Experiment 07

Aim: To write metadata in a Web App manifest file for an E-commerce Progressive Web App (PWA) to enable the **"Add to Home Screen"** feature. This functionality allows users to install the web app on their devices and use it like a native application.

Theory:

Regular Web App vs Progressive Web App (PWA):

- **Regular Web App:** A regular web app is simply a website that is accessible through a browser on mobile or desktop devices. While it adapts to screen sizes using responsive design, it lacks native app-like functionality.

- **Progressive Web App (PWA):** A PWA is an enhanced version of a regular web app that brings several features like offline functionality, push notifications, faster load times, and the ability to be added to the home screen. PWAs aim to offer a native app experience directly in the browser.

Key Differences:

- PWAs can be installed on the home screen of a device, and they function like native apps even without the need to visit an app store.
- PWAs work offline or in low-network conditions, ensuring a better user experience across various scenarios.

What are PWAs?

A **Progressive Web App (PWA)** is a web application built with modern web technologies such as HTML, CSS, and JavaScript. It delivers an app-like experience using a browser but also offers features like offline support, push notifications, and home screen installation, which makes it behave similarly to a native app. The key advantage of a PWA is that users can install the app without going through app stores and still get an optimized experience across all platforms.

Need for PWAs:

- **Cross-Platform Compatibility:** PWAs can run across different platforms (iOS, Android, Windows, etc.), eliminating the need for separate development for each platform.
- **User Engagement:** With features like push notifications and offline support, PWAs improve user retention and engagement.
- **Cost Efficiency:** PWAs require only one codebase, reducing the cost and effort needed for app development and maintenance.
- **Faster Performance:** PWAs are designed to load quickly, even with low network speeds, which improves overall performance.

Features of PWAs:

1. **Responsive:** Adapt to various screen sizes for both mobile and desktop devices.
2. **Offline Capability:** Continue functioning without an internet connection through caching and service workers.
3. **App-like Experience:** Provide an interface and functionality similar to native mobile apps.
4. **Installable:** Allow users to add the app to their home screen, providing an app-like experience without needing to install from an app store.
5. **Push Notifications:** Notify users of updates and new content even when the app is not open.
6. **Automatic Updates:** Automatically updates the app without requiring user intervention.
7. **Secure:** Use HTTPS to ensure that data transferred between the server and the client is secure.

Advantages of PWAs:

- **Native App Experience:** PWAs deliver an experience that feels like using a native app, providing easy access to mobile device features.
- **Cross-Platform:** One PWA works across multiple platforms (iOS, Android, desktop), reducing development costs and complexity.
- **No App Store Requirement:** PWAs do not need to be published in app stores, meaning faster access and easier updates

- **Offline Functionality:** PWAs can cache content and enable offline use, ensuring that the app remains functional even in areas with poor connectivity.
- **Reduced Data Usage:** The caching mechanisms in PWAs reduce data usage as content is stored locally.
- **Lower Development Costs:** Developing and maintaining a single codebase for a PWA is cheaper than building separate native apps for multiple platforms.

Implementation:

To enable the **"Add to Home Screen"** feature, the following steps were followed:

1. **Create a Web App Manifest File:** The manifest file contains essential metadata about the app, including the app's name, icons, starting URL, and display preferences. This is the core component that allows the app to be recognized and added to the user's home screen.

2. **Link the Manifest to the HTML File:** After creating the manifest file, it is linked within the HTML file to inform the browser about the app's metadata.

3. **Host the App Over HTTPS:** For security reasons, PWAs must be served over HTTPS. This is crucial for the **"Add to Home Screen"** feature to work, as modern browsers only allow this functionality on secure sites.

Github Execution:
https://github.com/brijeshforcollege/PWA-App

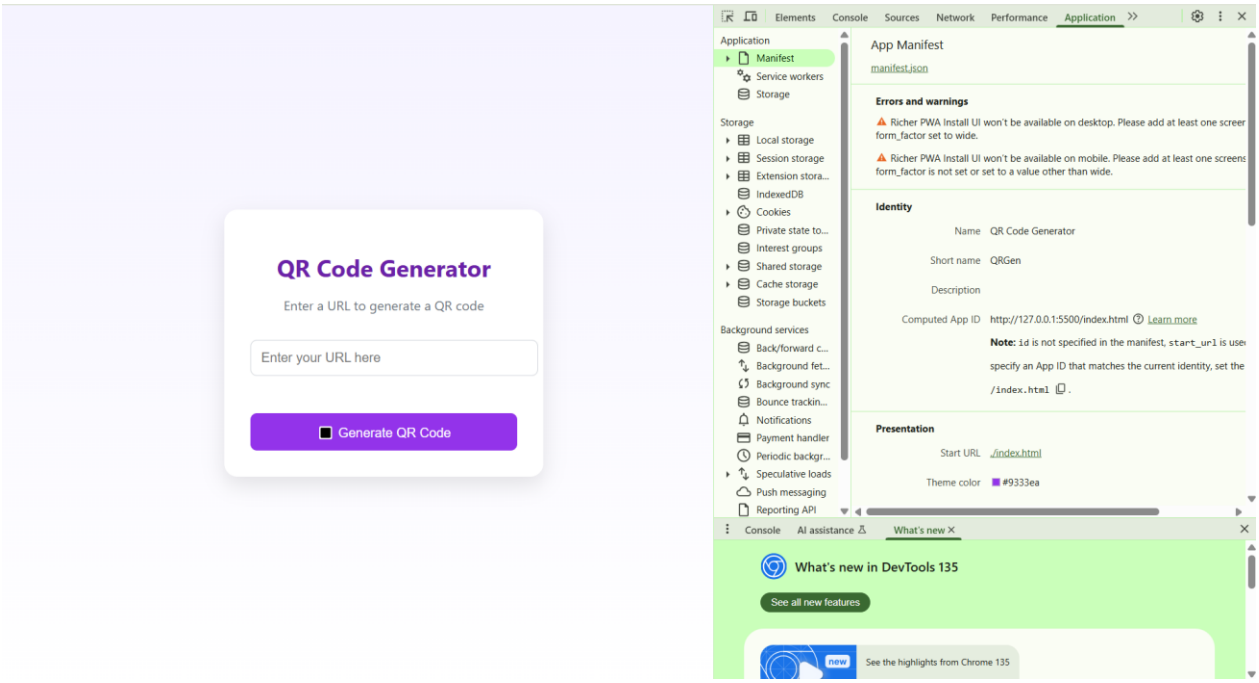
Code:

```
{
  "name": "QR Code Generator",
  "short_name": "QRGen",
  "start_url": "./index.html",
  "display": "standalone",
  "background_color": "#f5f3ff",
  "theme_color": "#9333ea",
  "orientation": "portrait",
  "icons": [
    {
      "src": "images/icon-192.png",
      "sizes": "192x192",
      "type": "image/png"
    },
    {
      "src": "images/icon-512.png",
      "sizes": "512x512",
      "type": "image/png"
    }
```

```
  ]
}
```

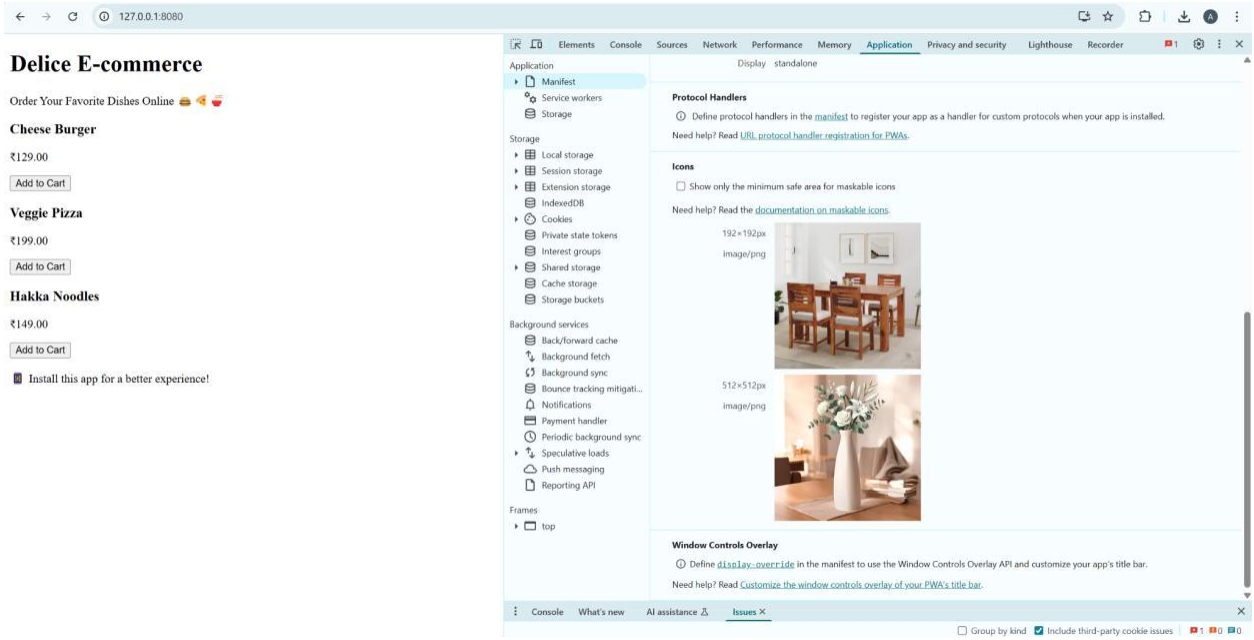
Run the website on local server
Check in the developer tools -> Application -> Manifest

Manifest



Conclusion:

After implementing the Web App manifest and linking it to the HTML file, users visiting the E-commerce website on their mobile or desktop browsers will be prompted with the option to add the app to their home screen. Once added, users can launch the web app from their home screen as if it were a native app. This enhances the user experience by providing easier access to the app and improving engagement.



Conclusion:

By adding metadata to the manifest file and ensuring the correct setup, the **"Add to Home Screen"** feature was successfully enabled for our E-commerce Progressive Web App. This experiment demonstrated the core principles behind PWAs and their advantages over traditional web applications. With features like offline support, cross-platform compatibility, and push notifications, PWAs offer an effective solution for delivering a seamless app-like experience on the web. The ability to install the app on a device's home screen without the need for an app store significantly enhances the accessibility and usability of web applications.