FINAL PROJECT

Full Stack Development/Online Store

React
Node/Express
MySql
Bootstrap
CSS

- Store must have 10+ products with store data in database
- Product name
- Image of product
- Price
- Any other useful information (size, special description, color, etc)
- User table

API built using Node.js & Express as well as any other packages necessary (helmet, cors, etc)Front End built in React, using React-Router to create pages

- Home Page
- All Products Page
- Single Product Page
- Error Page
- Form to receive new users => must be stored in database

Must be responsiveMust also use own CSS stylingProject MUST be on github

How to Succeed

- Take time to map out flow of database, api, and front end
- Diagram using flowchart or some other system
- Data model (Google Sheets)
- Figma to design front end

Documentation

- Comments, comments, comments!
- If you use any outside resources, cite them

Commit progress every day (preferably before lunch and at EOD)Set checkpoints before moving forward

- “When in doubt, log it out”
- Debug/Testing

Collaborate! Everyone must do their OWN work; however, you can work together to solve collective problemsFollow accessibility guidelinesDo not procrastinate!

Create db, react, node

DB - (schema, data)
Start with schema

```
DROP DATABASE IF EXISTS carryalls_db;
CREATE DATABASE carryalls_db;
USE carryalls_db;
```

```
CREATE TABLE brand(
    brand_id TINYINT UNSIGNED AUTO_INCREMENT NOT NULL,
    brand VARCHAR(40) NOT NULL,
    CONSTRAINT pk_brand PRIMARY KEY (brand_id)
);
```

```
CREATE TABLE category(
    category_id TINYINT UNSIGNED AUTO_INCREMENT NOT NULL,
    category VARCHAR(40) NOT NULL,
    CONSTRAINT pk_category PRIMARY KEY (category_id)
);
```

```
CREATE TABLE product(
    product_id TINYINT UNSIGNED AUTO_INCREMENT NOT NULL,
    product_name VARCHAR(40) NOT NULL,
    brand_id TINYINT UNSIGNED NOT NULL,
    product_image VARCHAR(40),
    price FLOAT(6, 2) NOT NULL,
    description TEXT,
    date_added TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_update TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
    CONSTRAINT pk_product PRIMARY KEY (product_id),
    CONSTRAINT fk_brand FOREIGN KEY (brand_id) REFERENCES brand
(brand_id)
);
```

```
DROP TABLE category;
```

```
ALTER TABLE product ADD column category enum('tote', 'handbag',
'crossbody', 'clutch');
```

NOTE****** NEXT

1. Add product to category, Add product to brand TABLES
2. FIND AUTO_INCREMENT reset snippet

Put into data.sql

```
INSERT INTO category(category)
VALUES
    ('crossbody'),
    ('tote'),
    ('handbag'),
    ('clutch');
    -- each one is a separate category.. is a string separated by (' '),
(' '), (' ');

INSERT INTO brand(brand)
VALUES
    ('Louis Vuitton'),
    ('Hobo'),
    ('Michael Kors'),
    ('Coach'),
    ('Gucci');

INSERT INTO product(product_name, brand_id, product_image, price,
description)
VALUES
    ("monogram canvas neverfull gm tote", 1, "product1image.jpg",
1275.00, "Lorem Ipsum.");



INSERT INTO product(product_name, brand_id, product_image, price,
description, category)
VALUES
    ("classic hobo handbag", 2, "product_image2.jpg", 348.00, "Lorem
Ipsum", "handbag"),
    ("jet set large print logo", 3, "product_image3.jpg", 378.00, "Lorem Ipsum", "crossbody"),
    ("monogram canvas speedy 30 satchel", 1, "product_image4.jpg", 840.00, "Lorem Ipsum", "handbag"),
    ("lauren clutch wallet", 2, "product_image5.jpg", 148.00, "Lorem Ipsum", "clutch"),
    ("1941 classic shoulder bag", 4, "product_image6.jpg", 350.00, "Lorem Ipsum", "tote"),
    ("gg marmont small shoulder bag", 5, "product_image7.jpg", 1790.00, "Lorem Ipsum", "crossbody"),
    ("ophidia medium tote bag", 5, "product_image8.jpg", 1750.00, "Lorem Ipsum", "tote"),
    ("val duffle bag", 4, "product_image9.jpg", 398.00, "Lorem Ipsum", "tote"),
    ("mia crocodile envelope clutch", 3, "product_image10.jpg", 1290.00, "Lorem Ipsum", "clutch");

-- ALREADY PUT ABOVE INTO MYSQL
```

Copy and pasted (above) into mysql terminal.

START NODE:

Refer to sticky note on node.
Open new terminal….

Open node in VScode

PACKAGE.JSON (MAKE CHANGES)

CREATE server.js file FIRST

THEN:

1. Create app folder
    a. Create folders:
        i. Config
        ii. Daos
        iii. Routes

2. Inside config:

    Create a file => dbconfig.js

3. Inside daos:

    Create 2 folders:

    a. api
    b. Common

    ALSO, create a file inside daos (outside of api and common)

    a. daos.js

Inside common:

a. daoCommon.js

Inside api:

    (this is where your individual daos will go)

4. Inside routes folder:

    Create a router.js file (overall router)

    Create an api folder (will have your different routes)

AFTER SETTING UP FILE STRUCTURE:

IN server.js

FIRST:

```
const express = require('express')
const server = express()
```

SECOND:

```
const PORT = process.env.PORT || 3005
```

THIRD:

```
server.listen(PORT, ()=> console.log(`Port ${PORT} is listening to
classy carryalls.`))
```

FORTH:

```
const router = require('./app/routes/router')
```

Comment this out for now

```
//localhost:3005 => router.js
server.use('/', router)
```

Comment out for now

FIFTH:

```
const helmet = require('helmet')
const cors = require('cors')
```

```
server.use(helmet.contentSecurityPolicy({
    userDefaults: true,
    crossOriginResourcePolicy: false,
    crossOriginEmbedderPolicy: false,
    directives: {
        "img-src": ["'self'", "https: data"],
        "scriptSrc": ["'self'", "cdn.jsdeliver.net"]
    }
}))
```

This handles security

```
server.use(cors())
    .use(express.json())
    .use(express.urlencoded({extended: true}))
```

IN router.js

```
const express = require('express')
const router = express.Router()
const PORT = process.env.PORT || 3005




module.exports = router
```

```
router.get('/api', (req, res)=> {
    res.json({
        'All Products': `http://localhost:${PORT}/api/product`,
        'All Brands': `http://localhost:${PORT}/api/brand`

    })
})
```

```
router.use('/api/product', require('./api/productRoutes'))
router.use('/api.brand', require('./api/brandRoutes'))
```

Comment out brand line for now

```
router.get('*', (req, res)=> {
    if (req.url === '/favicon.ico') {
        res.end()
    } else {
        res.send('<h1>404 ERROR: Page does not exist</h1>')
    }
})
```

IN dbconfig.js

```
const mysql = require("mysql2")

const pool = mysql.createPool ({
    connectionLimit: 10,
    host: "localhost",
    user: "root",
    password: "root",
    database: "WHATEVER DATABASE YOU ARE USING"
})

module.exports = pool
```

IN gitignore,

Add so no one can see username and password.

```
node_modules/
*-lock.json

app/config/dbconfig.js
```