Experiment No.7

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

- **Theory:**

PWA:

Progressive Web Apps (PWAs) are web applications designed to deliver a user experience akin to native mobile apps. They leverage web technologies like HTML, CSS, and JavaScript, enabling them to function seamlessly across various platforms and devices from a unified codebase. Offering features such as offline functionality, push notifications, and integration with device capabilities, PWAs provide users with native-like experiences directly within web browsers. They can be installed on devices, run offline, and integrate with other apps, offering benefits such as lower development costs and easier maintenance compared to traditional apps.

To qualify as a PWA, a site must meet specific technical criteria, including serving content over HTTPS, employing service workers for offline support, and referencing a web app manifest containing essential properties like name, start URL, display mode, and icons. PWAs can be added to the home screen, display splash screens, and provide interactions akin to native apps. Bridging the gap between native applications and the mobile web, PWAs offer a versatile and efficient means of delivering app-like experiences, thereby enhancing user engagement and accessibility across various devices and platforms.

PWA Use:

1. Native-Like Experience: PWAs offer a user experience similar to native mobile apps, with smooth animations, offline functionality, and push notifications.

2. Cross-Platform Compatibility: Built using web technologies like HTML, CSS, and JavaScript, PWAs work seamlessly across different platforms and devices, reducing development efforts and costs.

3. Offline Functionality: PWAs can function offline, allowing users to access content and perform actions even when they're not connected to the internet, enhancing accessibility and usability.

4. Lower Development Costs: With a single codebase for multiple platforms, PWAs offer cost-effective development compared to building separate native apps for each platform.

Steps to implement PWA:

1. **Ensure HTTPS:** Host your website over HTTPS to ensure security, as PWAs require a secure origin to work properly.

2. **Create a Web App Manifest:** Develop a manifest file (manifest.json) that includes metadata such as app name, icons, and display preferences.

3. **Implement Service Workers:** Create and register service workers to manage caching, enable offline functionality, and handle push notifications.

4. **Responsive Design:** Ensure your app is responsive and works well on various screen sizes and devices for a consistent user experience.

5. **Add to Homescreen:** Use web app manifest and service workers to enable the "Add to Homescreen" prompt, allowing users to install the PWA on their devices.

- **Code:-**

Manifest.json:

Create a manifest.json file in your project directory and define metadata about your PWA, such as name, icons, theme color, and starting URL.

Inside Index.html:

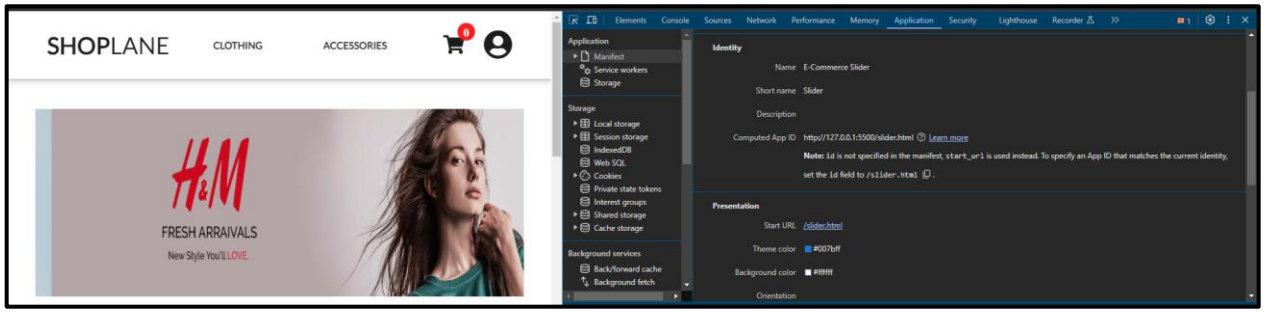
Add a reference to the manifest file in the <head> section of your HTML.

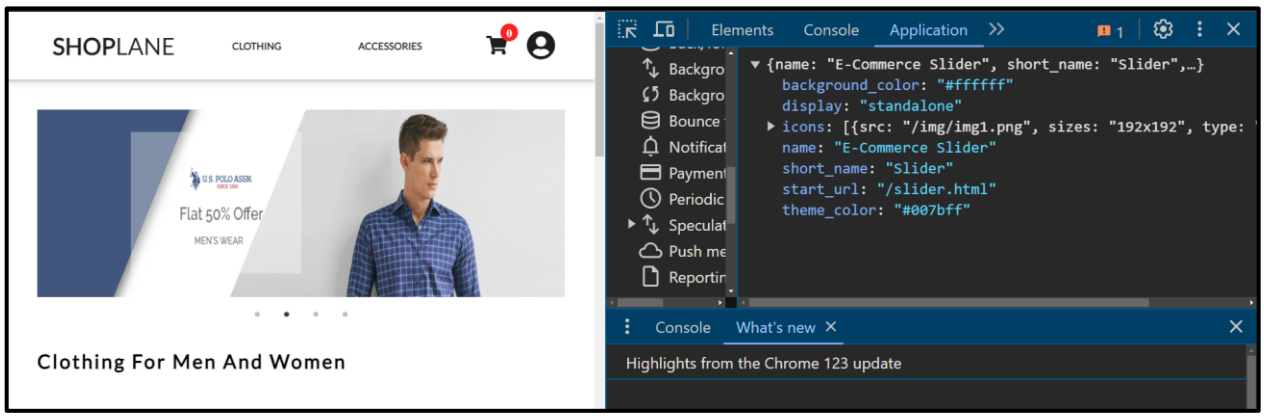
```
<link rel="manifest" href="/manifest.json">
```

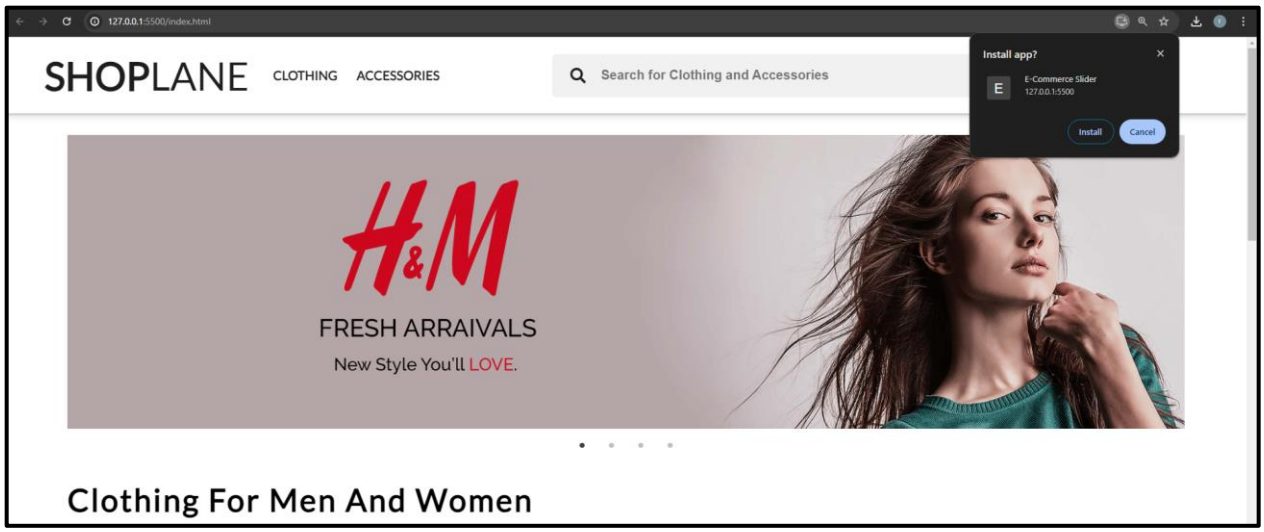
Register a service worker to enable caching and offline support for your PWA. You can use the same service worker registration script as in your main HTML file.

```
<script>
    if ('serviceWorker' in navigator) {
      window.addEv (property) Navigator.serviceWorker: ServiceWorkerContainer
        navigator.serviceWorker.register('/service-worker.js')
          .then(registration => {
            console.log('Service Worker registered:', registration);
          })
          .catch(error => {
            console.error('Service Worker registration failed:', error);
          });
      });
    }
  </script>
```

- **Output:-**







- **Conclusion:**

Enabling the "add to homescreen" feature for an Ecommerce Progressive Web App (PWA) involves crafting a manifest.json file with essential metadata like the app's name, icons, start URL, and display preferences. This streamlined process allows users to seamlessly install the PWA onto their device's homescreen, enhancing accessibility and engagement. By optimizing the Web app manifest file, the Ecommerce experience is transformed, offering the convenience of native app installation while leveraging the versatility and reach of web technologies, ultimately boosting user retention and satisfaction.