IDEA and USERS

What problem(s) does your app solve?
How does it solve those problems?
Who is your target user?
How much experience do they have with technology?

My app will allow the user to look through a variety of products on my online store, which they will then be able to add to a cart and checkout.

I would code and setup my database(w/ PostgreSQL Node.js, Massive, and Express) which will hold information such as the users credentials and their shopping cart. Then, I will setup my website using React and HTML and CSS.

My target is the typical e-Commerce user. This would be beneficial because they would be familiar with the setup.

FEATURES

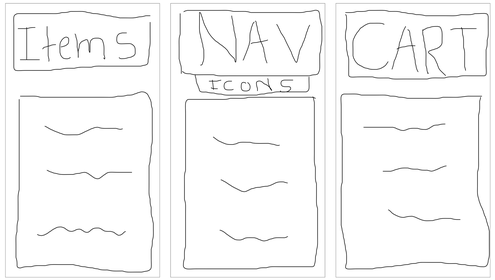
Write use cases for your app.
IE - As a user I can store a list of my favorite food.
Use color to separate features by 'Minimum Viable Product' and 'Bonus features'

MVP Features:
I will create components that will have their own tasks to complete with the of React and JS.
Using CSS, I will make my website aesthetic to the user.
Connect Heroku Data Base to update and store the users informatioon.
Use axios to authenticate endpoints.

Post MVP Features:
Complete any extra or needed touch-ups to the front-end.
Succeed in having a theme.

VIEW

What views do you need to create to meet each feature in your app?
How will the user get to each view?
Revisit this regularly. Simplify each time. Focus on the user.
Make a connection to the controller behind the view.
Draw each view and map out how the user will get to each of them.



The user would get to each view by selecting the different tabs for each page.

```
<Switch>
    <Route path ='/cart' Component={Cart}>
    <Route path ='/items' Component={Items}>
    <Route path ='/login' Component={Login}>
    <Route path ='/home' Component={Home}>
    <Route path ='/register' Component={Register}>
</Switch>
```

CONTROLLERS

Create a controller for each wireframe you made.
Make a connection to each endpoint it needs to use to get its data.

```
getCart: (req, res) => {

const db = req.app.get('db')
db.products.read_cart()
.then(cart => res.status(200).send(cart))
}

deleteCart: (req, res) => {

const db = req.app.get('db')
db.products.delete_cart()
.then(_ => res.sendStatus(200))
}


login: async (req, res) => {
        const { username, password } = req.body;
        const db = req.app.get('db');
        const result = await db.user.find_user_by_username([username]);
        const user = result[0];
        if(!user) {
            return res.status(403).send("No such user.");
        }
        const isAuthenticated = bcrypt.compareSync(password, user.password);
        if(!isAuthenticated){
            return res.status(403).send('Incorrect Password!');
        }
        req.session.user = { username: user.username, id: user.id };
        console.log(req.session)
        return res.send(req.session.user);
    }


register: async (req, res) => {
        const { username, password } = req.body;
        const db = req.app.get('db');
        const result = await db.user.find_user_by_username([username]);
        const existingUser = result[0];
        if(existingUser) {
            return res.status(409).send('Username taken.');
        }
```

ENDPOINTS

List the url, REST method, and a sample of the data being sent or received for every endpoint you need.
Make a connection to the parts of the schema you need to access from each endpoint to return or save the data needed.

- app.post('/api/auth/register', register)
-registers the user and adds them to database.

-.app.post('/api/auth/login', login)
-Logs the user in after verifying credentials in database.

-app.get('/api/products', getProducts)
-Displays all items in the database.

app.get('api/cart', getCart)
-Portrays current items held in the shopping cart.

app.delete('/api/cart', deleteCart)
-Deletes all the users items in the shopping cart.

SCHEMA

Create your schema here including your data types.

```
CREATE TABLE website_users (
    id SERIAL PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(50) NOT NULL,
);

CREATE TABLE website_cart(
id SERIAL PRIMARY KEY,
product_name VARCHAR(60) NOT NULL,
product_price DECIMAL,
product_quantity INT,
product_image TEXT,
user_id INT REFERENCES website_users(id),
date_added TIMESTAMP
);

CREATE TABLE website_products(
id SERIAL PRIMARY KEY,
product_name VARCHAR(60) NOT NULL,
product_price DECIMAL,
product_quantity INT,
product_image TEXT,
);
```

TRACKER

Views and Routes	Controllers and services with staged data in service.	Test front end	Create End points	Move staged data from service to server	Test endpoints with postman	Test front end with server	Replace staged data with queries	Test with Postman	Test end to end	Get site hosted