Objective: Modify E-bikes project to Car Dealers website.

In this exercise, you will make UI modifications to the E-bikes project to convert it to car dealer’s application. The existing images and logos in the E-bikes project are replaced with car images and logos and also the data-model, UI and data is modified according to Car data. The project is modified locally and then pushed/pulled to/from the scratch org. Finally push the modified project from your local git repository to the remote GitHub repository.

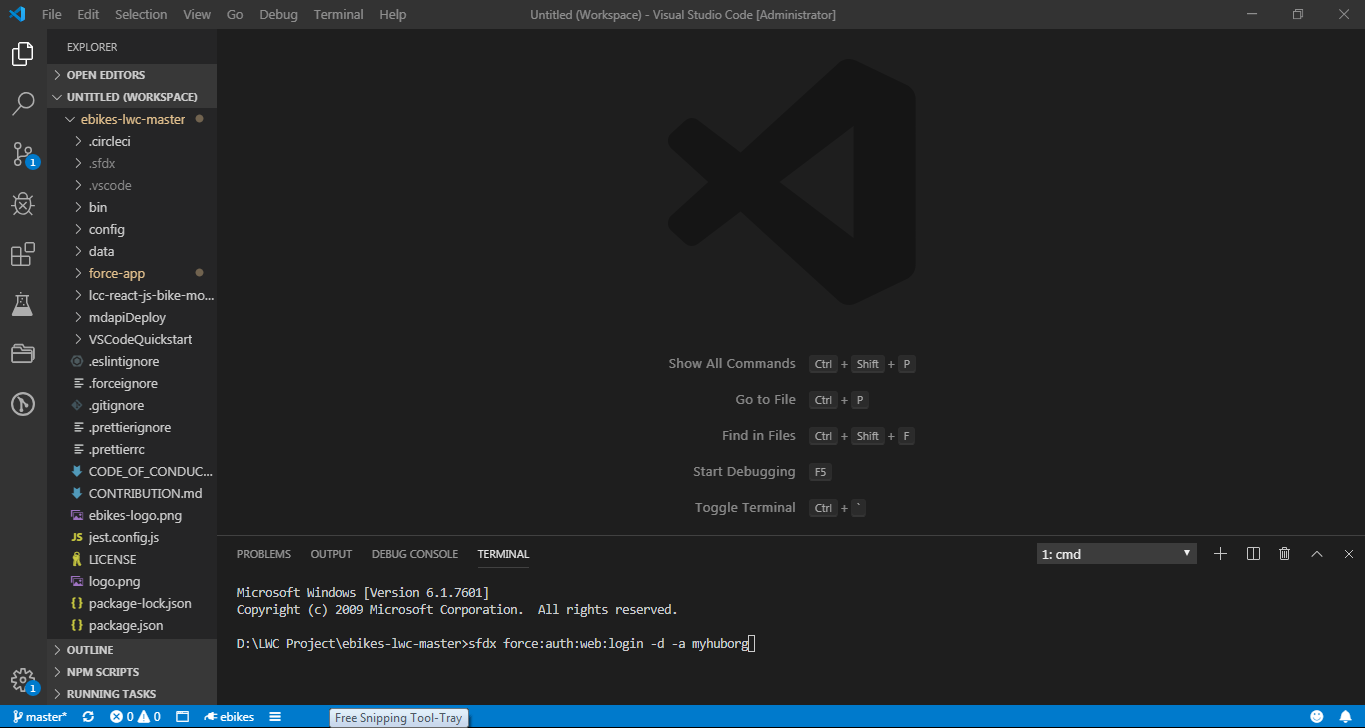
//They wont be pushing it to git again

Step 1.1: Create a new scratch org

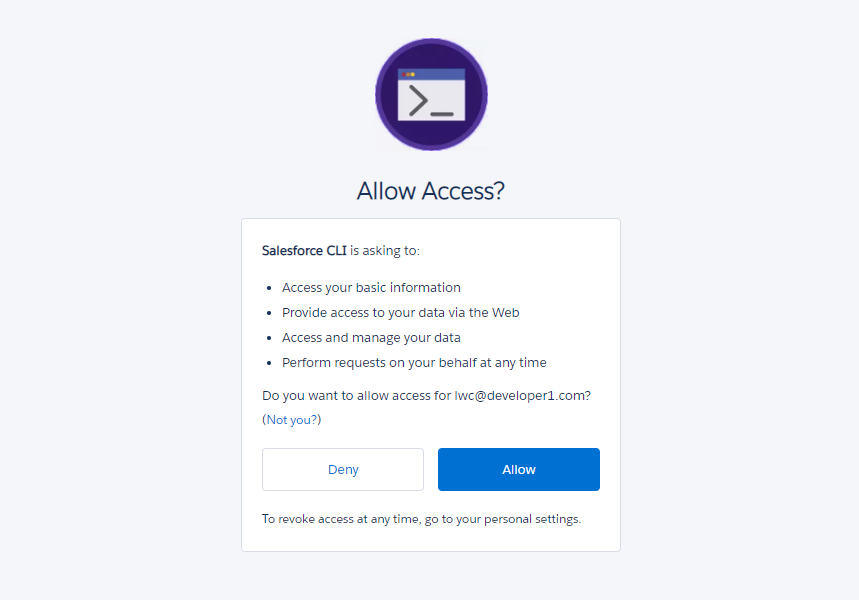
Step 1.1.1: Open ebikes project in your Visual Studio Code.

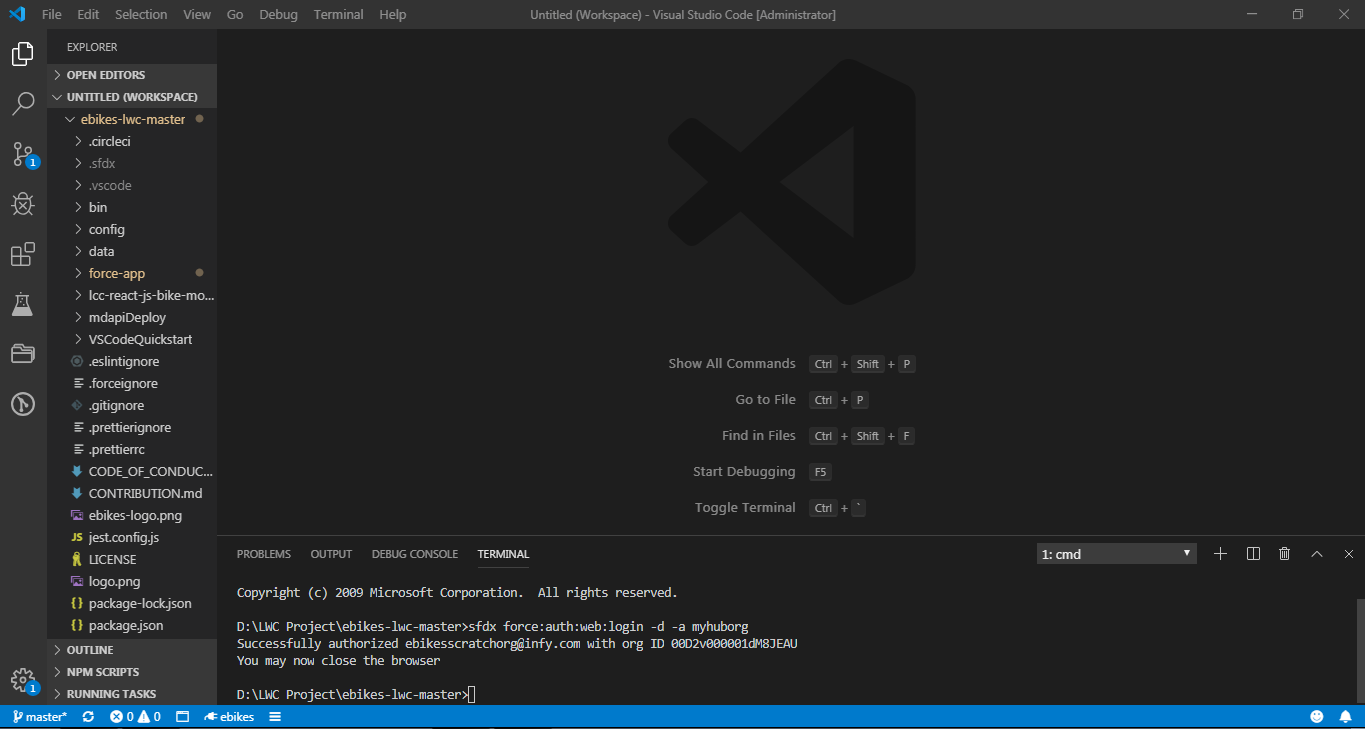
Step 1.1.2: Authenticate your hub org and provide it with an alias (myhuborg in the command below) in the terminal:

sfdxforce:auth:web:login -d -a myhuborg



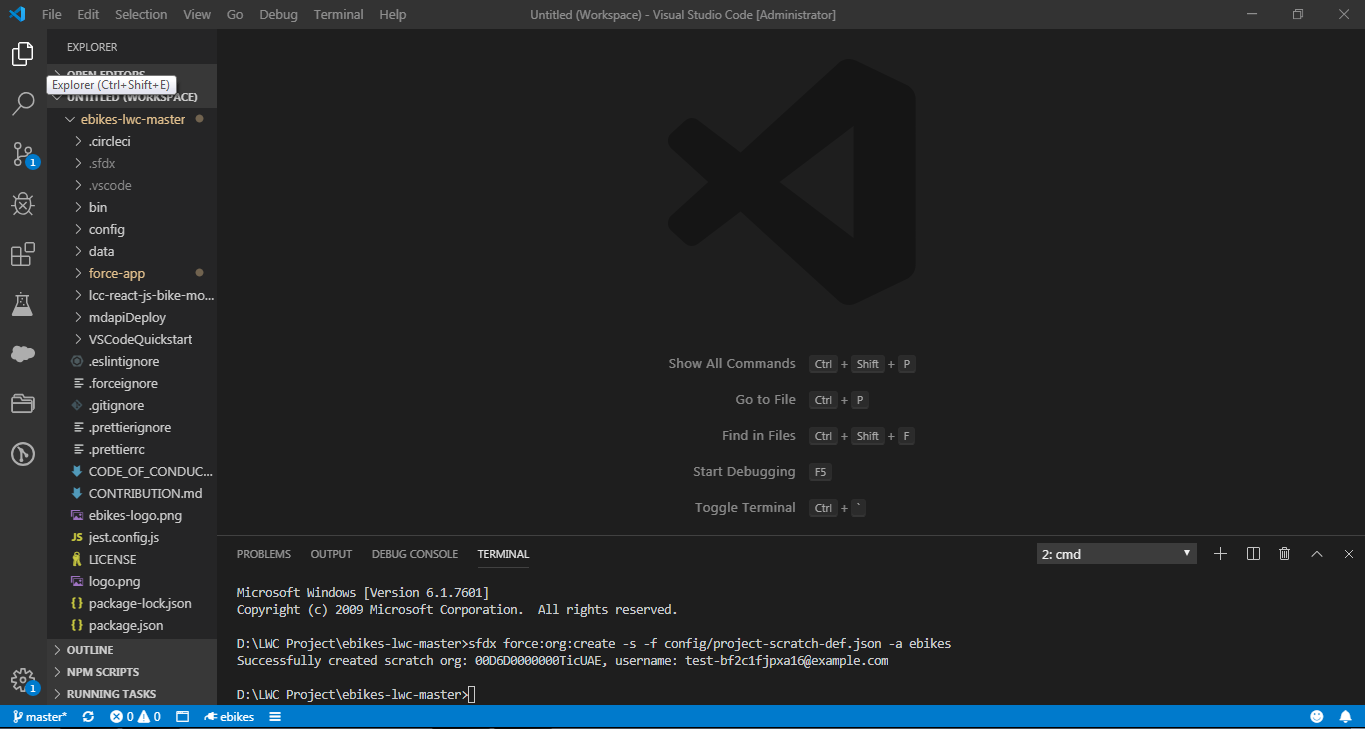
Step 1.1.3: Login to your org and authorize





Step 1.1.3: Create a scratch org and provide it with an alias (ebikes in the command below):

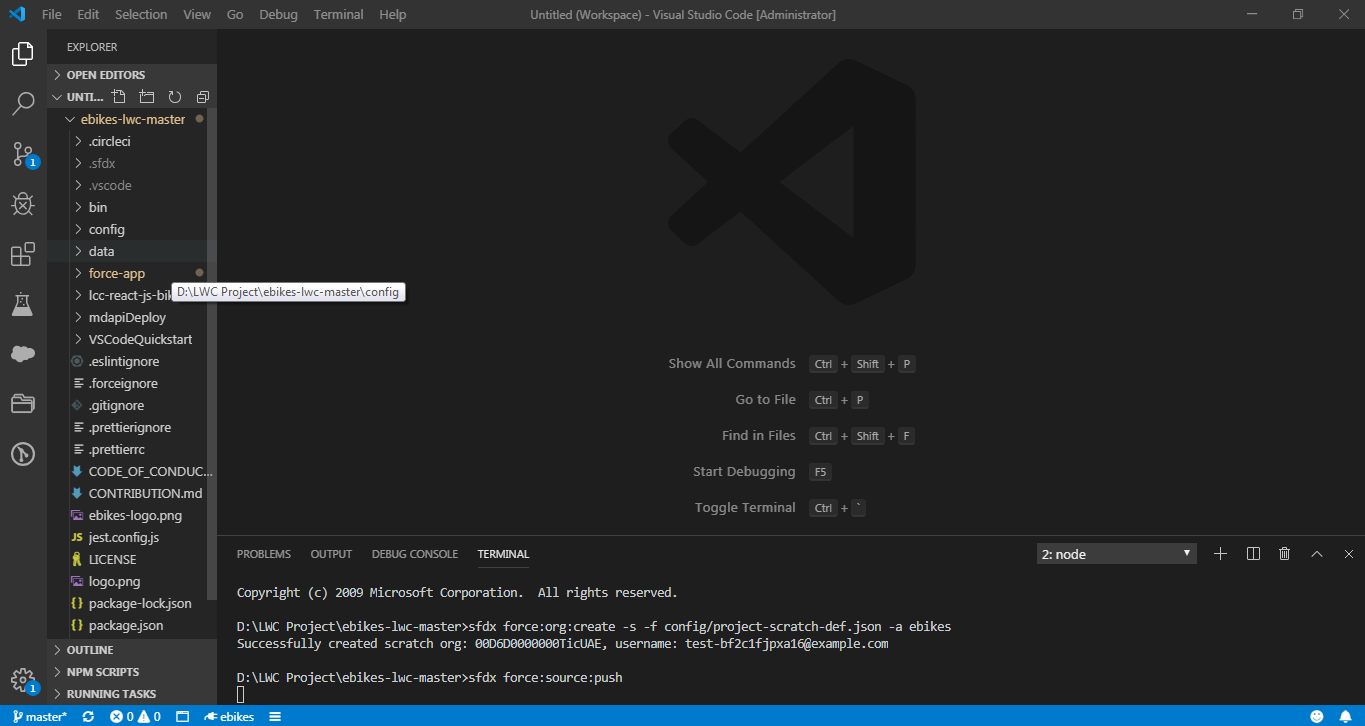
sfdxforce:org:create -s -f config/project-scratch-def.json -a ebikes



Step 1.2: Assign Permissions, Change the data model.

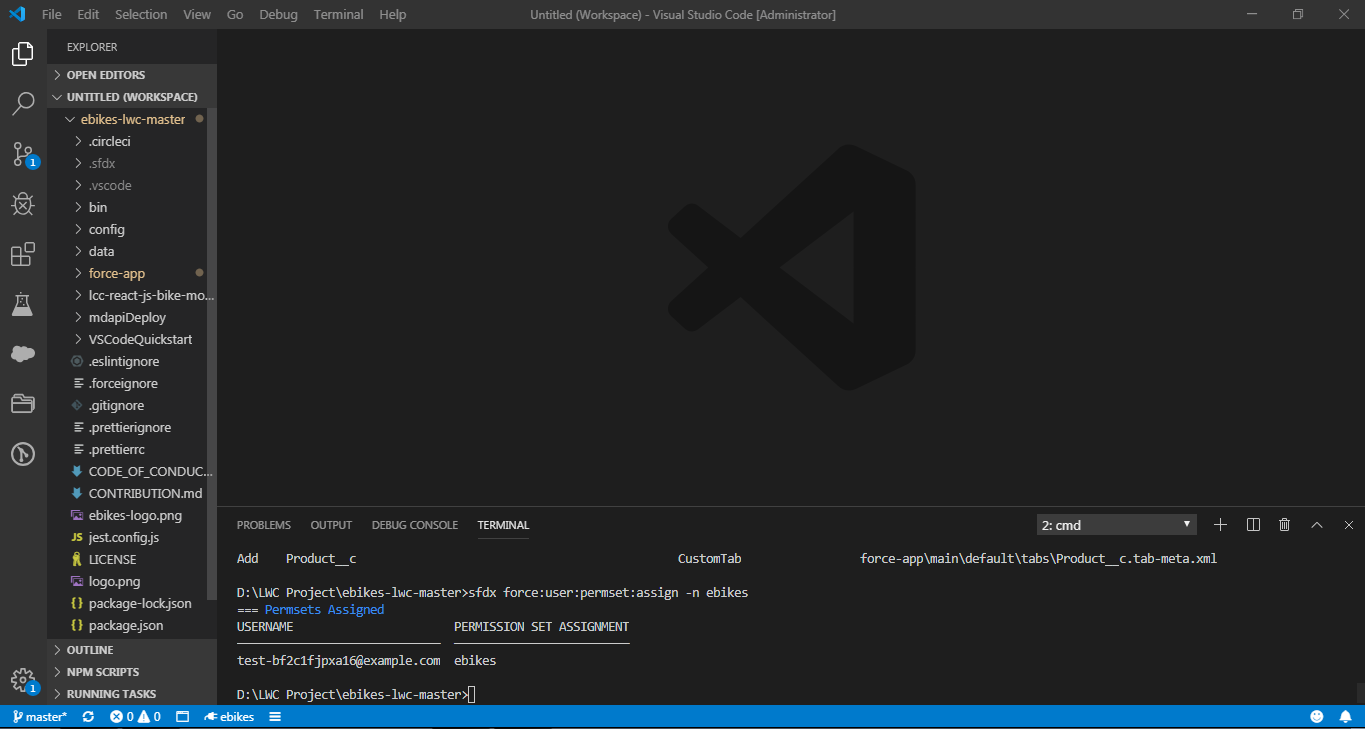
Step 1.2.1: Push the app to your scratch org:

Sfdx force:source:push



Step 1.2.2: Assign the ebikes permission set to the default user:

Sfdx force:user:permset:assign -n ebikes



Step 1.2.3: Push the app to your scratch org:

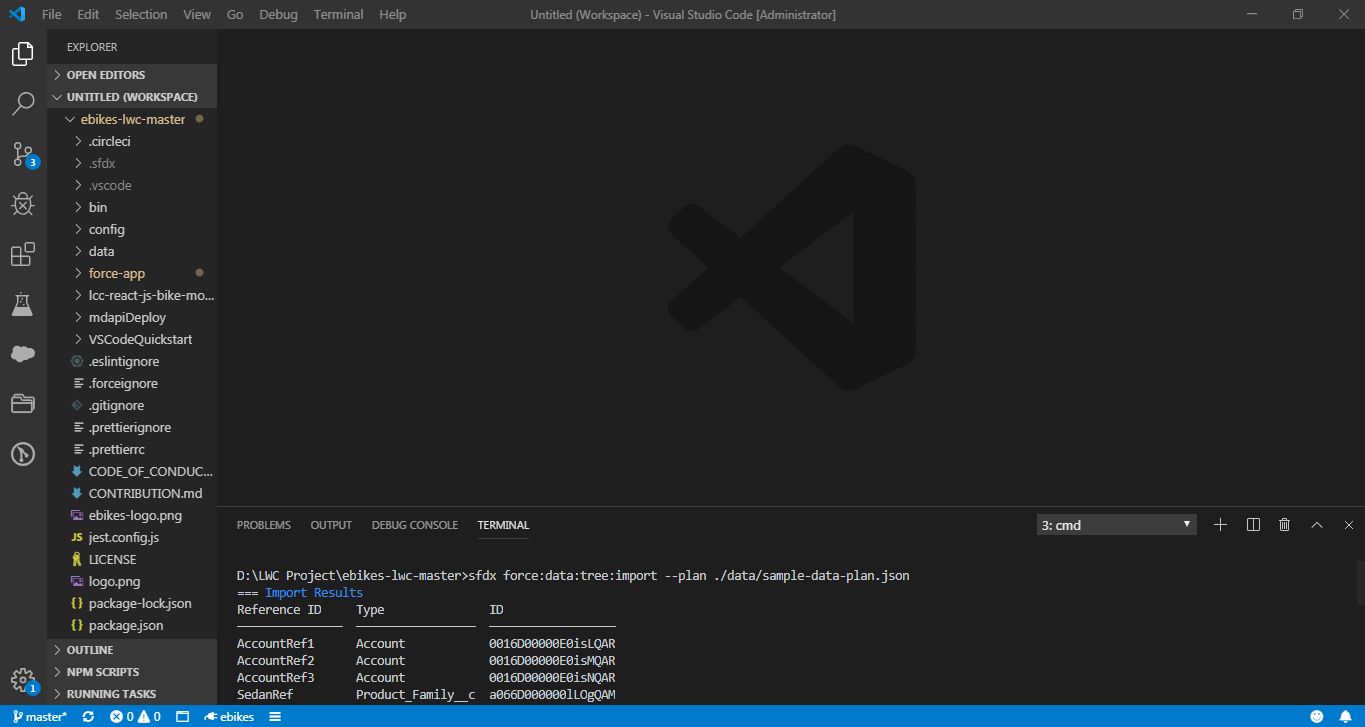
Sfdx force:source:push

//why pushing again?

/\*\*\*

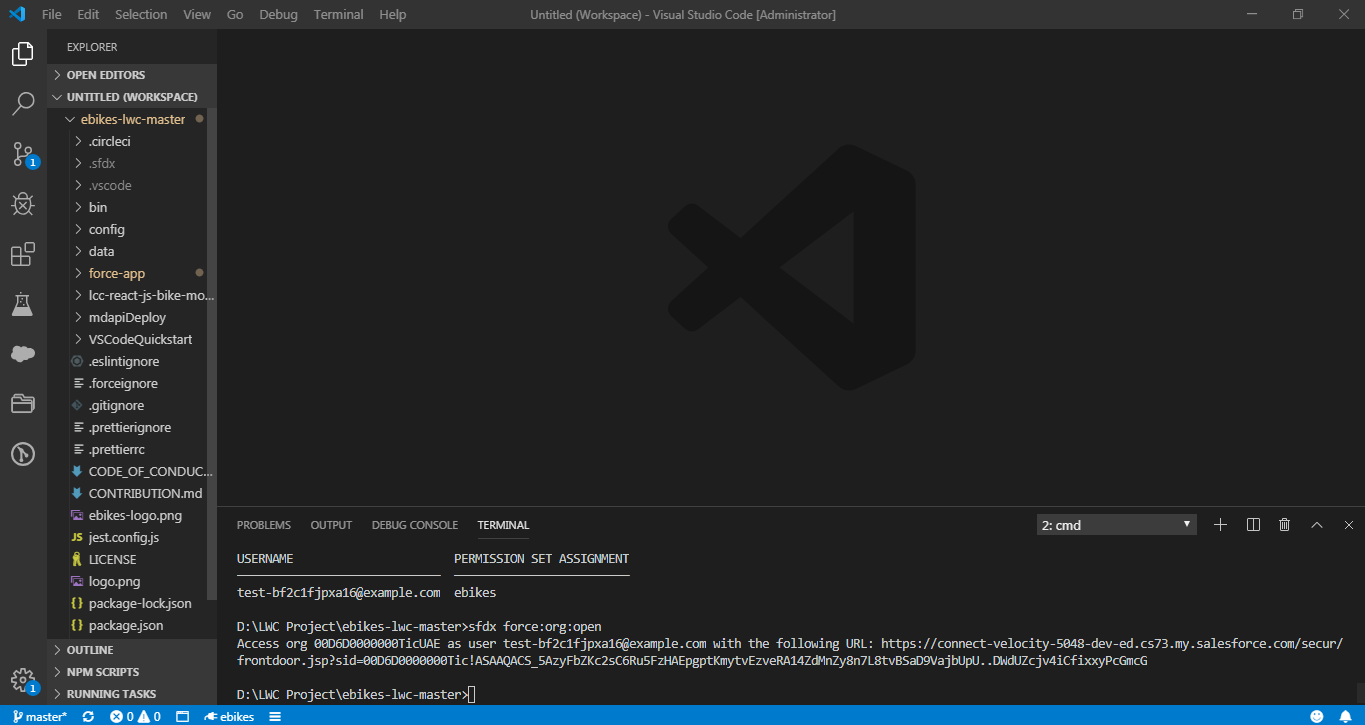
1.2.3: Load Sample Data

sfdx force:data:tree:import --plan ./data/sample-data-plan.json



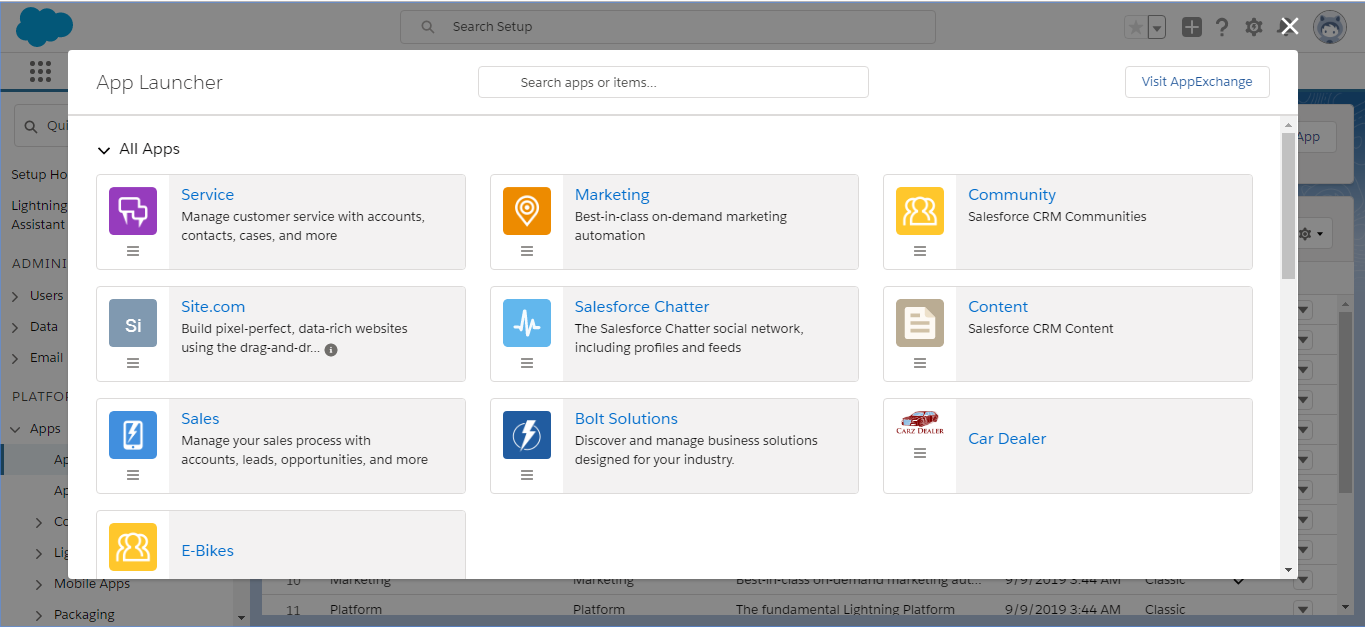
1.2.4. Open scratch org and explore Ebikes app

sfdxforce:org:open



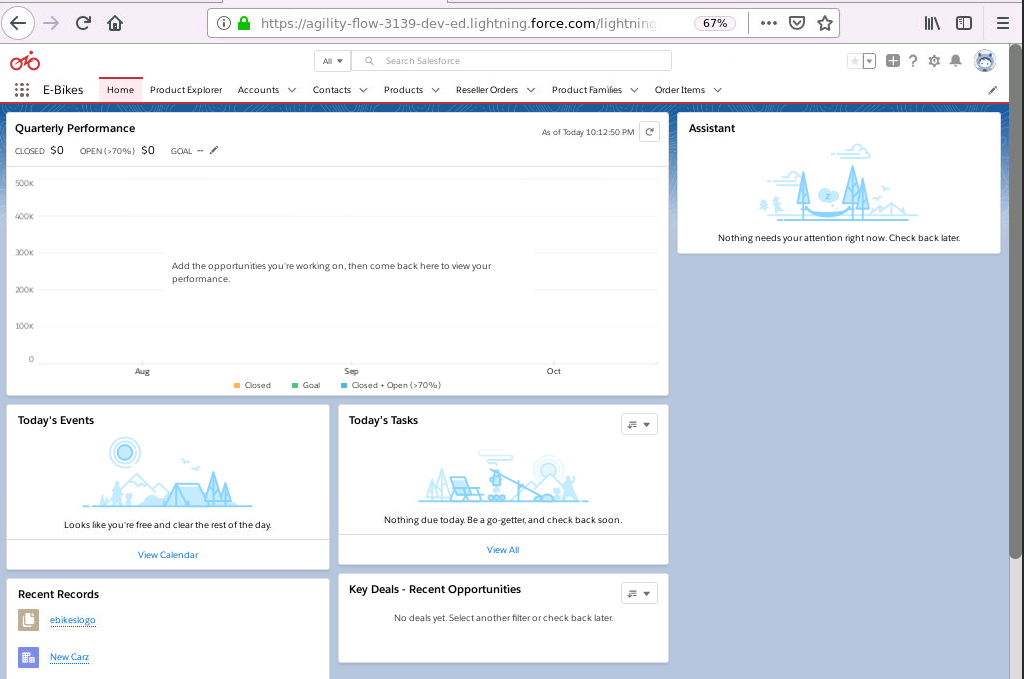
Once the scratch Org opens, go to App Launcher and open E-bikes app.

//Replace below image with image that shows E-bikes app instead of Car Dealer. I don’t have that.

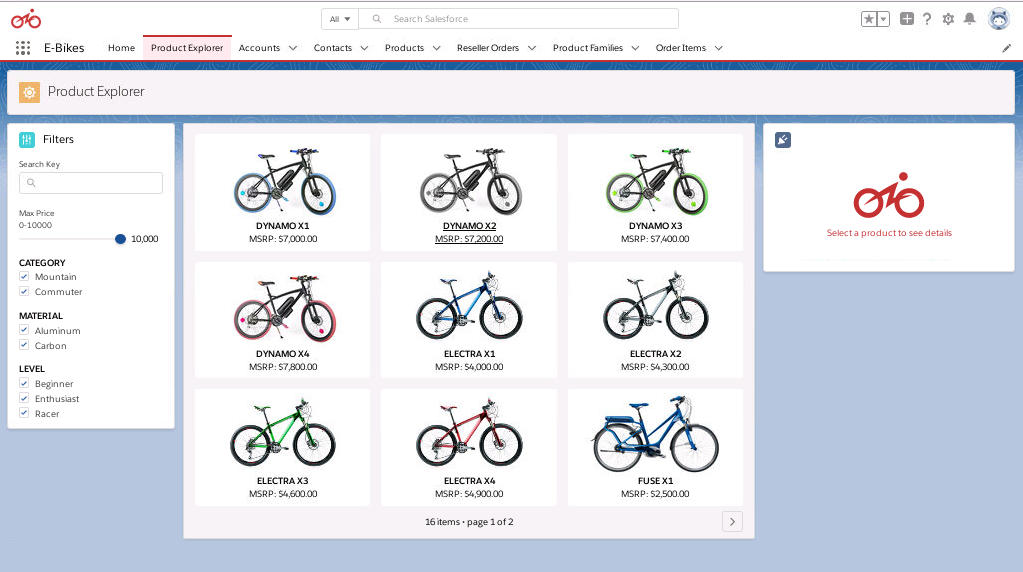


Explore the E-bikes app.

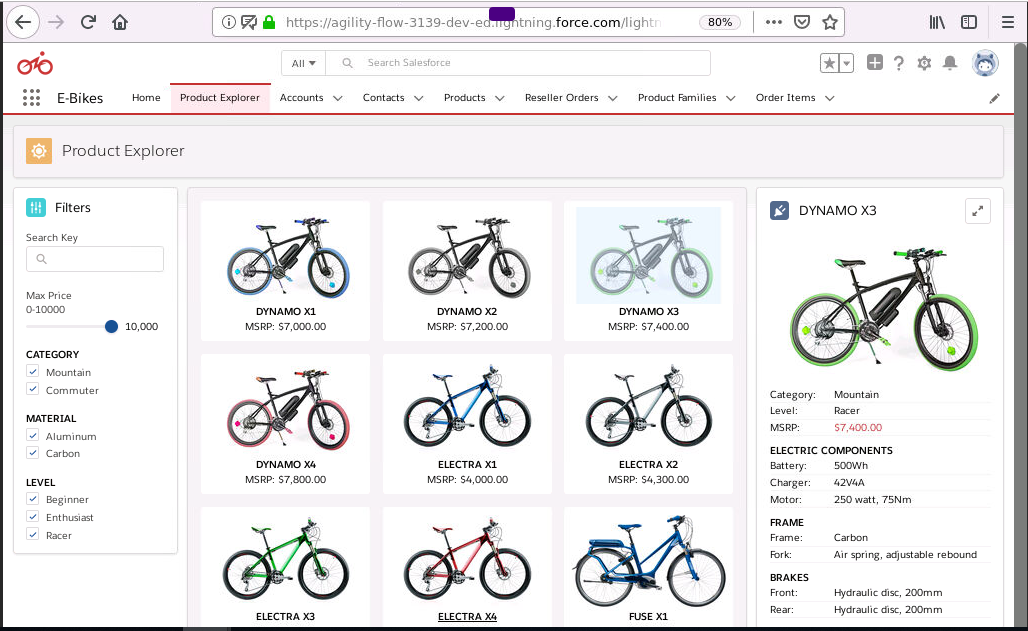
Home Page:



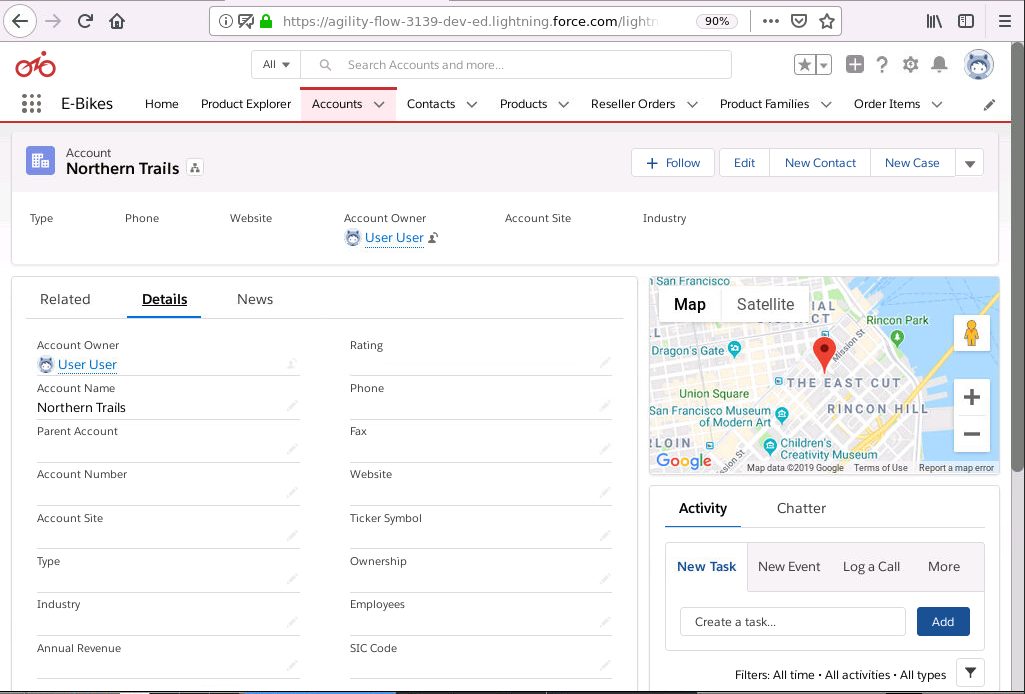
Product Explorer Page:



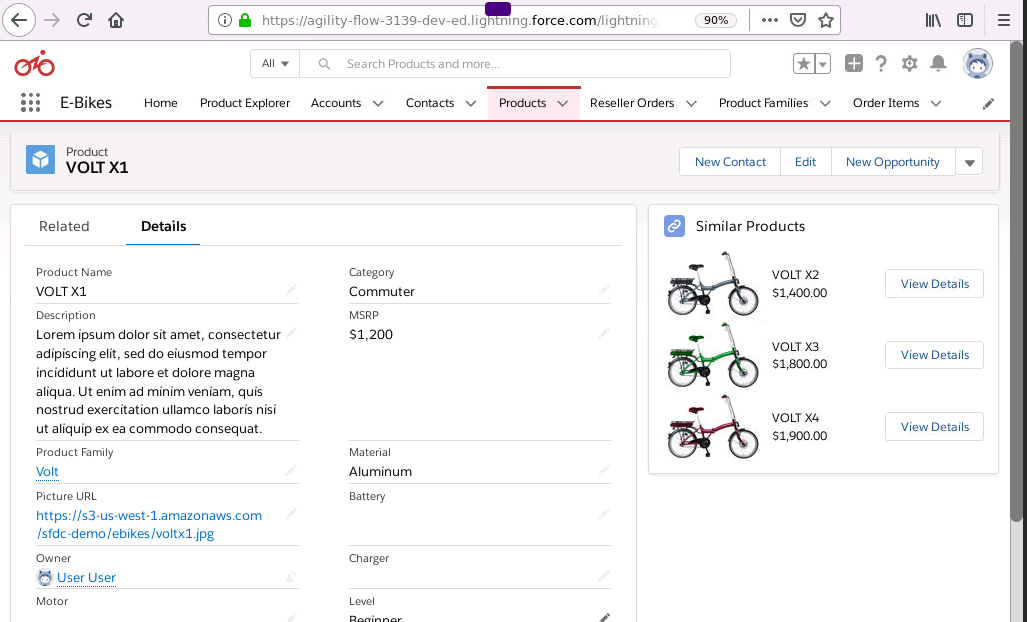
Product Explorer Page after clicking on a Product:



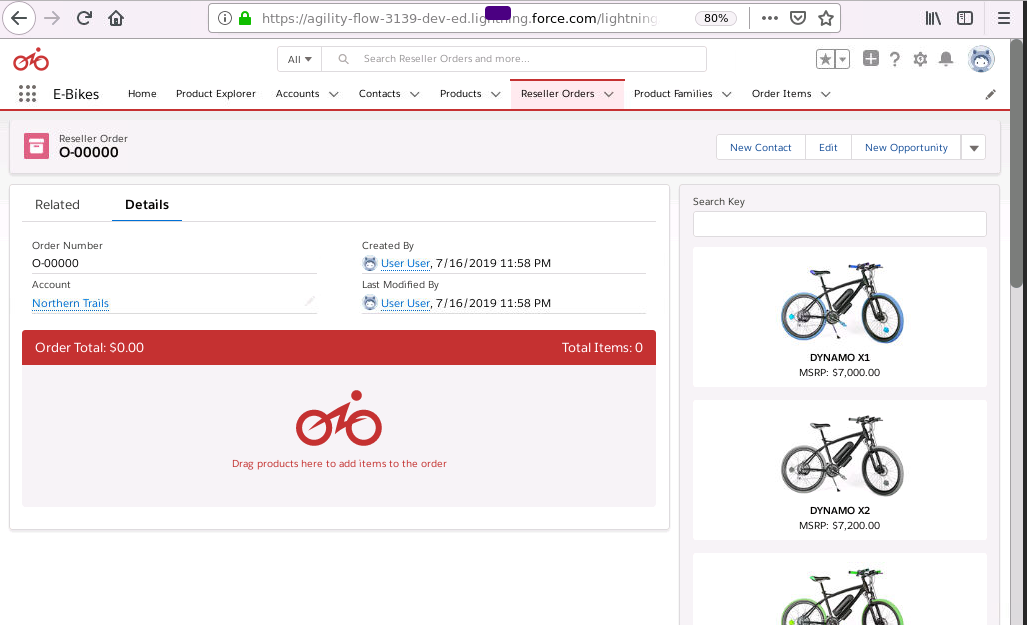
Account Record Page:



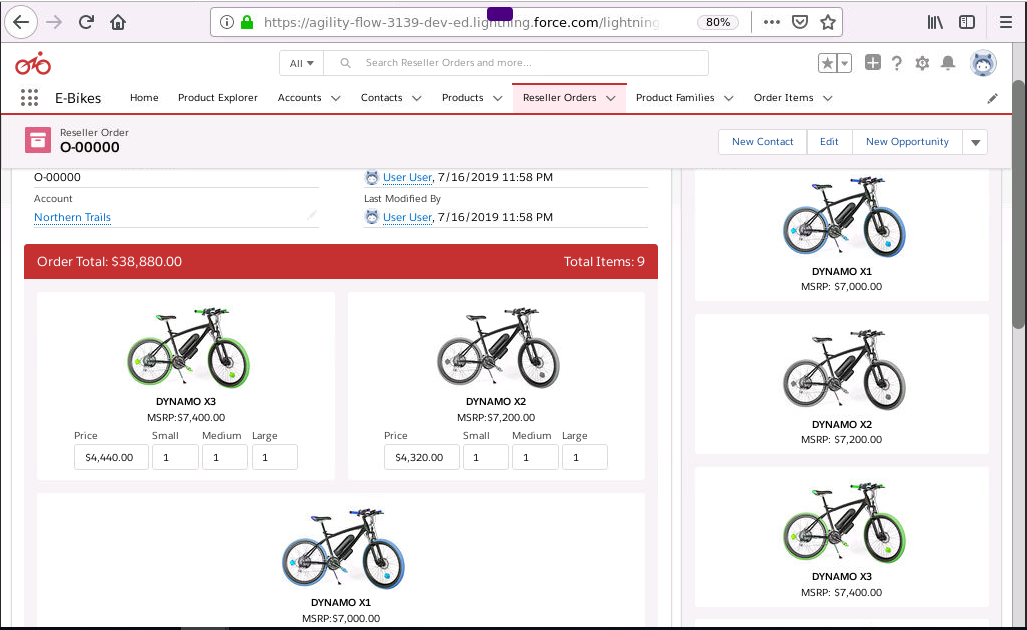
Product Record Page:



Reseller Order Record Page:



Reseller Order Record Page after dragging and dropping products:



\*\*\*/

//Now Modification of E-bikes to Car Dealer comes into Picture

Step 1.3: Change the data model.

Step 1.3.1: Open your scratch org:

sfdxforce:org:open

//No need to open since it is already open.

/\*\*\* Here on the steps will be:-

1)Modify the data model and do branding

2)Pull changes to vs codes

3)Delete scratch Org

4)Replace logos, add product images, replace json files.

5)Create scratch Org again and push code.

6)Load the new Data and open scratch org.

7)Make UI changes by making changes in the code of each component, push it and observe the changes.

\*\*\*/

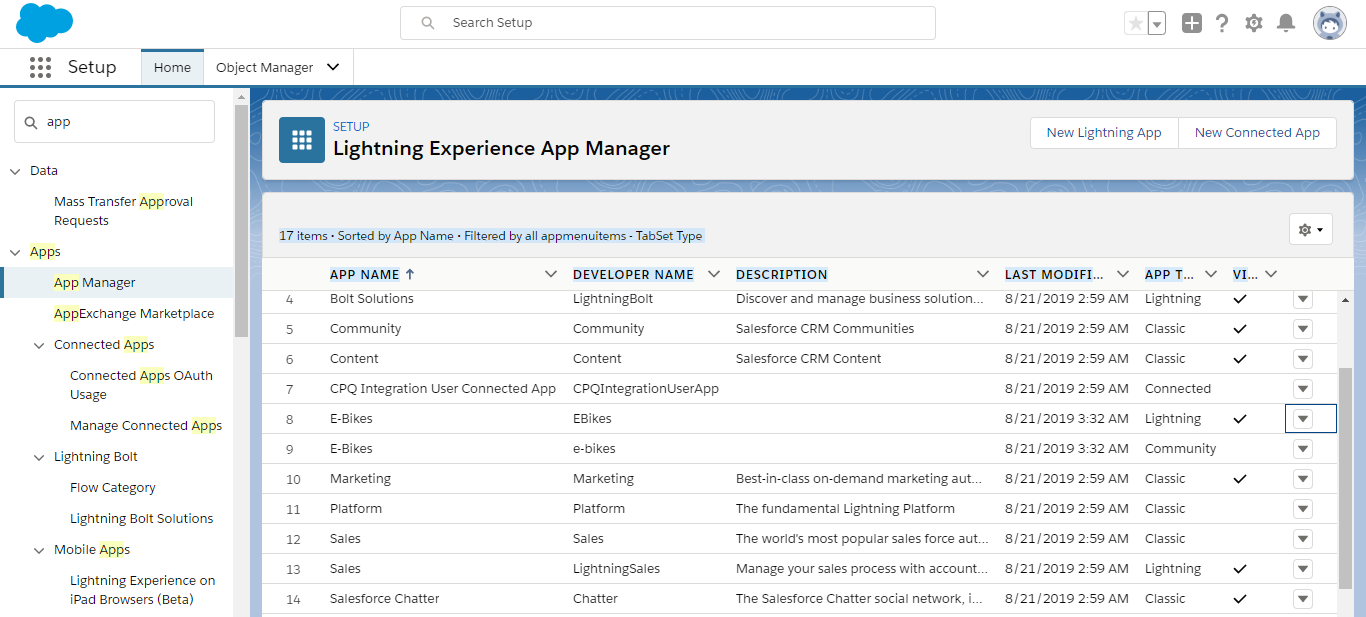
Step 1.3.2: Change the data model in your scratch org

//table

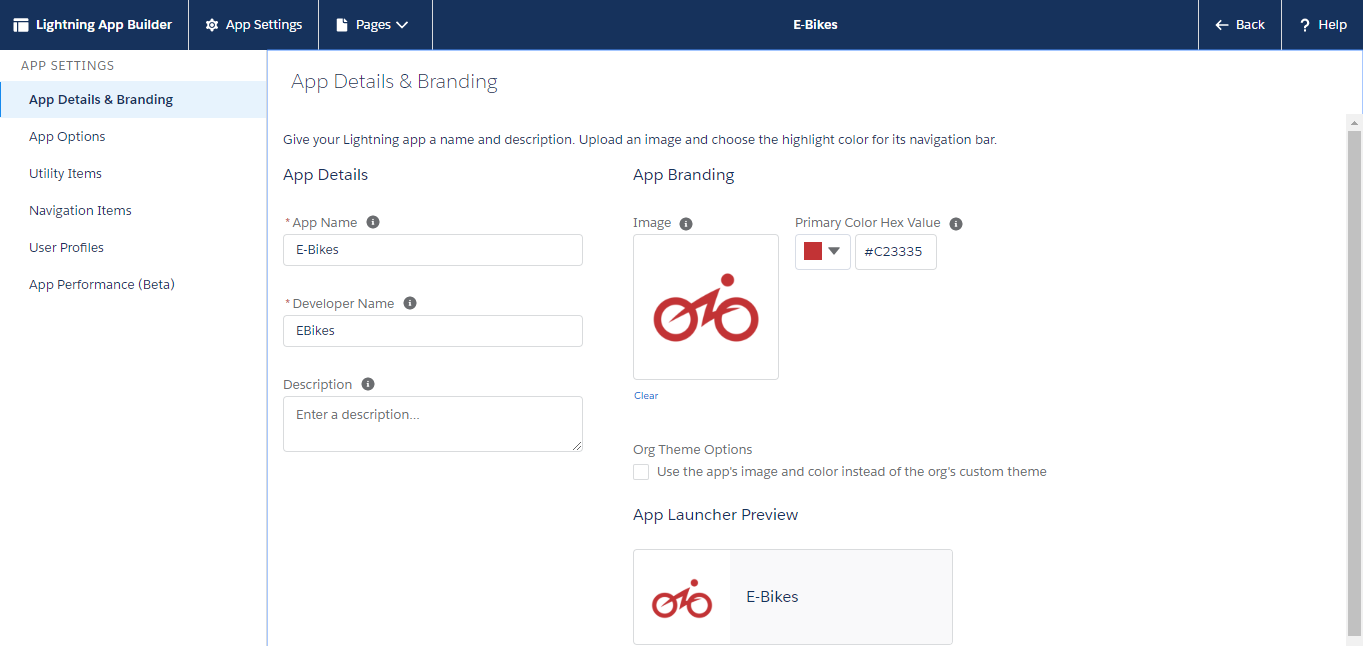
//Hide unwanted fields from Product layout.

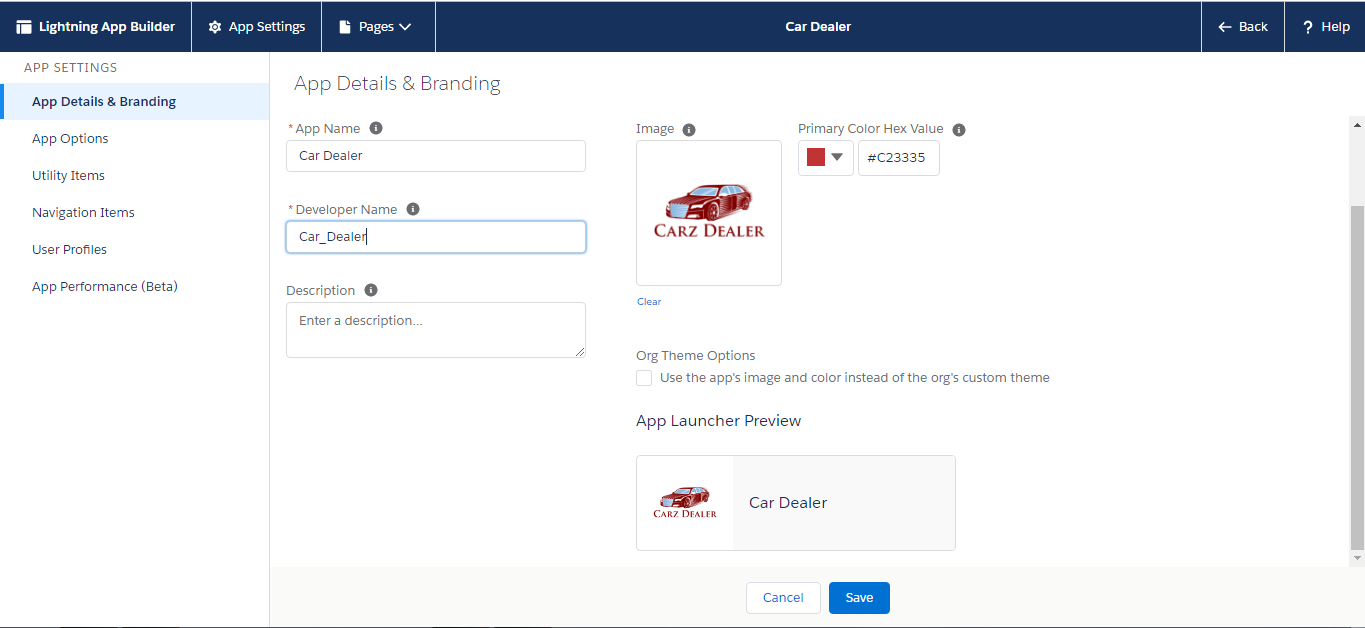
Step 1.4: Branding

Step 1.4.1: In Quick find search for App Manger. Edit the E-bikes app.



Step 1.4.2: Under App Details and Branding change the Image to the new logo and App name to Car Dealer.

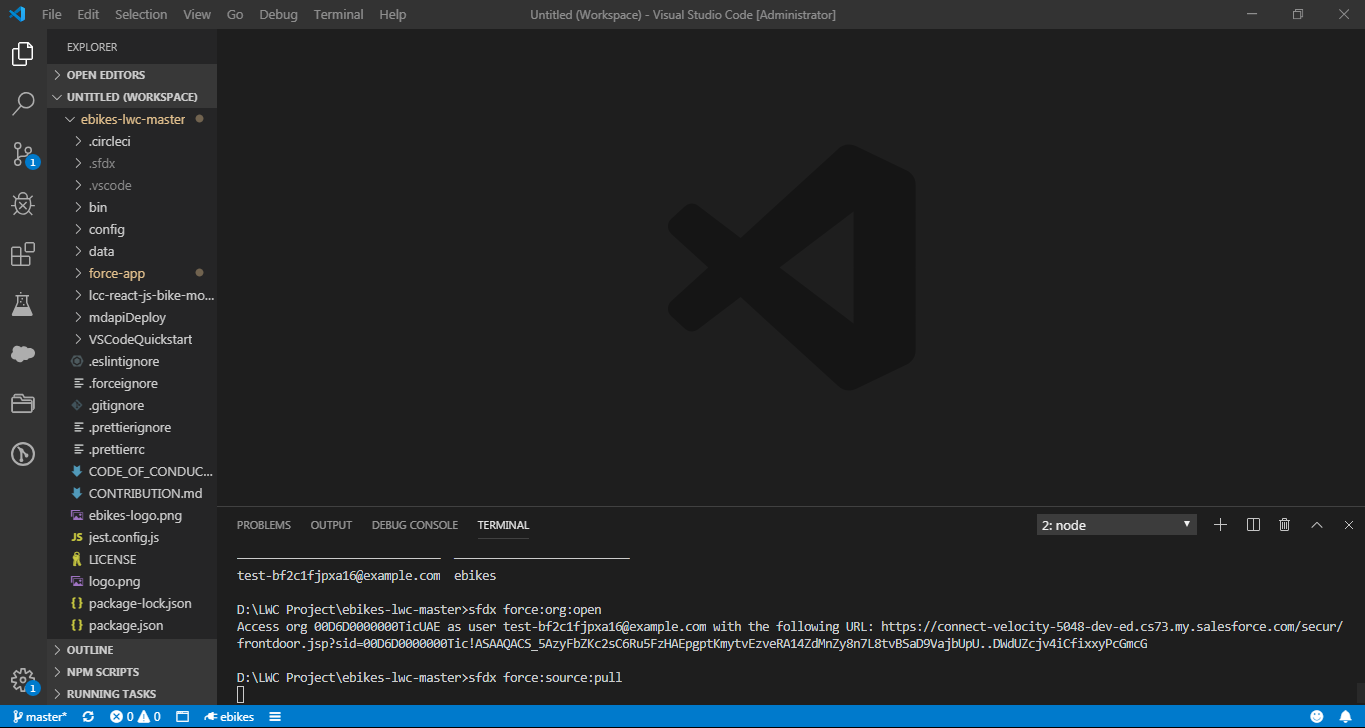




Step 1.4.3: Click Save.

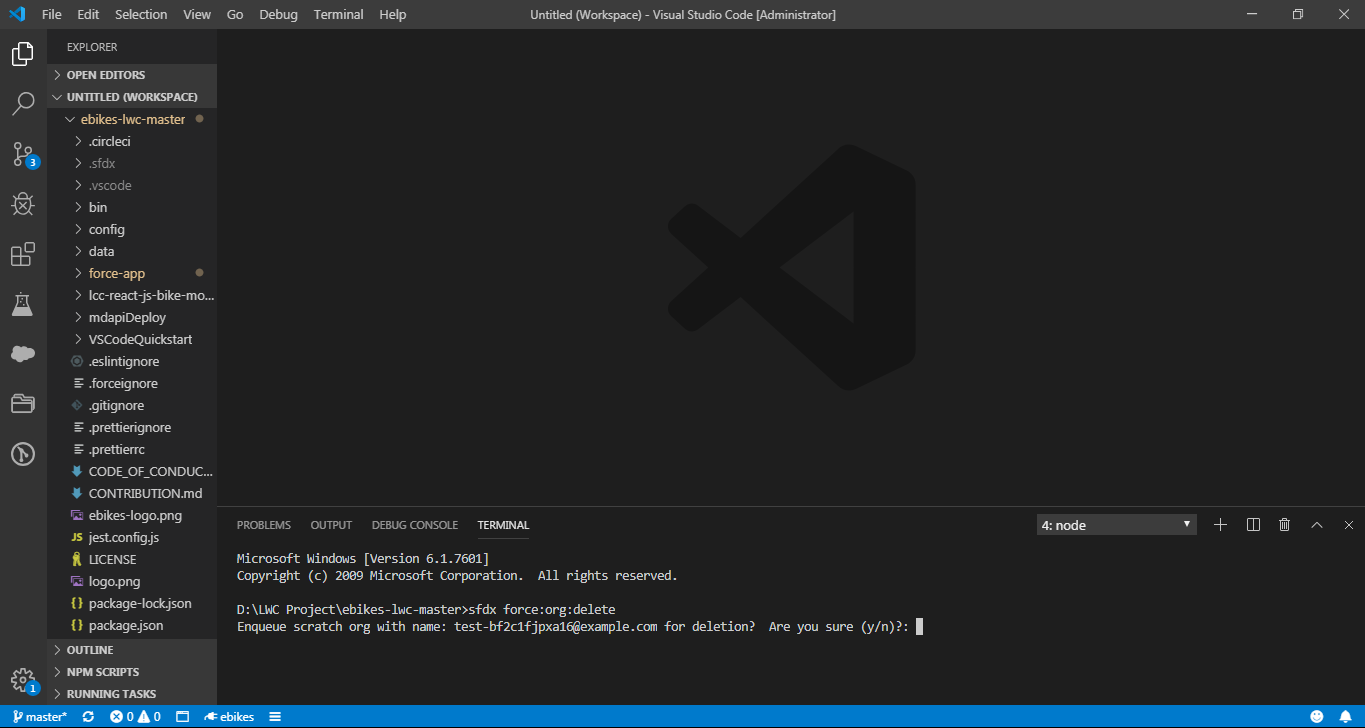
Step 1.4.4: Pull the changes:

sfdxforce:source:pull

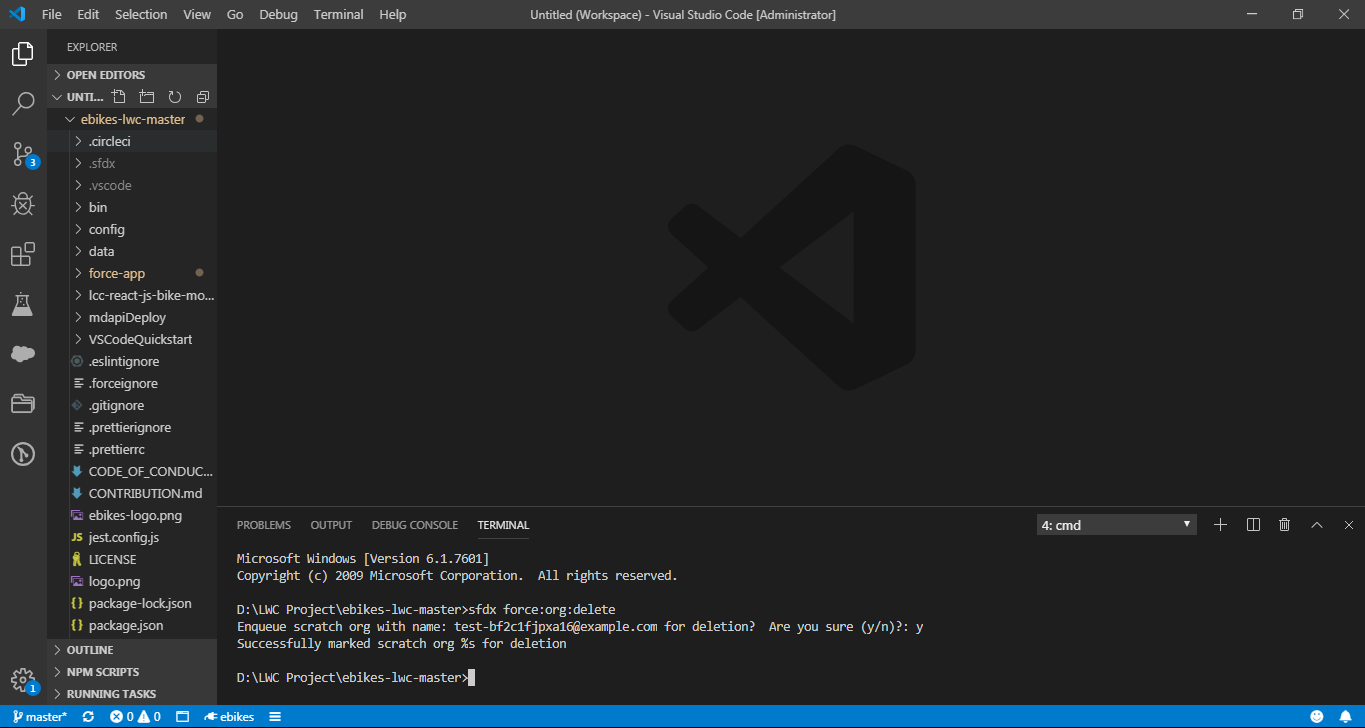


//Step 1.4.5: Delete Scratch Org

sfdx force:org:delete



Press Y

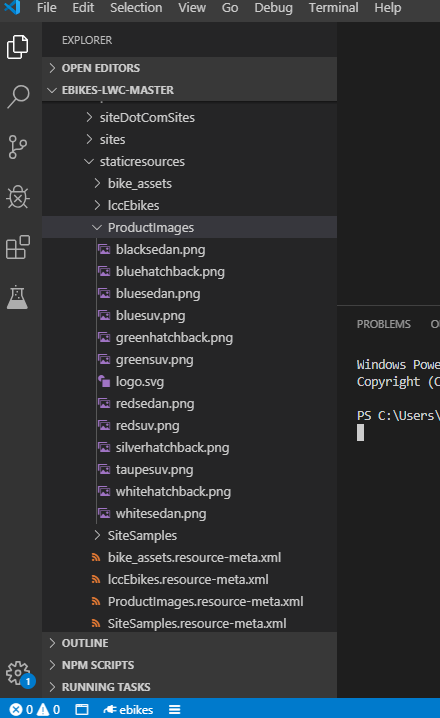


Step 1.5: Add static resource to your org

Step 1.5.1: Create a folder ProductImages in force-app>main>default>staticresources.

Step 1.5.2: Place images of cars inside this folder

//provide folder with the same car images from git



Step 1.5.3: Create a meta.xml file inside staticresources with the file name :

**ProductImages.resource-meta.xml**

Replace the existing code with the code given below:

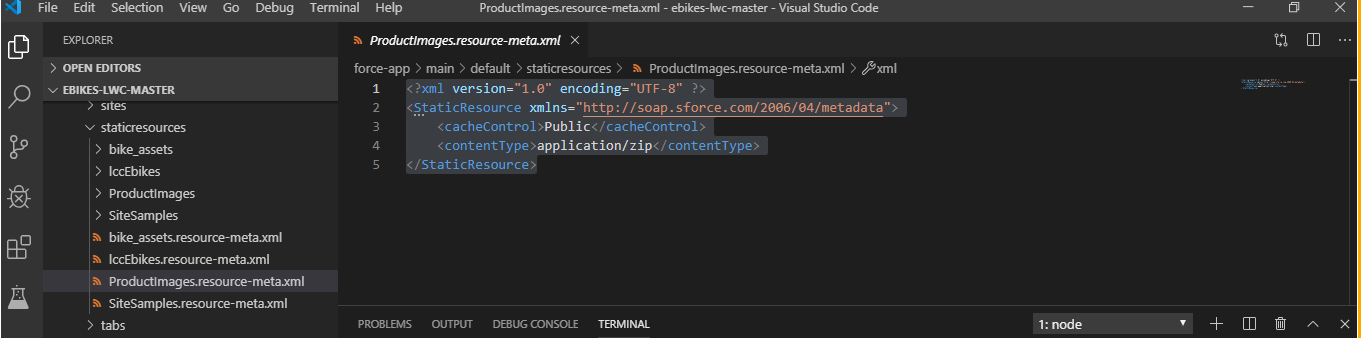
<?xml version="1.0" encoding="UTF-8" ?>

<StaticResourcexmlns="http://soap.sforce.com/2006/04/metadata">

<cacheControl>Public</cacheControl>

<contentType>application/zip</contentType>

</StaticResource>



Step 1.6: Replace existing E-bikes images with new ones in the code

Step 1.6.1: Replace ebikes-logo.png and logo.png images with the new logo.

// Replace ebikes-logo.png and logo.png images with the corresponding new logo files.

Step 1.6.2:Open the file Product\_\_cs.json inside data and change the "Picture\_URL\_\_c" attribute of the record to the corresponding format

/resource/1563540396000/ProductImages/whitesedan.png

//no need of this step because we are directly giving them the json files to download

Replace the image name to the respective ones.

//Replace ebikeslogo.png with new ebikeslogo.png file in ebikes-lwc-master\force-app\main\default\staticresources\IccEbikes

Step 1.6.3: Replace logo.svg file in the following location with the new logo.svg file

ebikes-lwc-master\force-app\main\default\staticresources\bike\_assets

//Logos and car images are provided separately.

Step 1.7: Upload JSON files

//Replace Json Files

//Json files are provided separately

//Step 1.8: Create scratch org, Push code, load sample data and open scratch org.

Step 1.7: UI modifications //You can make this as Step2

Step 1.7.1: Product filter:

Step 1.7.1.1: Open force-app>main>default>lwc>productFilter>productFilter.html replace exiting code with the code given below:

//Provide the code in separate text files and ask them to replace code.

Codes to replace in productFilter:

1)productFilter.html

2)productFilter.js

<template>

<lightning-card title="Filters" icon-name="standard:calibration">

<div class="slds-m-horizontal\_medium">

<lightning-input

label="Search Key"

type="search"

onchange={handleSearchKeyChange}

></lightning-input>

<section>

<lightning-slider

label="Max Price"

step="100"

min="0"

max="100000"

value={maxPrice}

onchange={handleMaxPriceChange}

></lightning-slider>

</section>

<!--<section>

<h1>Fuel Type</h1>

<template if:true={categories.data}>

<template

for:each={categories.data.values}

for:item="category"

>

<lightning-input

key={category.value}

label={category.label}

data-filter="categories"

data-value={category.value}

type="checkbox"

checked

onchange={handleCheckboxChange}

></lightning-input>

</template>

</template>

<template if:true={categories.error}>

<c-inline-message

message="Error loading categories"

errors={categories.error}

></c-inline-message>

</template>

</section> -->

<!--<section>

<h1>Material</h1>

<template if:true={materials.data}>

<template

for:each={materials.data.values}

for:item="material"

>

<lightning-input

key={material.value}

label={material.label}

data-filter="materials"

data-value={material.value}

type="checkbox"

checked

onchange={handleCheckboxChange}

></lightning-input>

</template>

</template>

<template if:true={materials.error}>

<c-inline-message

message="Error loading materials"

errors={materials.error}

></c-inline-message>

</template>

</section> -->

<!--<section>

<h1>Number of Gears</h1>

<template if:true={levels.data}>

<template for:each={levels.data.values} for:item="level">

<lightning-input

key={level.value}

label={level.label}

data-filter="levels"

data-value={level.value}

type="checkbox"

checked

onchange={handleCheckboxChange}

></lightning-input>

</template>

</template>

<template if:true={levels.error}>

<c-inline-message

message="Error loading levels"

errors={levels.error}

></c-inline-message>

</template>

</section> -->

</div>

</lightning-card>

</template>

Step 1.7.2: Product Tile:

Codes to replace in ProductTile:

1)productTile.css

Step 1.7.3: Product Card:

Codes to replace in ProductCard:

1)productCard.html

Open force-app>main>default>lwc>productCard>productCard.html replace exiting code with the code given below:

<template>

<lightning-card icon-name="standard:apex\_plugin">

<template if:true={product.data}>

<span slot="title">{product.data.fields.Name.value}</span>

<lightning-button-icon

icon-name="utility:expand\_alt"

slot="actions"

onclick={handleNavigateToRecord}

></lightning-button-icon>

<div class="slds-m-horizontal\_medium">

<!--src={product.data.fields.Picture\_URL\_\_c.value} -->

<!-- src="https://images.pexels.com/photos/67636/rose-blue-flower-rose-blooms-67636.jpeg?auto=compress&cs=tinysrgb&dpr=1&w=500" -->

<img

src={product.data.fields.Picture\_URL\_\_c.value}

class="product"

alt="Product picture"

/>

<div class="row">

<div>Fuel Type:</div>

<div>{product.data.fields.Category\_\_c.value}</div>

</div>

<div class="row">

<div>Number of Gears:</div>

<div>{product.data.fields.Level\_\_c.value}</div>

</div>

<div class="row">

<div>MSRP:</div>

<div>

<lightning-formatted-number

format-style="currency"

currency-code="USD"

value={product.data.fields.MSRP\_\_c.value}

class="price"

></lightning-formatted-number>

</div>

</div>

<section>Engine Specifications</section>

<div class="row">

<div>BHP:</div>

<div>500</div>

</div>

<div class="row">

<div>Mileage:</div>

<div>30 miles</div>

</div>

<div class="row">

<div>Cooling System:</div>

<div>Turbo</div>

</div>

<section>Fuel</section>

<div class="row">

<div>Type:</div>

<div>{product.data.fields.Category\_\_c.value}</div>

</div>

<div class="row">

<div>Capacity:</div>

<div>Air spring, adjustable rebound</div>

</div>

<section>Brakes</section>

<div class="row">

<div>Front:</div>

<div>Hydraulic disc, 200mm</div>

</div>

<div class="row">

<div>Rear:</div>

<div>Hydraulic disc, 200mm</div>

</div>

</div>

</template>

<template if:true={noData}>

<c-placeholder

message="Select a product to see details"

></c-placeholder>

</template>

<template if:true={product.error}>

<c-inline-message errors={product.error}></c-inline-message>

</template>

</lightning-card>

</template>

Step 1.7.4: Product Tile List: //change it to Product List Item

Codes to replace in Product List Item:

1)productListItem.html

Step 1.7.5: Reseller Order:

To see changes in Reseller order we need to make changes in orderBuilder and orderItemTile.

Codes to replace:

1)orderBuilder.js

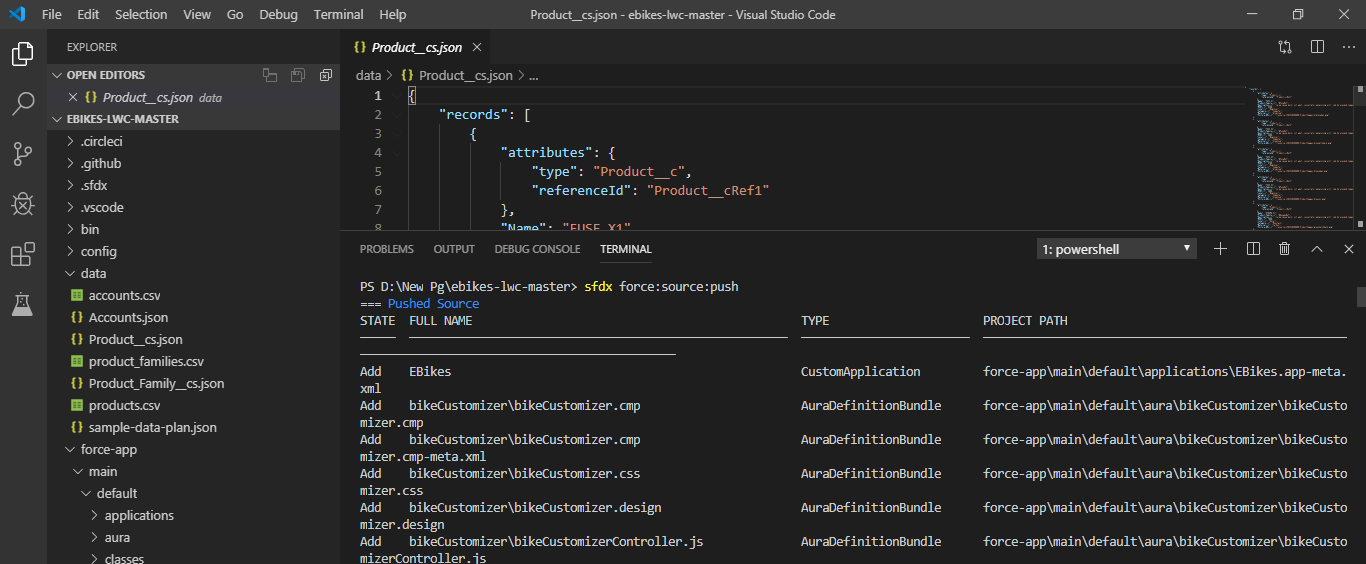
2)orderItemTile.html

Step 1.8: Push changes to the scratch org

//I think instead of making all the UI changes at once and pushing them, they can make changes, push and open for each component individually so that they can notice the changes of UI.

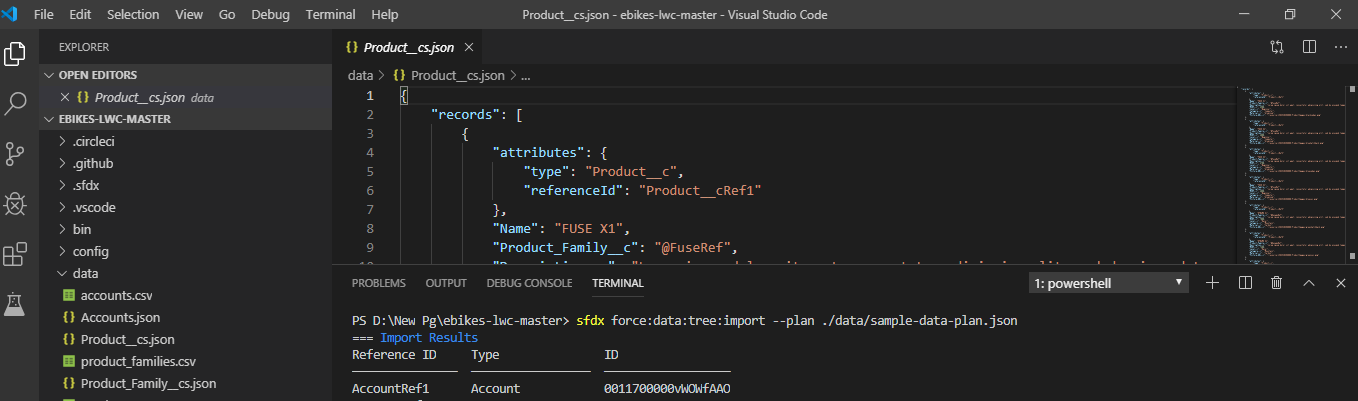
Step 1.4.1: Push the app to your scratch org:

sfdxforce:source:push



Step 1.4.3: Load sample data:

sfdxforce:data:tree:import --plan ./data/sample-data-plan.json



Step 1.4.4: Open the scratch org:

sfdxforce:org:open

