

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](#) 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:-

```
rakesh@rakesh-HP:~/Documents/shopify-dawn$ shopify theme info
THEME CONFIGURATION
Store                dolphin-123.myshopify.com
Development Theme ID #132454482098

TOOLING AND SYSTEM
Shopify CLI          3.46.0-pre.0
OS                   linux-amd64
Shell                /bin/bash
Node version         v18.14.2
Ruby version         3.0.4
```

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:-

```
pci126@pci126:~/Documents/Local_StoreFront$ shopify theme list
name role id
Dawn 11.0.0 Fresh [live] #156072182861
Dawn [unpublished] #153921519917
Theme export www-roarliving-com Fresh [unpublished] #160850018605
Theme export www-roarliving-com Start Backup [unpublished] #160951894317
www-roarliving-com Start Swatch - Prop done [unpublished] #160962150701
ev-lectron-con-empire 22sep2023 [unpublished] #161032601901
Updated of Dawn 11.0.0 Fresh [unpublished] #161036009773
After 22/09 www-roarliving-com All Done [unpublished] #161048363309
Roarliving.com Fresh Latest Update Backup [unpublished] #161802944813
Debut (vintage theme) [unpublished] #153921552685
Start bundle Roarliving.com Fresh Latest Update [unpublished] #161803043117
Test data live Bundle Done [unpublished] #153921585453
Test data After 15/09 Bundles Swatches [unpublished] #160477544749
Copy of Test data After 18/09 Complete [unpublished] #160693322829
Copy of Test data Reduced [unpublished] #160787628333
www-roarliving-com Start [unpublished] #160849887533
pci126@pci126:~/Documents/Local_StoreFront$ shopify theme pull
? Select a theme to pull from deno-darshan-test.nyshopify.com (Choose with ↑ ↓ ↵, filter with 'f', enter option with 'e')
1. Dawn 11.0.0 Fresh [live]
2. www-roarliving-com Start [unpublished]
3. After 22/09 www-roarliving-com All Done [unpublished]
4. Start bundle Roarliving.com Fresh Latest Update [unpublished]
5. Roarliving.com Fresh Latest Update Backup [unpublished]
6. Dawn [unpublished]
7. Updated of Dawn 11.0.0 Fresh [unpublished]
8. Test data live Bundle Done [unpublished]
9. ev-lectron-con-empire 22sep2023 [unpublished]
10. www-roarliving-com Start Swatch - Prop done [unpublished]
11. Theme export www-roarliving-com Start Backup [unpublished]
12. Theme export www-roarliving-com Fresh [unpublished]
13. Copy of Test data Reduced [unpublished]
14. Copy of Test data After 18/09 Complete [unpublished]
15. Test data After 15/09 Bundles Swatches [unpublished]
16. Debut (vintage theme) [unpublished]
Interrupt
pci126@pci126:~/Documents/Local_StoreFront$ shopify theme pull
? Select a theme to pull from deno-darshan-test.nyshopify.com (You chose: Dawn 11.0.0 Fresh [live])
Pulling theme files from Dawn 11.0.0 Fresh (#156072182861) on deno-darshan-test.nyshopify.com
100% (131.01s)
✓ Theme pulled successfully.
pci126@pci126:~/Documents/Local_StoreFront$
```

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](#) 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.

```

● pc126@pc126:~/Documents/Local_StoreFront$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:     git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:     git branch -m <name>
Initialized empty Git repository in /home/pc126/Documents/Local_StoreFront/.git/

```

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**

```

● pc126@pc126:~/Documents/Local_StoreFront$ git branch
* main

```

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.

```

● pc126@pc126:~/Documents/Local_StoreFront$ git branch
* main
● pc126@pc126:~/Documents/Local_StoreFront$ git branch header1
● pc126@pc126:~/Documents/Local_StoreFront$ git checkout header1
Switched to branch 'header1'
● pc126@pc126:~/Documents/Local_StoreFront$ git branch
* header1
  main

```

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.

```

pc126@pc126:~/Documents/Local_StoreFront$ git branch
* main
pc126@pc126:~/Documents/Local_StoreFront$ git branch header1
pc126@pc126:~/Documents/Local_StoreFront$ git checkout header1
Switched to branch 'header1'
pc126@pc126:~/Documents/Local_StoreFront$ git branch
* header1
  main
pc126@pc126:~/Documents/Local_StoreFront$ git pull origin main
From https://github.com/darshanP712/Local_StoreFront
* branch      main      -> FETCH_HEAD
Already up to date.
pc126@pc126:~/Documents/Local_StoreFront$ git status
On branch header1
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   sections/header.liquid

no changes added to commit (use "git add" and/or "git commit -a")
pc126@pc126:~/Documents/Local_StoreFront$ git add .
pc126@pc126:~/Documents/Local_StoreFront$ git status
On branch header1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   sections/header.liquid

pc126@pc126:~/Documents/Local_StoreFront$ git commit " class add in header "
error: pathspec ' class add in header ' did not match any file(s) known to git
pc126@pc126:~/Documents/Local_StoreFront$ git commit -m "class add in header"
[header1 b068528] class add in header
1 file changed, 1 insertion(+), 1 deletion(-)
pc126@pc126:~/Documents/Local_StoreFront$ git push origin header1
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 391 bytes | 391.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
remote:
remote: Create a pull request for 'header1' on GitHub by visiting:
remote:   https://github.com/darshanP712/Local_StoreFront/pull/new/header1
remote:
To https://github.com/darshanP712/Local_StoreFront.git
 * [new branch]      header1 -> header1
pc126@pc126:~/Documents/Local_StoreFront$

```

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”**

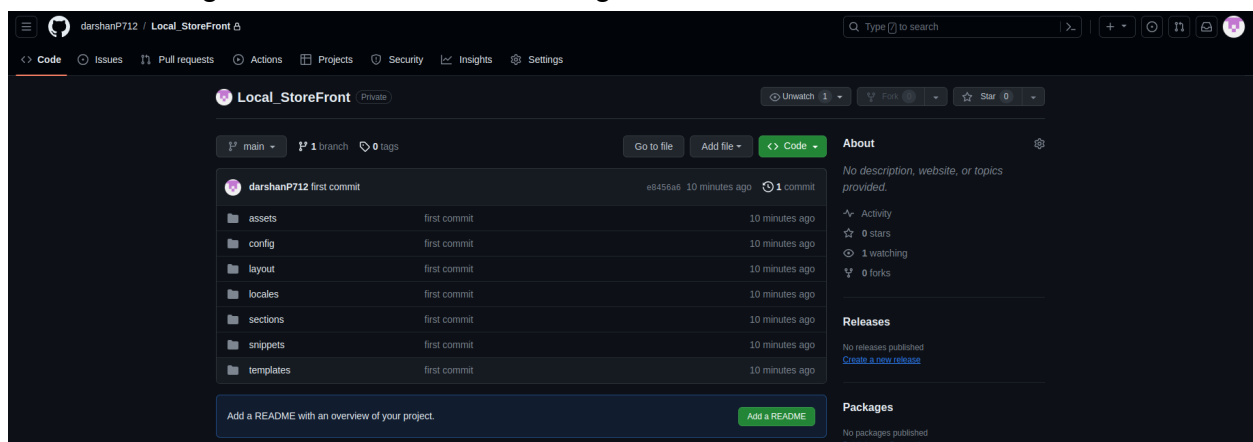
In the above example, the message shows that we add a class to the header.

- 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**

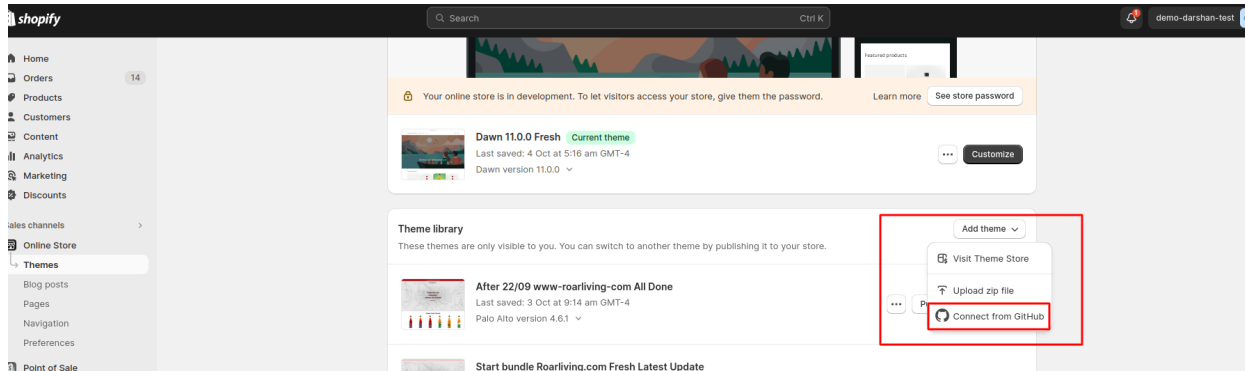
(git push origin <branch name>)

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.



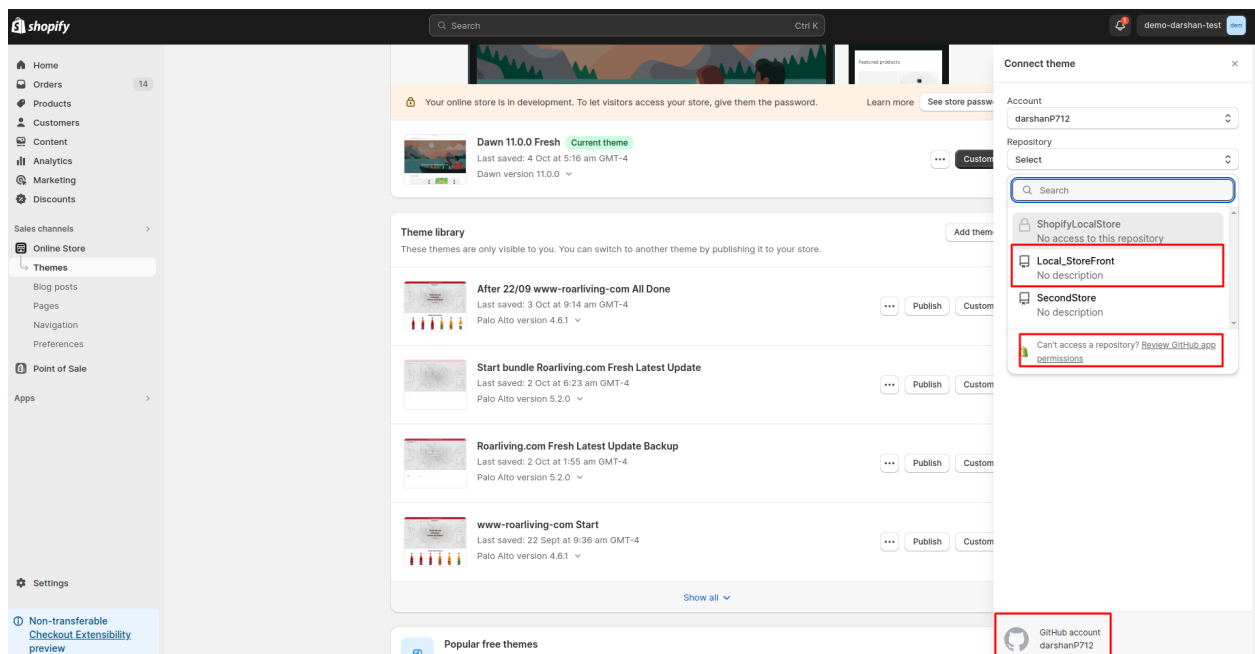
- Now we are going to connect the GitHub repository with the Shopify Store from the Shopify admin.

1. Connect GitHub from shopify admin.



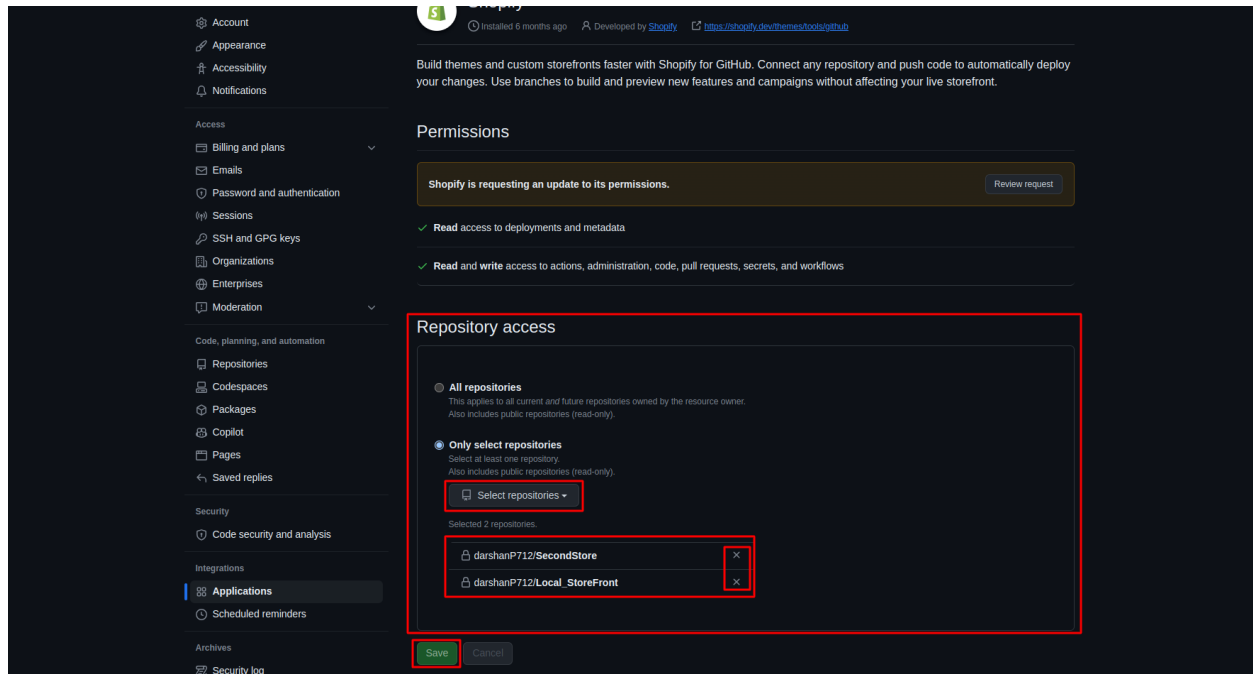
See the above image, From the shopify admin, click the **Add theme** option and then click **Connect from GitHub**.

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

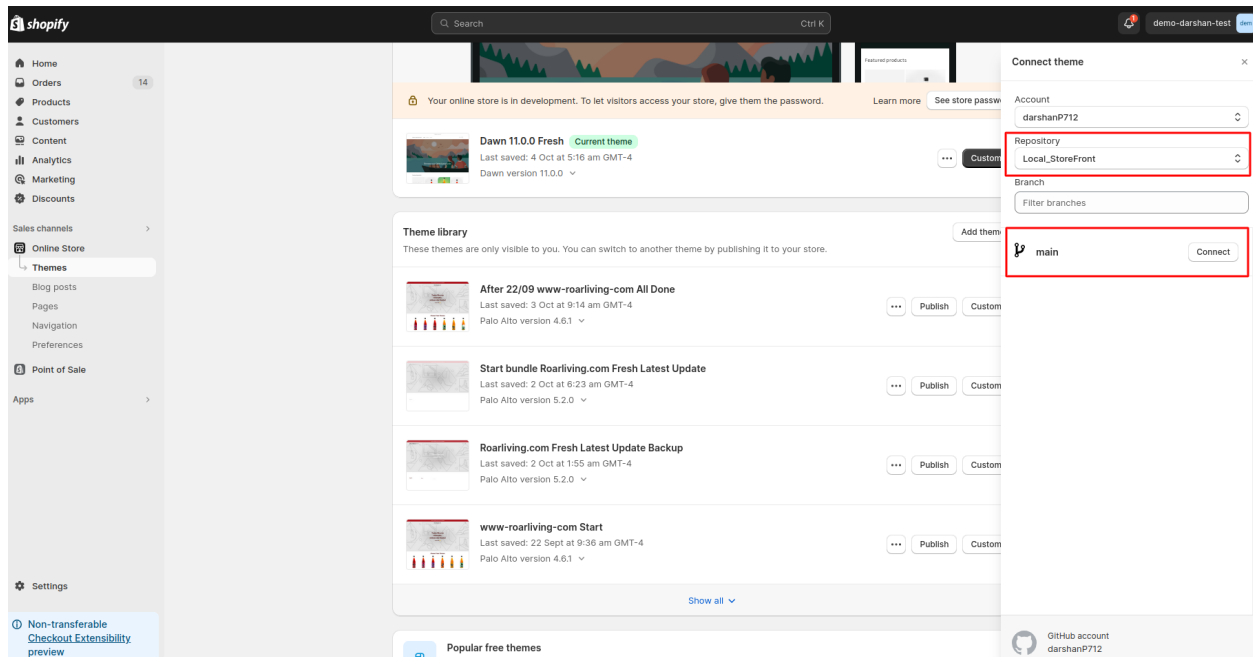


After clicking **Connect from GitHub**, We have output like the above image and then add a repository we want to add. If the GitHub's repository is not accessed from the Shopify admin then give access to the repository.

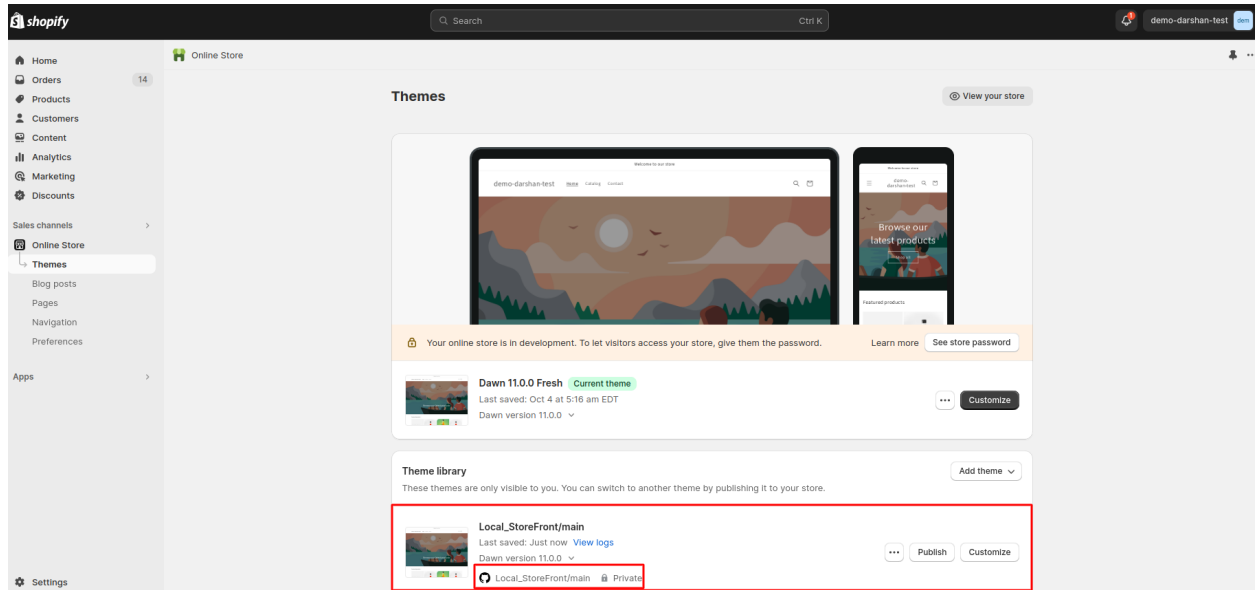
3. Give access to the shopify store to add the theme from our GitHub repository.



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



5. 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 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](#)

→ [TerserPlugin](#)

→ [Babel minify webpack plugin](#)

- To minify the CSS you must add some plugins which are mentioned below. Add any one of the following.

→ [MiniCssExtractPlugin](#)

→ [CssMinimizerWebpackPlugin](#)

★ ADD TAILWIND CSS

- Run the below commands for initialization of the TailwindCSS

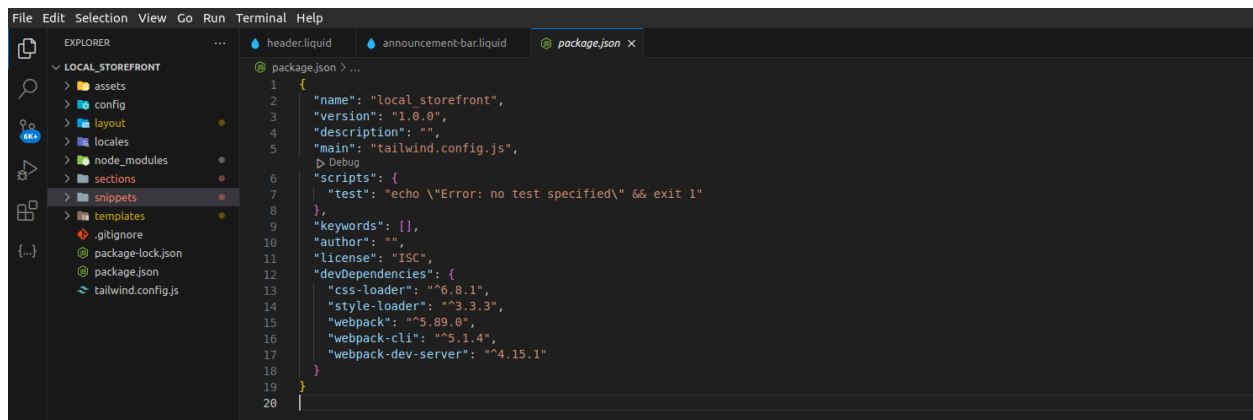
```
npm install -D tailwindcss
```

```
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**



The screenshot shows a code editor with the Explorer view on the left displaying the project structure. The main editor area shows the content of the package.json file. The file includes fields for name, version, description, main, scripts, keywords, author, license, and devDependencies. The devDependencies list includes css-loader, style-loader, webpack, webpack-cli, and webpack-dev-server.

```
1 {
2   "name": "local storefront",
3   "version": "1.0.0",
4   "description": "",
5   "main": "tailwind.config.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "keywords": [],
10  "author": "",
11  "license": "ISC",
12  "devDependencies": {
13    "css-loader": "^6.8.1",
14    "style-loader": "^3.3.3",
15    "webpack": "^5.89.0",
16    "webpack-cli": "^5.1.4",
17    "webpack-dev-server": "^4.15.1"
18  }
19 }
20
```

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