Aim: To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.

Theory:

PWA stands for Progressive Web App. It's a type of application software delivered through the web, built using common web technologies such as HTML, CSS, and JavaScript. PWAs are intended to work on any platform that uses a standards-compliant browser. They have several characteristics that distinguish them from traditional web applications:

Features:

1. **Progressive Enhancement :** PWAs are designed to work for every user, regardless of the browser they are using, by employing progressive enhancement principles.

2. **Responsive:** They are responsive and adapt to any device size, whether it's a desktop, tablet, or mobile phone.

3. **Connectivity Independent:** PWAs can work even when the device is offline or has a poor internet connection, thanks to service workers and caching strategies.

4. **App-like Interface:** PWAs offer an app-like experience with features such as push notifications and full-screen mode.

5. **Discoverable:** PWAs are discoverable by search engines and can be indexed like traditional websites.

PWA vs Traditional web app:
Progressive Web Apps (PWAs) differ from traditional native apps in their platform independence, installation process, distribution model, and update mechanism. Unlike native apps, PWAs can run on any device with a modern web browser and are installed directly from the browser without the need for app stores. They are distributed through URLs and receive automatic updates, ensuring users always have the latest version. PWAs also offer offline functionality and leverage web storage mechanisms. While they may not have access to the full range of native device capabilities, PWAs provide a cost-effective and efficient way to deliver app-like experiences across different platforms and devices directly from the web.

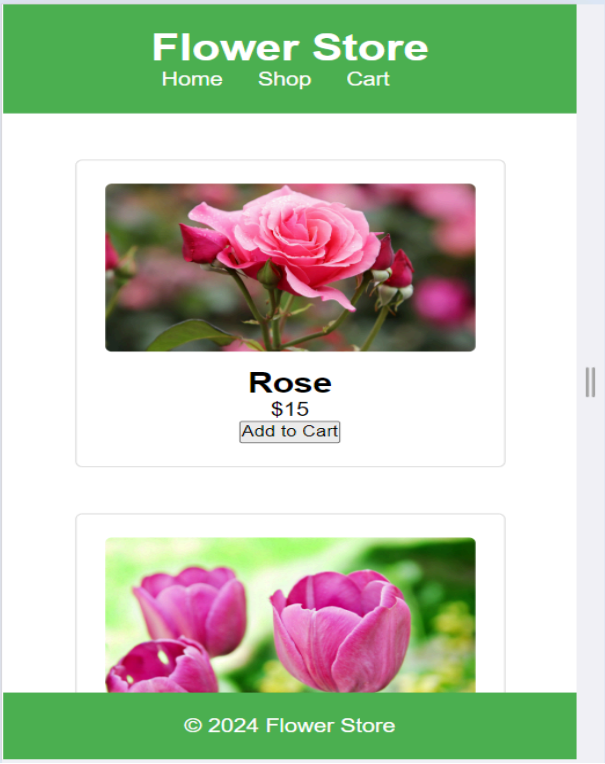
OUTPUT:

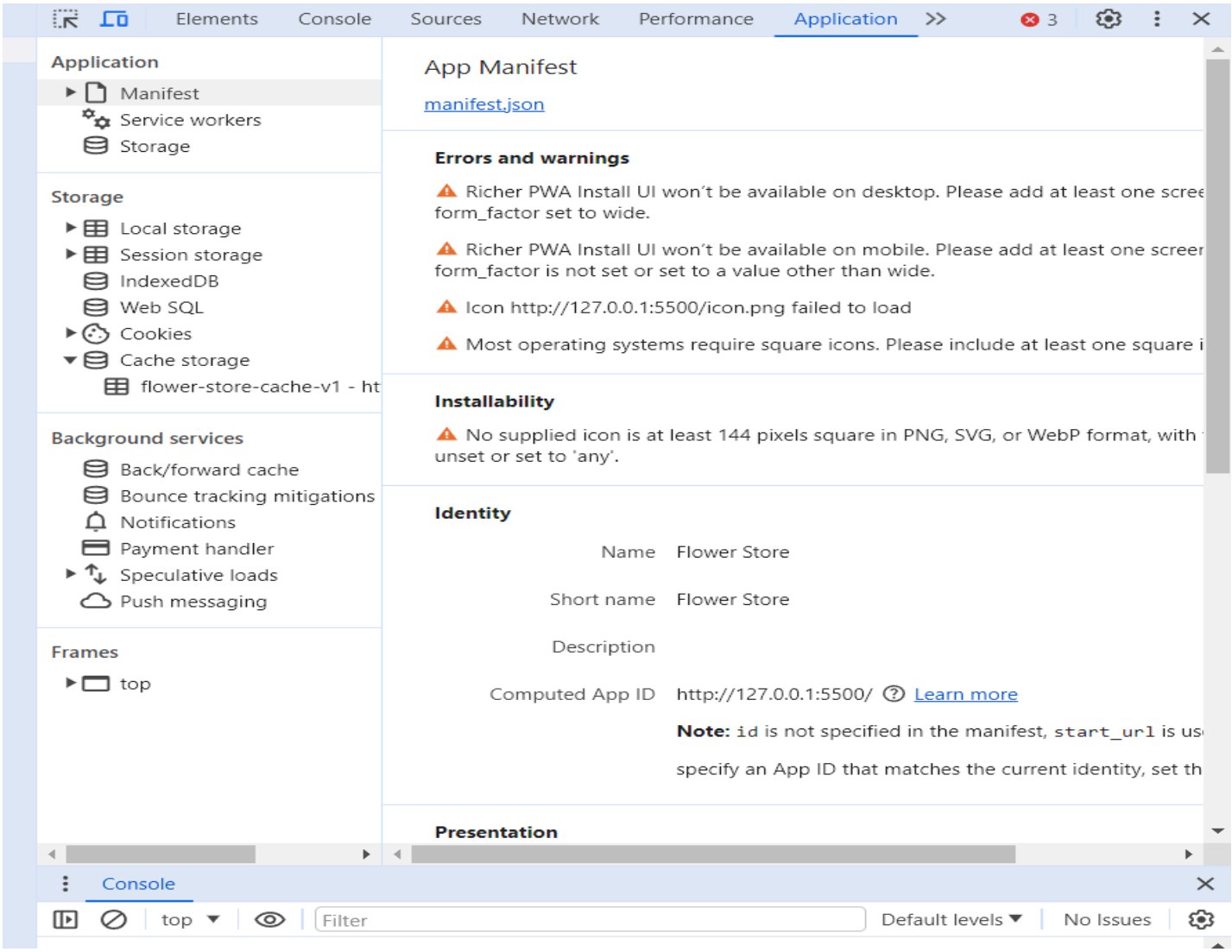
index.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Flower Store</title>
    <link rel="stylesheet" href="style.css">
    <link rel="manifest" href="manifest.json">
</head>
<body>
    <header>
        <h1>Flower Store</h1>
        <nav>
            <ul>
                <li><a href="#">Home</a></li>
                <li><a href="#">Shop</a></li>
                <li><a href="#">Cart</a></li>
            </ul>
        </nav>
    </header>
    <main>
        <!-- Product listing will be dynamically generated here -->
    </main>
    <footer>
        <p>&copy; 2024 Flower Store</p>
    </footer>
    <script src="app.js"></script>
</body>
</html>
```

manifest.json

```
{
    "name": "Flower Store",
    "short_name": "Flower Store",
    "start_url": "/",
    "display": "standalone",
    "background_color": "#ffffff",
    "theme_color": "#ffffff",
    "icons": [
        {
            "src": "icon.png",
            "sizes": "192x192",
            "type": "image/png"
        }
    ]
}
```





Conclusion: Thus writing metadata for the PWA, especially for an eCommerce application, is crucial for enabling features like the "add to homescreen" functionality. By crafting a well-structured manifest.json file with accurate metadata properties such as name, description, icons, and colors, developers can enhance the accessibility and user experience of their PWAs.