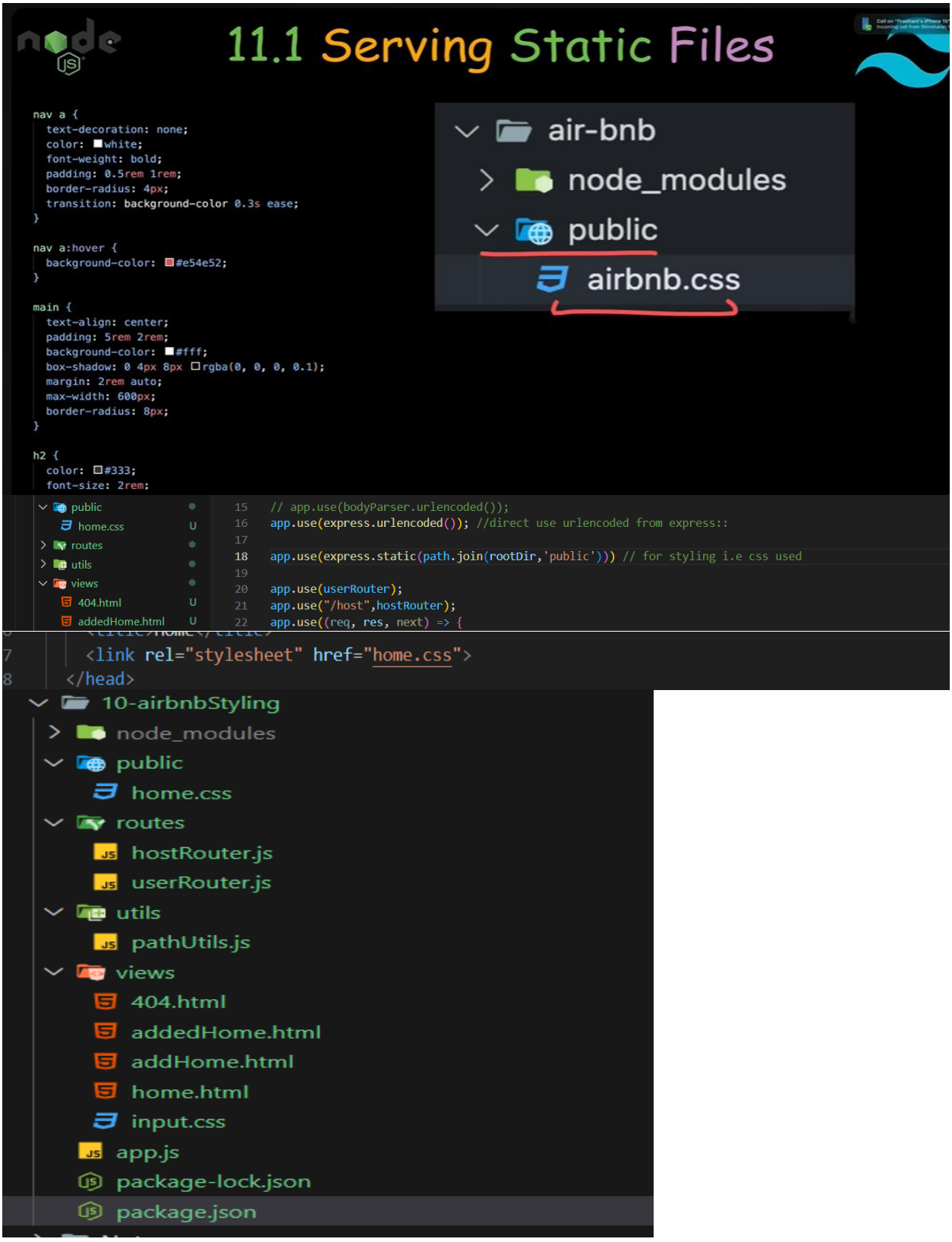
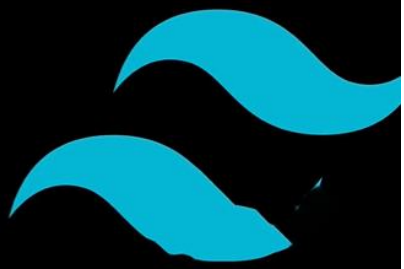
Add Css File:::


11.1 Serving Static Files
nav a {
text-decoration: none;
color: white;
font-weight: bold;
padding: 0.5rem 1rem;
border-radius: 4px;
transition: background-color 0.3s ease;
}
nav a:hover {
background-color: #e54e52;
}
main {
text-align: center;
padding: 5rem 2rem;
background-color: #fff;
box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
margin: 2rem auto;
max-width: 600px;
border-radius: 8px;
}
h2 {
color: #333;
font-size: 2rem;
air-bnb
node_modules
public
airbnb.css
public
home.css
routes
utils
views
404.html
addedHome.html
// app.use(bodyParser.urlencoded());
app.use(express.urlencoded()); //direct use urlencoded from express::
app.use(express.static(path.join(rootDir,'public'))) // for styling i.e css used
app.use(userRouter);
app.use("/host",hostRouter);
app.use((req, res, next) => {
<link rel="stylesheet" href="home.css">
</head>
10-airbnbStyling
node_modules
public
home.css
routes
hostRouter.js
userRouter.js
utils
pathUtils.js
views
404.html
addedHome.html
addHome.html
home.html
input.css
app.js
package-lock.json
package.json

11.2 Introduction to Tailwind CSS

1. Responsive: Mobile-first design for all device sizes.
2. Utility-First: Provides low-level utility classes for building custom designs.
3. Highly Customizable: Easily extendable through a config file.
4. Responsive Design: Built-in responsive utilities (e.g., sm:, md:).
5. No Predefined Components: Focuses on building custom components.
6. Purge CSS: Removes unused styles in production for smaller files.
7. Fast Development: Style elements directly in markup for speed.



11.3 Utility Classes

```css
/* Colors */
.text-primary { color: #007bff; }
.bg-primary { background-color: #007bff; }

/* Sizing */
.w-full { width: 100%; }
.h-full { height: 100%; }

/* Typography */
.text-center { text-align: center; }
.font-bold { font-weight: bold; }

/* Spacing */
.m-1 { margin: 0.25rem; }
.m-2 { margin: 0.5rem; }
.p-1 { padding: 0.25rem; }
```

```css
/* Layout */
.d-flex { display: flex; }
.flex-col { flex-direction: column; }
.items-center { align-items: center; }
.justify-center { justify-content: center; }

/* Misc */
.rounded { border-radius: 0.25rem; }
.hidden { display: none; }
```

11.4 Installing Extension

Tailwind CSS IntelliSense v0.12.10

Tailwind Labs tailwindcss.com | 7,381,739 | ★★★★★ (99)

Intelligent Tailwind CSS tooling for VS Code

Install | Auto Update

11.6 Installing Tailwind CSS

1. Install:
   npm init -y
   npm install -D tailwindcss postcss autoprefixer
2. Initialize Tailwind CSS Config
   npx tailwindcss init

11.6 Installing Tailwind CSS

Configure Tailwind in the Configuration Files (tailwind.config.js)

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: ["./views/*.html"],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

11.6 Installing Tailwind CSS

Add Tailwind Directives to views/input.css

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

5. Include public/output.css into your index.html
6. Run Command
   npx tailwindcss -i ./views/input.css -o ./public/output.css --watch
7. Declare Shortcut:

```
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "tailwind": "npx tailwindcss -i ./src/input.css -o ./src/output.css --watch
},
```

8. Run Command
   npm run tailwind

```
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "tailwind": "tailwindcss -i ./views/input.css -o ./public/output.
  css --watch",
  "start": "nodemon app.js & npm run tailwind"
}
```

##Since you are working on a **Node.js & Express** project and want to integrate **Tailwind CSS**, follow these steps carefully.

Step 1: Install Tailwind CSS

In your project root (10-airbnbStyling), open the terminal and run:

npm install -D tailwindcss postcss autoprefixer

This installs Tailwind CSS along with PostCSS and Autoprefixer, which help with processing styles.

Step 2: Initialize Tailwind CSS

Once installed, generate the Tailwind configuration file by running:

npx tailwindcss init -p

This will create:

- tailwind.config.js (Tailwind configuration)
- postcss.config.js (PostCSS configuration)

Step 3: Configure Tailwind

Open **tailwind.config.js** and update the **content** section to include your HTML and template files:

```
/** @type {import('tailwindcss').Config} */
module.exports = {
 content: ["./views/**/*.html", "./public/**/*.css"],
 theme: {
  extend: {},
 },
 plugins: [],
};
```

This tells Tailwind to scan all your .html files inside views and .css files inside public for class names.

Step 4: Create Tailwind CSS Entry File

Inside the public folder, create a new CSS file, e.g., tailwind.css, and add:

@tailwind base;

@tailwind components;

@tailwind utilities;

Step 5: Build Tailwind CSS

Run the following command to generate the final CSS:

npx tailwindcss -i ./view/input.css -o ./public/output.css --watch

- -i (input) → Uses tailwind.css
- -o (output) → Generates output.css inside public
- --watch → Watches for changes and updates automatically

Step 6: Link Tailwind CSS in Your HTML

In your views/home.html (or other HTML files), link the generated CSS file:

<link href="/public/output.css" rel="stylesheet">

Step 7: Test Tailwind CSS

Add this inside home.html to check if Tailwind is working:

<h1 class="text-3xl font-bold text-blue-600">Tailwind is Working!</h1>

Start your Node.js server and open the page in a browser to see if Tailwind applies styles.

Final Notes

- Keep the npx tailwindcss -i ./public/tailwind.css -o ./public/output.css --watch command running while developing.
- If npx tailwindcss init -p fails, manually create tailwind.config.js and postcss.config.js.

Let me know if you run into issues! 🚀

If step2 fail.It looks like npx is failing to find the Tailwind executable. Let's go step by step to fix this issue.

✅ **Step 2: Manually Create Tailwind Config**

If npx tailwindcss init -p is still failing, manually create the configuration files.

1 Create tailwind.config.js

Inside your project root (10-airbnbStyling), create a new file tailwind.config.js and paste:

```
/** @type {import('tailwindcss').Config} */
module.exports = {
 content: ["./views/**/*.html", "./public/**/*.css"],
 theme: {
  extend: {},
 },
 plugins: [],
};
```

2 Create postcss.config.js

Create another file postcss.config.js and paste:

```
module.exports = {
 plugins: {
  tailwindcss: {},
  autoprefixer: {},
 },
};
```