**Shopify-GitHub Integration & TailwindCSS + Js Bundle with Webpack in Shopify**

* **Get Theme files into the local machine**

To get these files into the local machine we need to install Shopify in the local machine.

Just go through with these [mentioned](https://shopify.dev/docs/themes/tools/cli/install) commands and steps in the local machine.

Now, log in with your shopify account. Follow the steps below.

**# Shopify CLI commands for themes**

- local folder in add shopify theme

**1)**  **shopify theme init**

* Clones a Git repository to your local machine as the starting point for building a theme.
* Automatically get git’s code

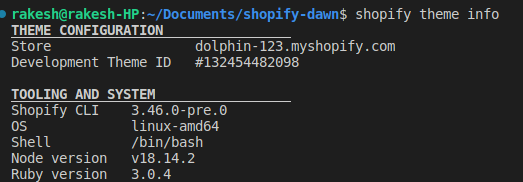
# Connecting to a store- specify the store that you want

**Note:-** Delete the git code in the local file (this is automatically generated when we init the shopify theme)

**2)** **shopify theme dev --store dolphin-123 (**go with your store name**)**

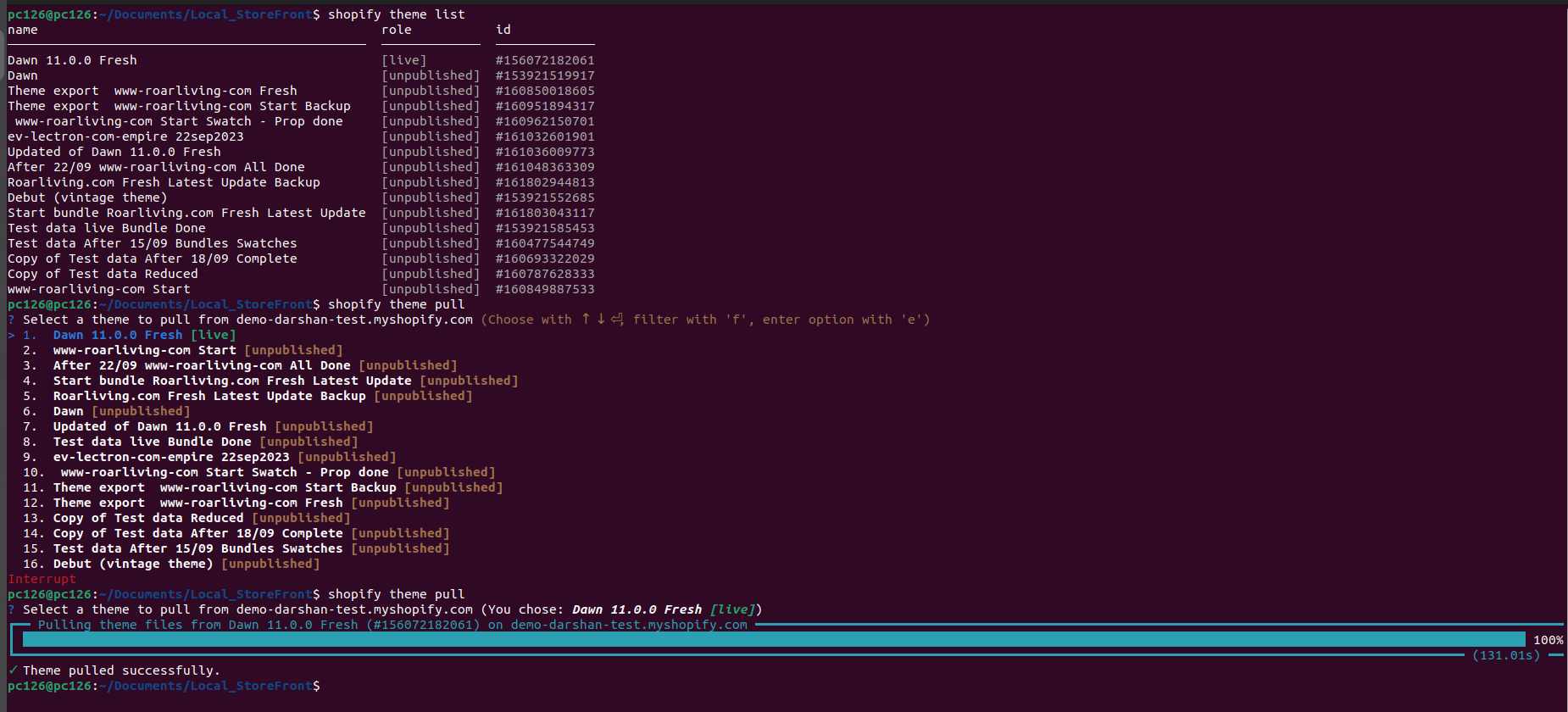
**3)** **shopify theme info**

**output:-**



**4) shopify theme list -** Gives a list of themes that are in the store.

**5) shopify theme pull -** Pull the theme files of the selected theme.

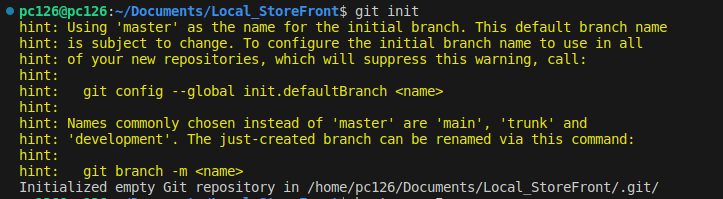
**output:- **

Now, we have all the theme files on our local machine. In the next steps, we are working with Git and shopify integration to push our code of theme files and connect the GitHub repository with our shopify store.

* **GitHub integration with Shopify store**
* First, you need a GitHub account as well as a Shopify store.
* Install [GIT](https://github.com/git-guides/install-git) in your local with commands in Terminal or VScode (as per preference).
* After GIT installation, go with your account of the GitHub. Make a repository in GitHub.
* Now move to the local machine terminal or any code editor and initialize GIT with

**git init** command.

By applying the above command you have a git in the local machine. So that you can work with git through the local.

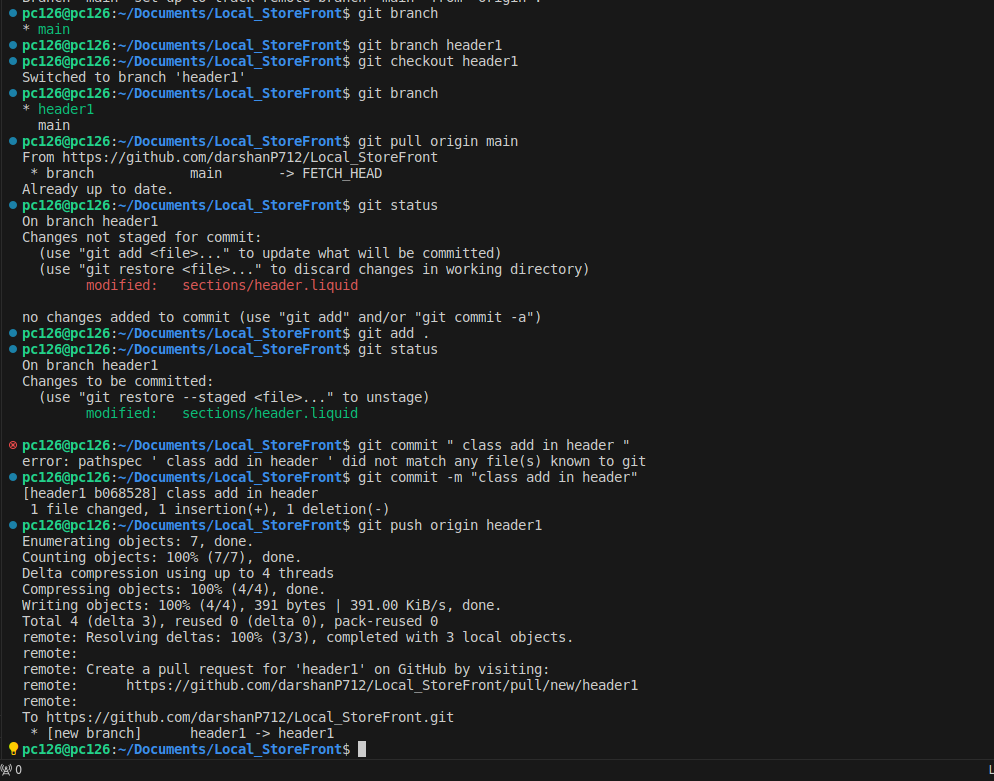


After initialization, Add a Shopify theme local with cmd or VScode.

**Now, check a branch.**

* So that you will know that you are in the main branch or the sub-branch. To check the branch run the below command.

Ex:- **git branch**

****

From the above output, we can see that we are in the main branch.

* Now make a branch through the below command and get to that new branch for our work. (git branch <branch name>)

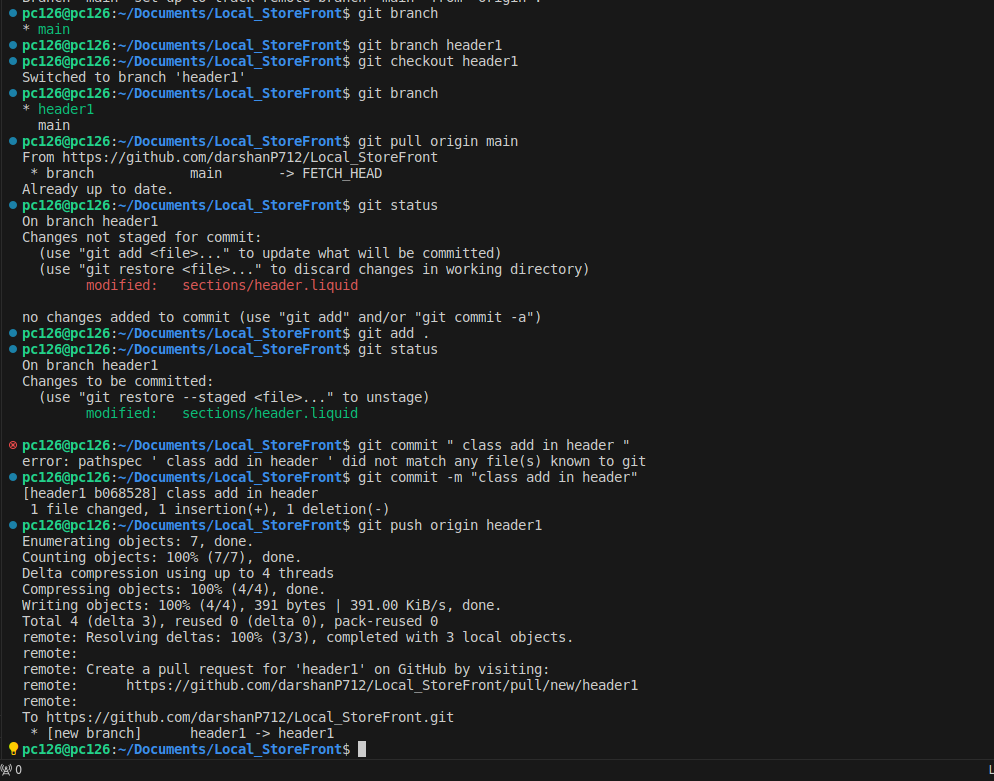
Ex:- **git branch header1**

* To get into that new branch run the following command.

(git checkout <branch name>)

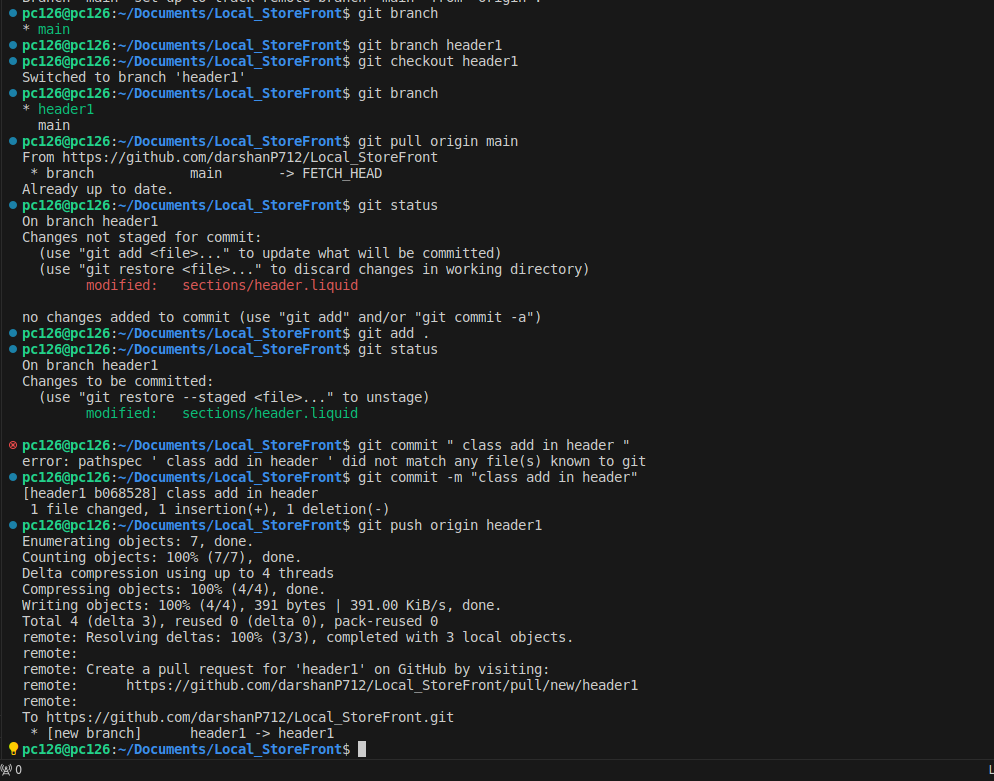
Ex:- **git checkout header1**

Now, we are at our new branch.



From the above output, We can understand how to make a branch and get into that new branch and also list all branches.

* Now, Let’s move to our shopify theme files and check the output of those changes with commands. For example, we have made a change to the header.liquid file. So check those changes and add them to the branch just go through the below steps.



From the above output, We can see how the whole flow of these steps is working with git and shopify files changes with commands.

* If our code or files are already in the main branch then check the branch with

**git branch** command and pull the code or files into the new branch to make changes

Pull from the main branch:- **git pull origin main**

* Now, make a change in any file (in the above output changes are made in “**header.liquid**” file). To check whether changes are made or not, run the below command,

Ex:- **git status**

It gives information about in which file we are making changes.

* To add all the changes in the git run the following command.

Ex:- **git add .** or **git add sections/header.liquid**

* Using ‘ **git add . ’** we are going to all the files whichever we change or not, and add all the files to commit changes**.**
* Using **‘ git add sections/header.liquid ‘ (**git add <file name/path>**)** we add only a particular file that we want to add for the commit changes.
* Now, we have to commit these changes to our git repository. Run the following command to commit the changes. Please add a commit with the message. The message is nothing but a comment(information) that informs what and for which purpose you have to make a change and commit it.

Ex:- **git commit -m “class add in header”**

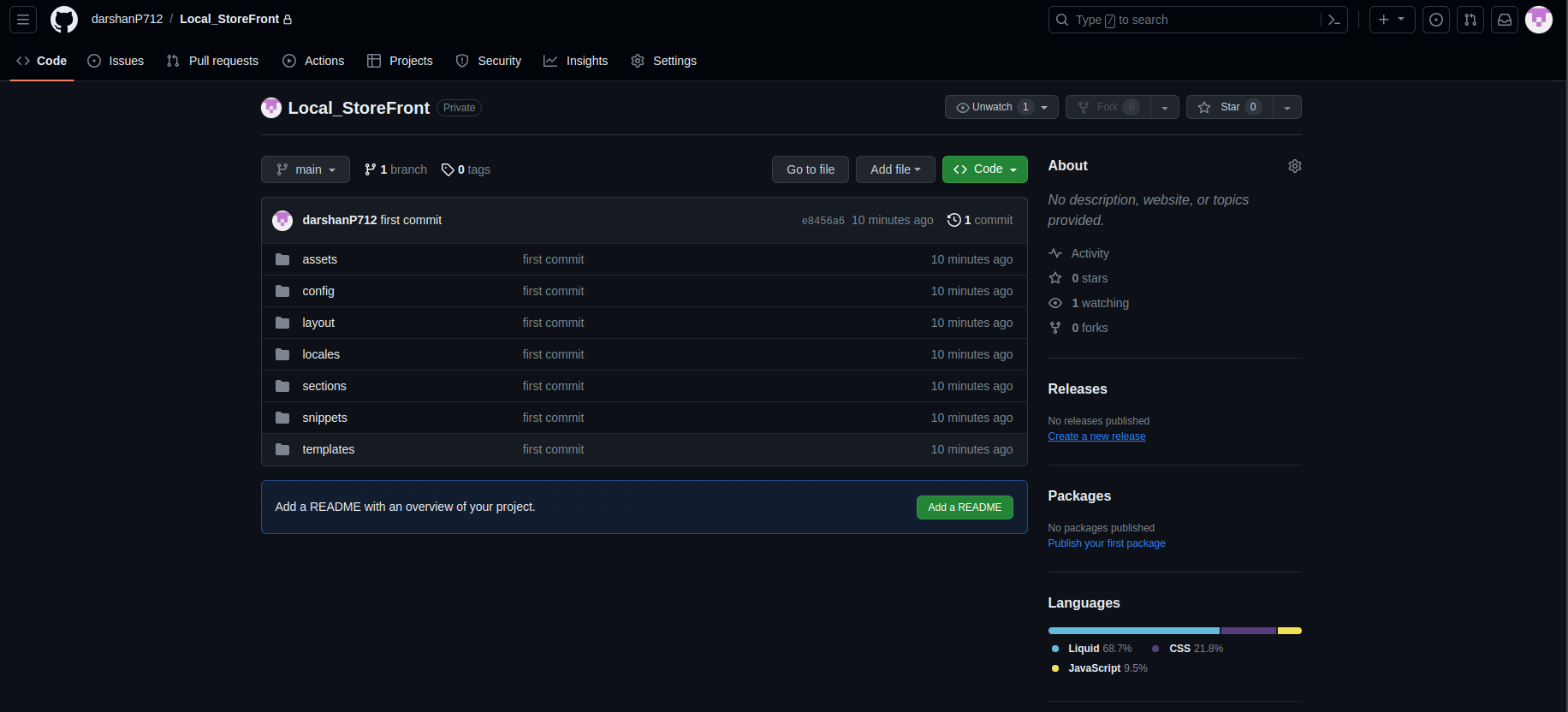
In the above example, the message shows that we add a class to the header.

* After making all the changes and committing all the changes we have to push these changes into our GitHub repository’s branch. To push these changes run the following commands.

Ex:- **git push origin header1**

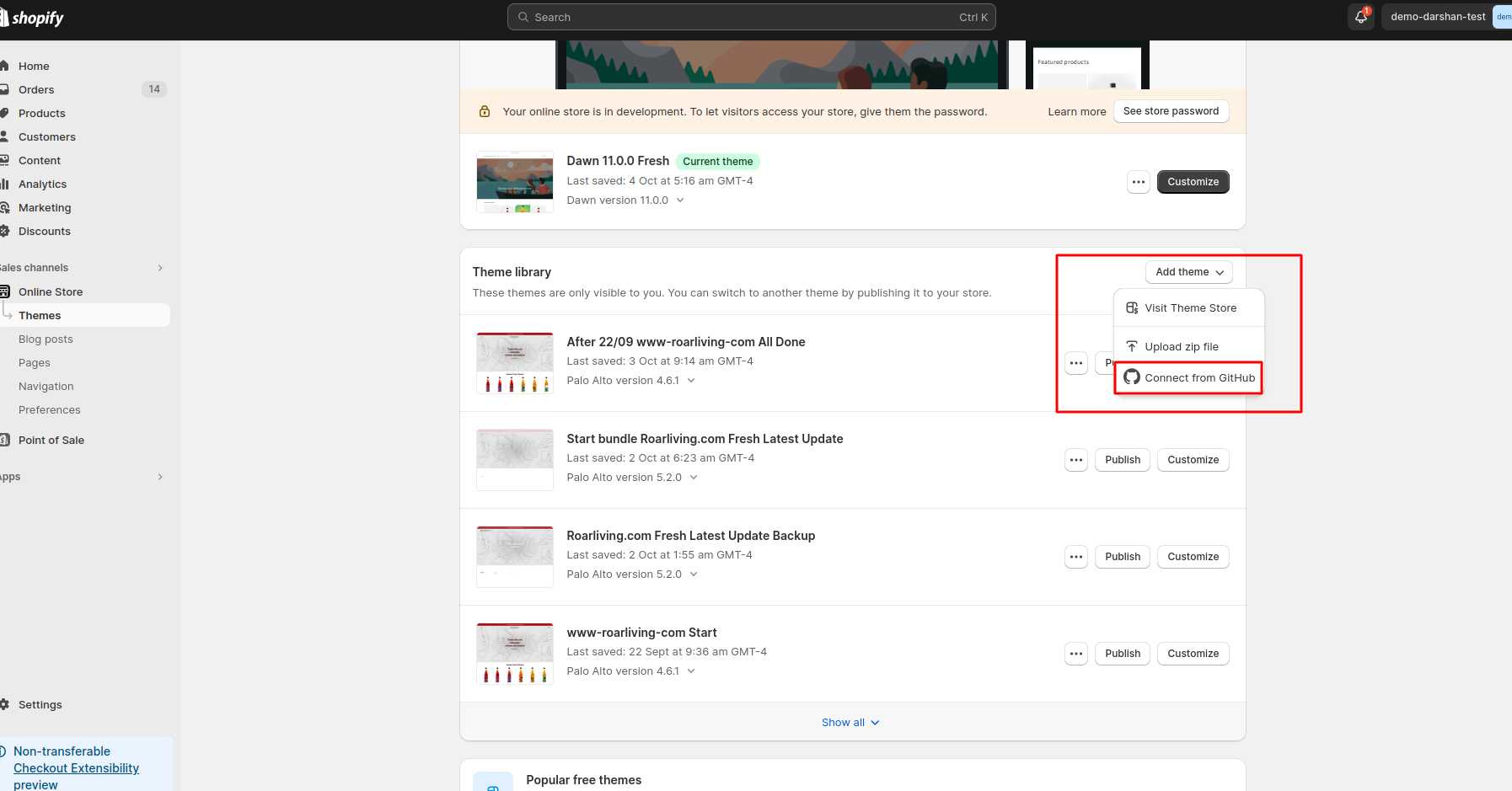
(git push origin <branch name>)

Now check your GitHub account’s repository and branch whichever we made a changes that all are displaying at the GitHub with our commented message of the commit. See the below image for a better understanding.



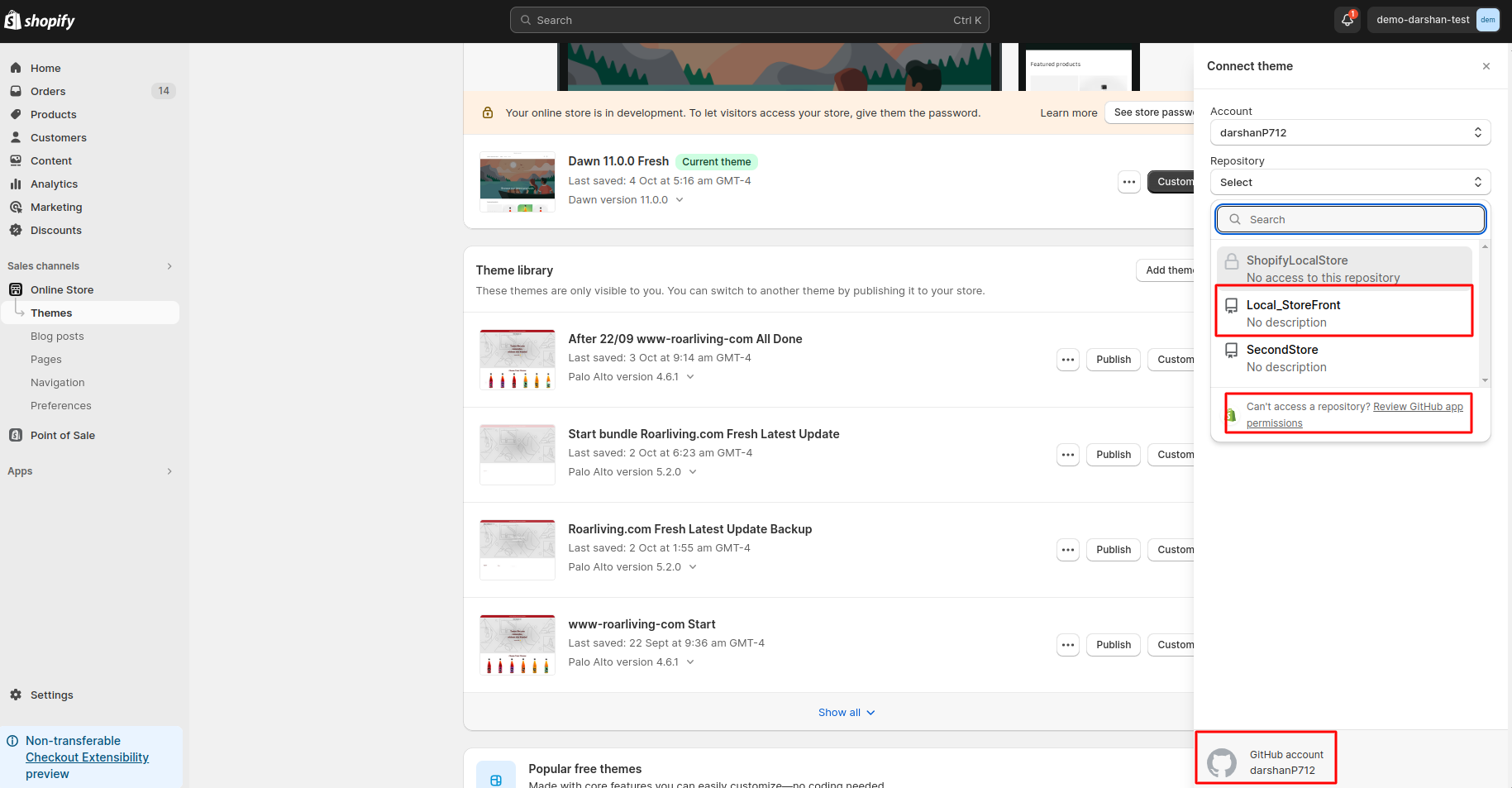
* **Now we are going to connect the GitHub repository with the Shopify Store from the Shopify admin.**

1. **Connect GitHub from shopify admin.**



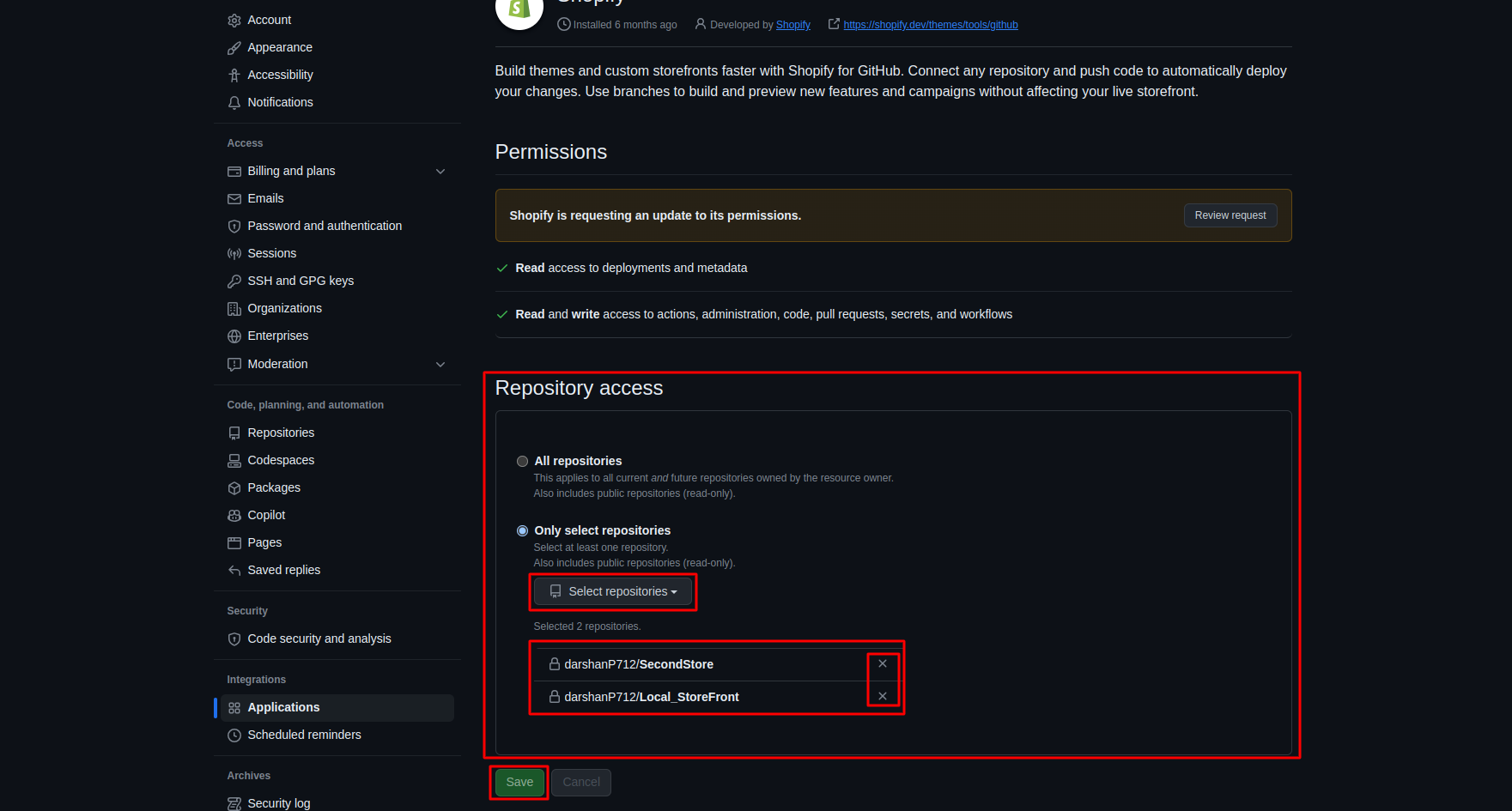
See the above image, From the shopify admin, click the **Add theme** option and then click **Connect from GitHub**.

1. **Add a repository whichever you want to add or the repository that is worked us from the above steps.**

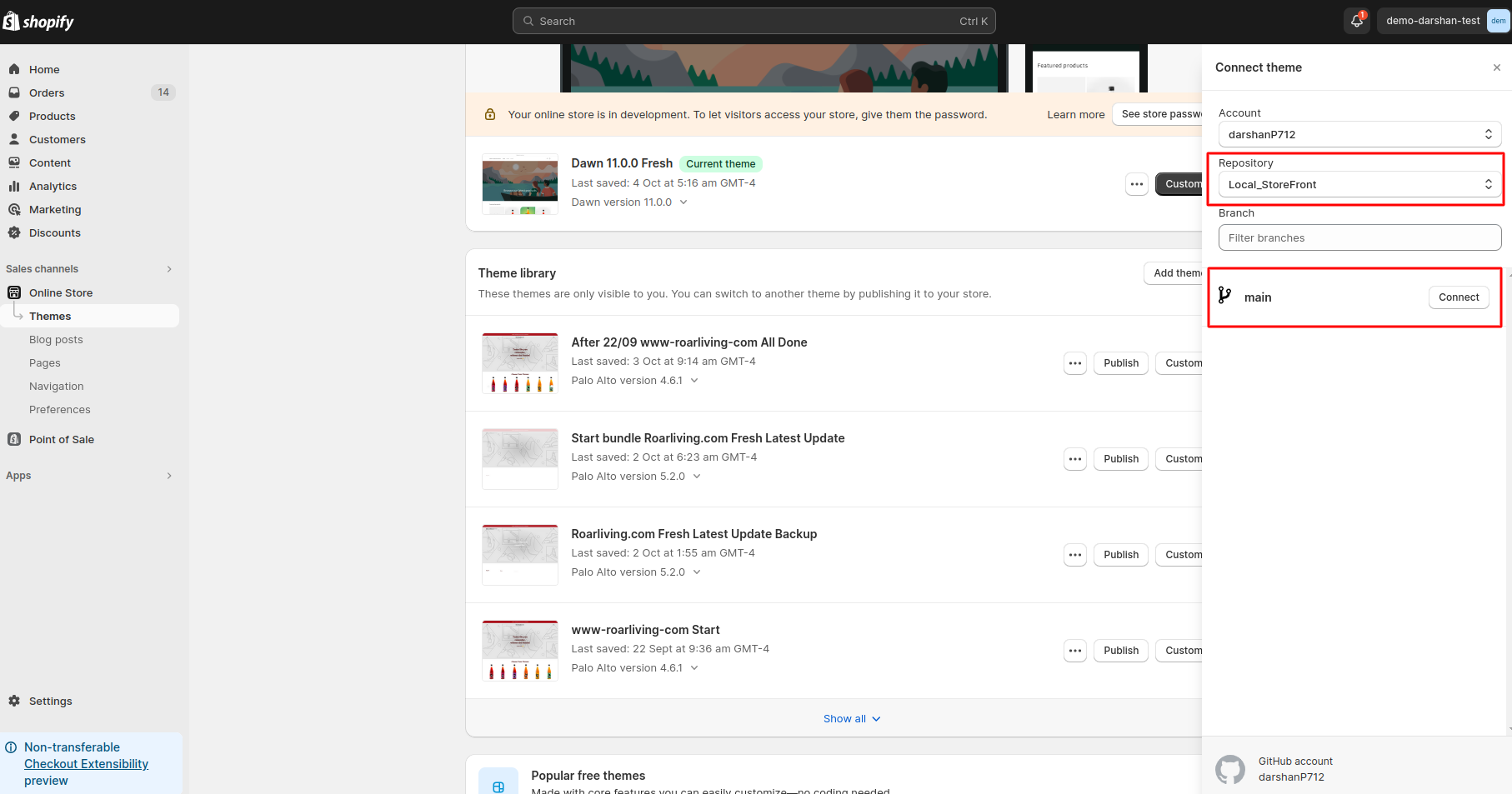


After clicking **Connect from GitHub,** We have output like the above image and then add a repository we want to add. If the GitHub’s repository is not accessed from the Shopify admin then give access to the repository.

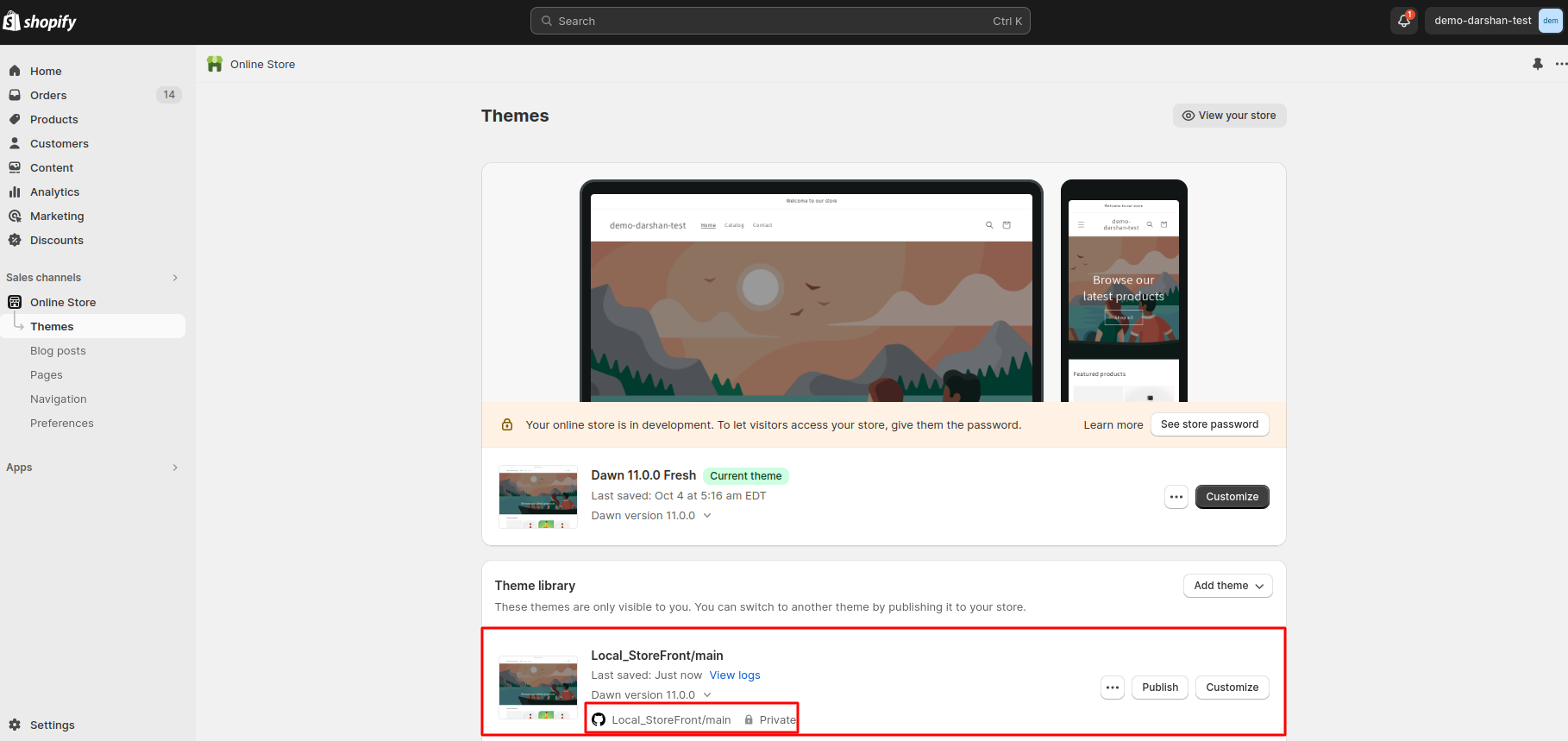
1. **Give access to the shopify store to add the theme from our GitHub repository.**



1. **Add Repository and connect the branch whichever we want to add to the shopify theme.**

****

1. **By adding the repository and its particular branch we can see the new theme which is committed and pushed by us in the GitHub with local commands. See the below image.**



We have successfully integrated Shopify with GitHub. So that whenever we change the local files to commit & push it in the git branch then its output is reflected in the Shopify theme which is added by us in our store.

* **Add** [**Webpack**](https://webpack.js.org/concepts/) **to our theme file which is in the local**

**Webpack is used for Minifying your JS and CSS in a bundle file**

Step 1: **npm init -y**

o/p:- package.json file create

Step 2: webpack command interface

**npm install webpack webpack-cli --save-dev**  // webpack-cli install

Or

**npm i -D webpack**

**npm i -D webpack-cli**

Or

**npm i -D webpack webpack-cli**

Add package.json in two changes

"scripts": {

"build": "webpack"

}

**npm install terser-webpack-plugin --save-dev** // module install

- node\_modules folder auto-create file webpack

- Like other modules same as install

**Run cmd use in module**

npm install --save-dev mini-css-extract-plugin

npm install --save-dev optimize-css-assets-webpack-plugin

npm install --save-dev terser-webpack-plugin

npm install --save-dev path

npm install --save-dev babel-loader

npm install --save-dev sass-loader

npm install --save-dev css-loader

npm install --save-dev webpack

npm install --save-dev webpack-dev-server

npm install --save-dev optimize-css-assets-webpack-plugin

npm install --save-dev laravel-mix

npm install --save-dev style-loader

npm install --save-dev clean-webpack-plugin

npm install --save-dev html-webpack-plugin

npm install --save-dev rimraf

npm install --save-dev sass

npm install --save-dev webpack-dev-server

npm install --save-dev webpack-merge

**Dependencies if used in the project all commands run**

npm install --save-dev @babel/core

npm install --save-dev @babel/preset-env

npm install --save-dev @types/clean-webpack-plugin

npm install --save-dev @types/html-webpack-plugin

npm install --save-dev @types/mini-css-extract-plugin

npm install --save-dev @types/node-sass

npm install --save-dev @types/optimize-css-assets-webpack-plugin

npm install --save-dev @types/sass-loader

npm install --save-dev @types/terser-webpack-plugin

npm install --save-dev @types/webpack

npm install --save-dev @types/webpack-dev-server

npm install --save-dev @types/webpack-merge

sudo chown -R $USER /usr/local/lib/node\_modules

npm install uglifyjs-webpack-plugin

**Error**:-

npm ERR! code ERESOLVE

npm ERR! ERESOLVE unable to resolve the dependency tree

**Solutions**: -

npm config set legacy-peer-deps true

**Error:-**

WARNING in bundle.js contains invalid source map

**Solutions**

Add code in file name webpack.config.js

devtool: 'source-map',

Or

devtool: 'eval-source-map',

**create a new file and add the below content in that file:-** webpack.config.js

const path = require("path");

const HtmlWebpackPlugin = require("html-webpack-plugin");

module.exports = {

entry: "./src/js/app.js",

module: {

rules: [

{

test: /\.svg$/,

use: "svg-inline-loader",

},

{

test: /\.css$/i,

use: ["style-loader", "css-loader"],

},

{

test: /\.(js)$/,

use: "babel-loader",

},

],

},

resolve: {

extensions: ['.js', '.jsx', '.ts', '.tsx', '.json', '.css', '.scss'],

modules: ['src', 'node\_modules']

},

output: {

path: path.resolve(\_\_dirname, "./assets"),

filename: "bundle.js",

},

plugins: [new HtmlWebpackPlugin()],

mode: process.env.NODE\_ENV === "production" ? "production" : "development",

};

Step 3:

npx webpack

or

npm run

npm run build

npm start

o/p

> tiedroots-new@1.0.0 build

> NODE\_ENV='production' webpack

asset bundle.js 941 bytes [emitted] [minimized] (name: main)

asset index.html 216 bytes [emitted]

./assets/custom.js 1.85 KiB [built] [code generated]

webpack 5.88.1 compiled successfully in 615 ms

ERROR

1) ERROR in main

Module not found: Error: Can't resolve './src'

or

The field 'browser' doesn't contain a valid alias configuration

Furthermore, in red, it'll log my file directory with /index doesn't exist (.js / .json / .wasm).

Solution:- webpack.config.js file in add

resolve: {

extensions: ['.js', '.jsx', '.ts', '.tsx', '.json', '.css', '.scss'],

modules: ['src', 'node\_modules'] // Assuming that your files are inside the src dir

},

**All list extensions for example**

resolve: {

extensions: [nn

'.js','.jsx','.ts','.tsx','.json','.css','.scss','.html','.htm','.xml','.svg','.jpg','.jpeg','.png','.gif','.bmp','.ico','.webp','.woff','.woff2','.ttf','.eot',

'.otf','.csv','.xls','.xlsx','.txt','.md','.pdf','.doc','.docx','.ppt','.pptx',

],

modules: ['src', 'node\_modules'], // Assuming that your files are inside the src dir

},

2) npm ERR! could not determine the executable to run

solution :- **npm install @capacitor/cli@latest @capacitor/core@latest**

* **To minify the javascript you must add some plugins which are mentioned below. Add any one of the following.**
* [**UglifyJsPlugin**](https://www.npmjs.com/package/uglifyjs-webpack-plugin)
* [**TerserPlugin**](https://webpack.js.org/plugins/terser-webpack-plugin/)
* [**Babel minify webpack plugin**](https://www.npmjs.com/package/babel-minify-webpack-plugin)
* **To minify the CSS you must add some plugins which are mentioned below.**

**Add any one of the following.**

* [**MiniCssExtractPlugin**](https://webpack.js.org/plugins/mini-css-extract-plugin/)
* [**CssMinimizerWebpackPlugin**](https://webpack.js.org/plugins/css-minimizer-webpack-plugin/)
* **ADD** [**TAILWIND**](https://tailwindcss.com/docs/installation) **CSS**
* **Run the below commands for initialization of the TailwindCSS**

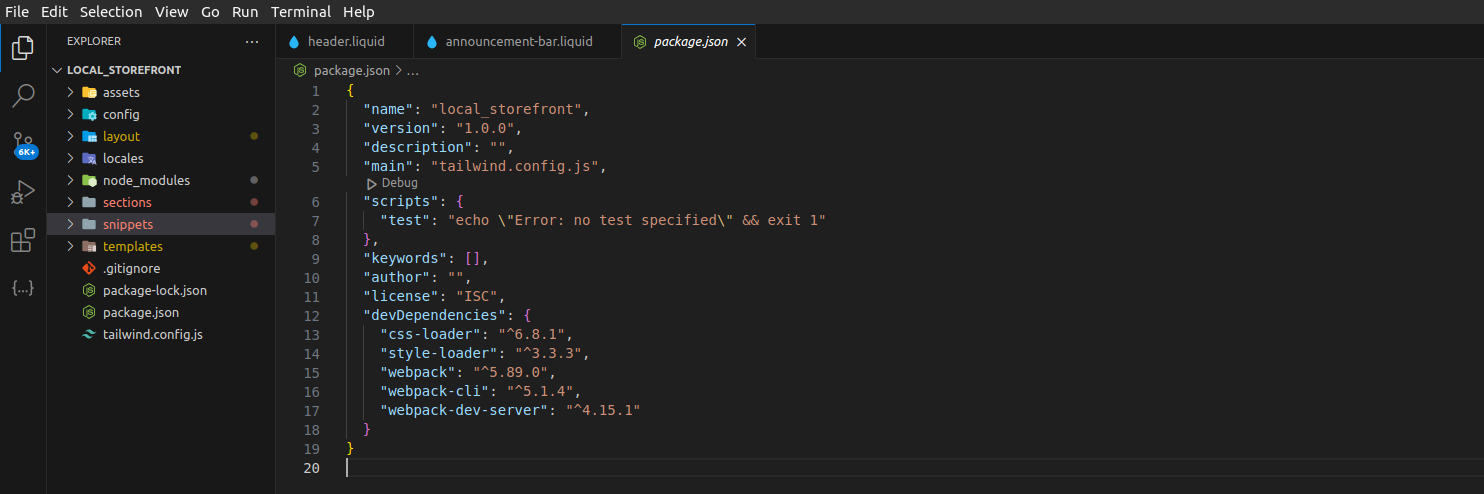
npm install -D tailwindcss

npx tailwindcss init -> it will make "tailwind.config.js"

**// LARAVEL MIX**

1. **npm init -y** :- This will make a **package.json** file.

Then run:- **npm install laravel-mix --save-dev**

****

2. Create webpack.mix.js file run following command:- **touch webpack.mix.js**

Add below content in the webpack.mix.js file,

let mix = require('laravel-mix');

mix.js('src/js/app.js', 'assets')

.css('src/css/app.css', 'assets');

* **Now, Add these two following directories with their files in the theme**

src/js/app.js ,

src/css/app.css

* **Add the following content into css/app.css**

@tailwind base;

@tailwind components;

@tailwind utilities;

* **In the "tailwind.config.js" file add this in the content,**

'./config/\*.json',

'./layout/\*.liquid',

'./assets/\*.liquid',

'./sections/\*.liquid',

'./snippets/\*.liquid',

'./templates/\*.liquid',

'./templates/\*.json',

'./templates/customers/\*.liquid'

* Now run the command "**npx mix**"

Then see the **assets/app.css** it will show the configurable classes and with its css

* **now run this command for the output of css.**

npx tailwindcss -i ./src/css/app.css -o ./assets/app.css --watch

* **add this asset file css into a liquid file wherever you want**

{{ 'app.css' | asset\_url | stylesheet\_tag }}

Then refresh the particular page that has TailwindCSS output