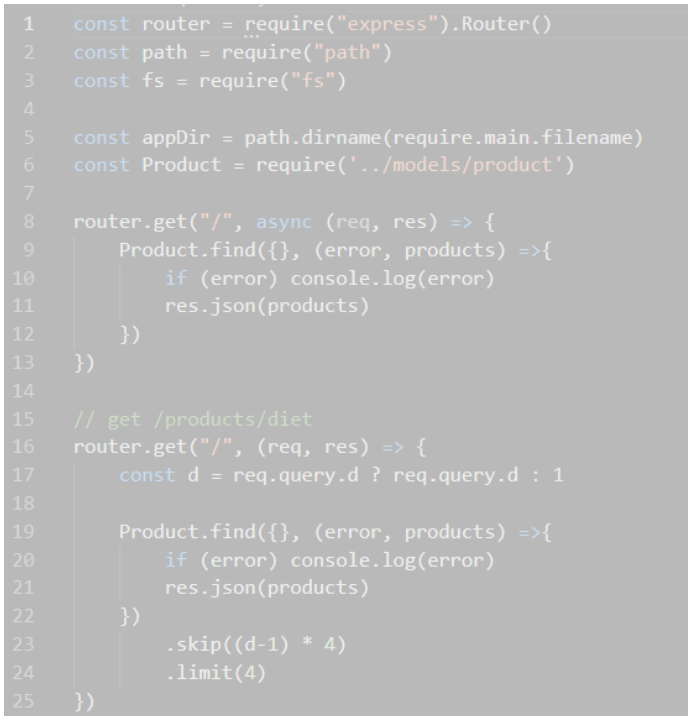
```
const express = require("express");
const path = require("path");
const mongoose = require("mongoose");
const formidableMiddleware = require("express-formidable");
const database = require('./database')

/** Connect to DB */
//connect mongodb
//property in database.js file is 'db'
mongoose.connect(database.db, {
    useNewUrlParser: true,
    useUnifiedTopology: true
}).then(() => {
    console.log("database connected")
    },
    error => {
        console.log("database couldn't be connected to: " + error)
    })

/** Init app*/
const app = express();

/** Formidable middleware */
app.use(formidableMiddleware());

/** Set public folder */
app.use(express.static(path.join(__dirname, "public")));
```

Figure 6)app.js, used to establish connection, instantiate middleware constants, etc.



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

const { Schema } = mongoose
// Product Schema
const ProductSchema = new Schema({
    name: {
        type: String,
    },
    category: {
        type: String,
    },
    image: {
        type: String,
    },
    diets: [
        {
            //may need to say {Schema} = mongoose
            type: String,
            enum: ["ibs-friendly", "keto", "gerd-friendly", "lactose-free", "gluten-free"]
        }
    ],
    price: {
        type: Number,
    },
});
```

Figure 7)Sample Model Code



```
const router = require("express").Router()
const path = require("path")
const fs = require("fs")

const appDir = path.dirname(require.main.filename)
const Product = require('../models/product')

router.get("/", async (req, res) => {
    Product.find({}, (error, products) =>{
        if (error) console.log(error)
        res.json(products)
    })
})

// get /products/diet
router.get("/", (req, res) => {
    const d = req.query.d ? req.query.d : 1

    Product.find({}, (error, products) =>{
        if (error) console.log(error)
        res.json(products)
    })
        .skip((d-1) * 4)
        .limit(4)
})
```

Figure 8)Sample Custom Route Code