Ronak Katariya 23
Prajjwal Pandey 32
Snehal Patil 38

Experiment - 8 : Registering and Activating a Service Worker for E-commerce PWA

Objective:
To code and register a service worker and complete the install and activation process for an eCommerce Progressive Web App (PWA).

Theory:

Service Worker

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

- A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
- Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
- The service worker becomes idle when not in use and restarts when it's next needed. You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.

What can we do with Service Workers?

- You can dominate **Network Traffic**
  You can manage all network traffic of the page and do any manipulations. For example, when the page requests a CSS file, you can send plain text as a

response or when the page requests an HTML file, you can send a png file as a response. You can also send a true response too.

- You can **Cache**
  You can cache any request/response pair with Service Worker and Cache API and you can access these offline content anytime.

- You can manage **Push Notifications**
  You can manage push notifications with Service Worker and show any information message to the user.
- You can **Continue**
  Although Internet connection is broken, you can start any process with Background Sync of Service Worker.

What can’t we do with Service Workers?

- You can’t access the **Window**
  You can’t access the window, therefore, You can’t manipulate DOM elements. But, you can communicate to the window through post Message and manage processes that you want.

- You can’t work it on **80 Port**

Service Worker just can work on HTTPS protocol. But you can work on localhost during development.

A service worker follows three main lifecycle phases:

1. **Installation** - The service worker is downloaded and installed.
2. **Activation** - The service worker takes control of the pages and manages caching.
3. **Fetching and Events Handling** - The service worker intercepts network requests and serves cached responses when necessary.

Requirements:

- A code editor (e.g., VS Code, Sublime Text)
- A basic eCommerce web app
- A browser that supports service workers (e.g., Chrome, Edge, Firefox)

Procedure:

1. Creating the Service Worker File

- In the root directory of the eCommerce PWA, create a file named `service-worker.js`.
- Add the following code to set up basic caching:

```
const CACHE_NAME = 'ecommerce-pwa-v1';
const urlsToCache = [
  '/',
  '/index.html',
  '/styles.css',
  '/script.js',
  '/icons/icon-192x192.png',
  '/icons/icon-512x512.png'
];

// Install event - Caching files
self.addEventListener('install', (event) => {
  event.waitUntil(
    caches.open(CACHE_NAME).then((cache) => {
      return cache.addAll(urlsToCache);
    })
  );
});

// Activate event - Cleanup old caches
self.addEventListener('activate', (event) => {
  event.waitUntil(
    caches.keys().then((cacheNames) => {
      return Promise.all(
        cacheNames.filter((cache) => cache !== CACHE_NAME).map((cache) =>
caches.delete(cache))
```

```
      );
    })
  );
});

// Fetch event - Serve files from cache
self.addEventListener('fetch', (event) => {
  event.respondWith(
    caches.match(event.request).then((response) => {
      return response || fetch(event.request);
    })
  );
});
```

---

2. Registering the Service Worker

- In the `index.html` or `app.js` file, register the service worker with the following code:

```
if ('serviceWorker' in navigator) {
  window.addEventListener('load', () => {
    navigator.serviceWorker.register('/service-worker.js')
      .then((registration) => {
        console.log('Service Worker registered with scope:', registration.scope);
      })
      .catch((error) => {
        console.log('Service Worker registration failed:', error);
      });
  });
}
```

---

3. Testing the Service Worker

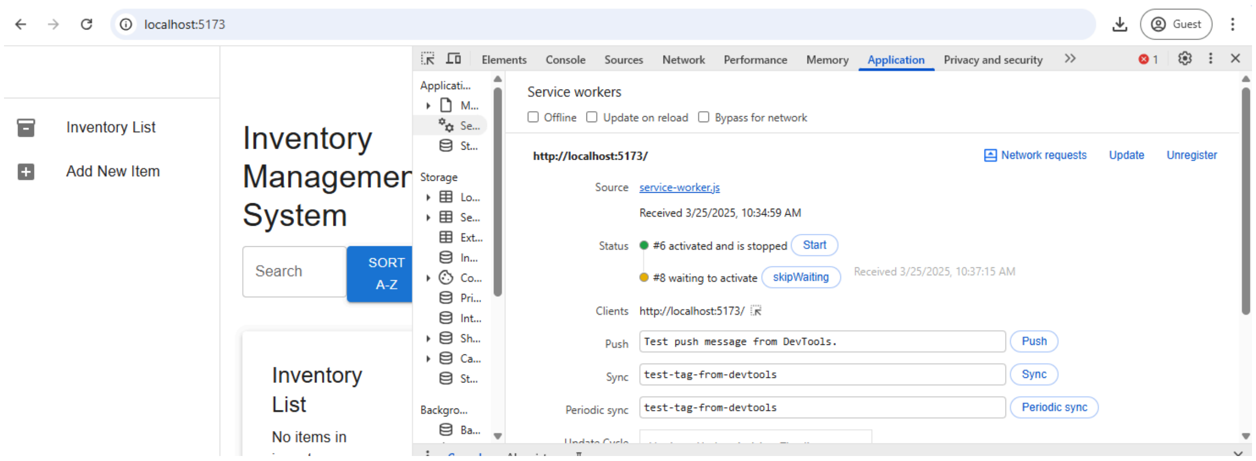
- Host the application on a local server (e.g., using `Live Server` in VS Code or `http-server` in Node.js).
- Open the web application in a browser.
- Open Chrome DevTools (`F12` or `Ctrl+Shift+I`).

- Go to the "Application" tab → "Service Workers".
- Verify the service worker is installed and activated.

Observations:

- The service worker successfully caches essential files.
- The app works offline using cached files.
- The activation process ensures that old caches are removed.

Conclusion:
By coding and registering a service worker, we successfully enabled caching and offline support for the eCommerce PWA, improving performance and reliability.



Service workers

Offline ☐ Update on reload ☐ Bypass for network ☐

http://localhost:5173/

Source service-worker.js

Received 3/25/2025, 10:34:59 AM

Status ● #6 activated and is stopped

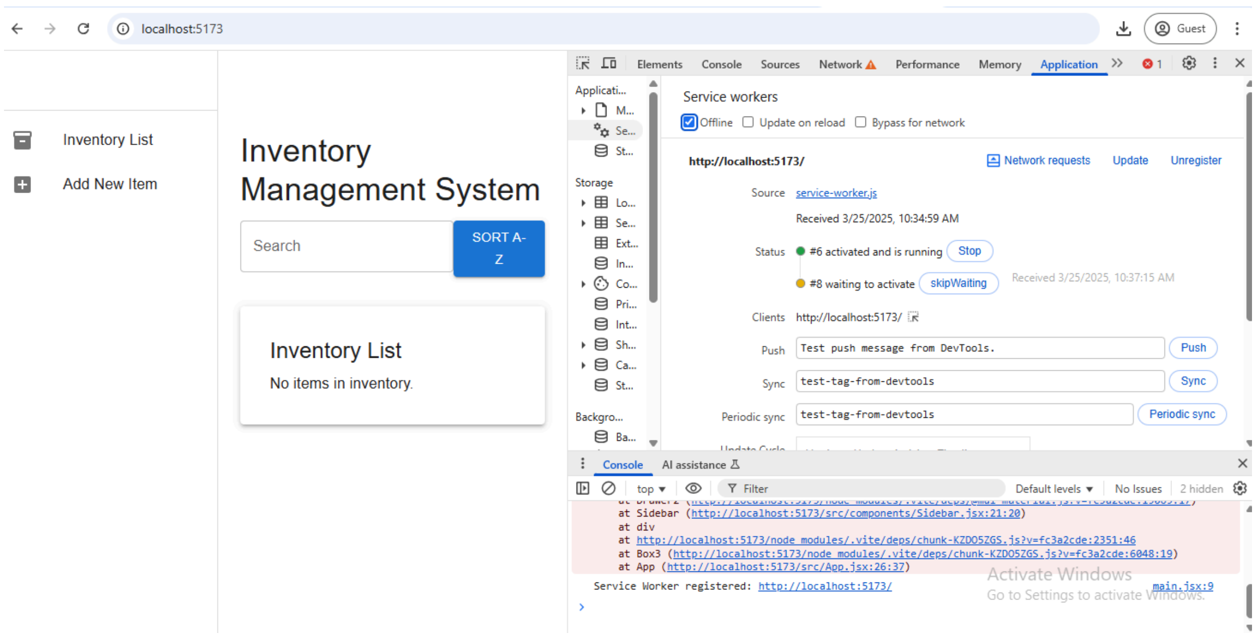
● #8 waiting to activate Received 3/25/2025, 10:37:15 AM

Clients http://localhost:5173/

Push `Test push message from DevTools.`

Sync `test-tag-from-devtools`

Periodic sync `test-tag-from-devtools`



Inventory Management System

Search

SORT A-Z

Inventory List

No items in inventory.

Service workers

[✓ Offline] Update on reload Bypass for network

http://localhost:5173/ Network requests Update Unregister

Source service-worker.js

Received 3/25/2025, 10:34:59 AM

Status #6 activated and is running Stop

#8 waiting to activate skipWaiting Received 3/25/2025, 10:37:15 AM

Clients http://localhost:5173/

Push `Test push message from DevTools.` Push

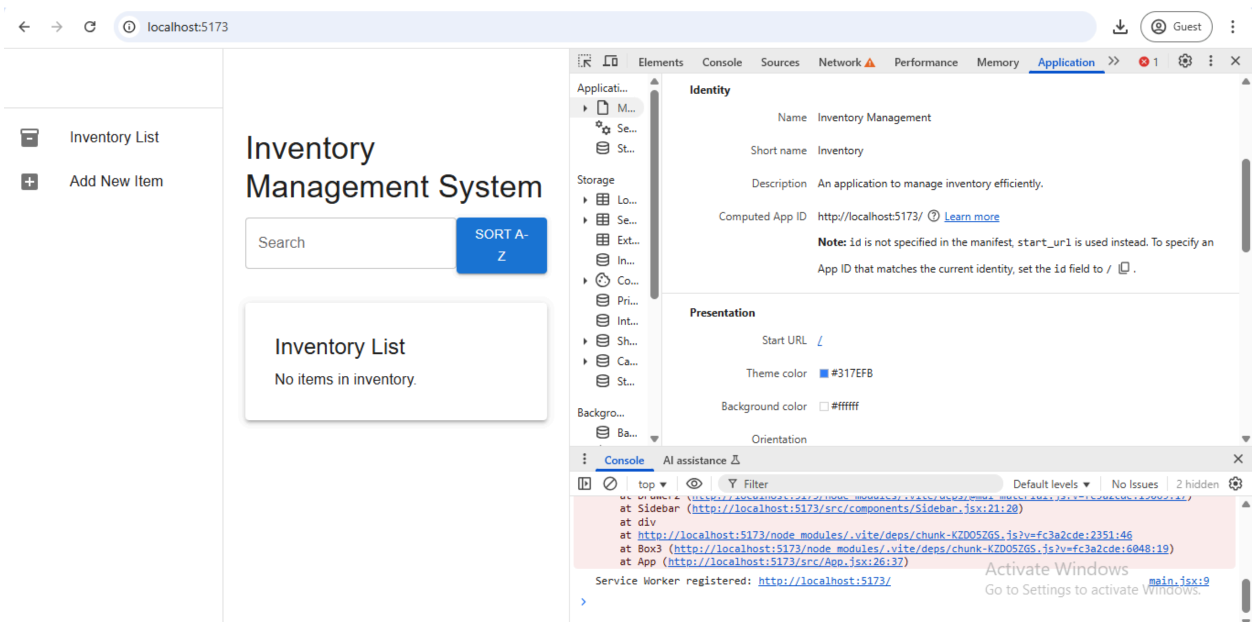
Sync `test-tag-from-devtools` Sync

Periodic sync `test-tag-from-devtools` Periodic sync

Console AI assistance

top Filter Default levels No Issues 2 hidden

```
    at Sidebar (http://localhost:5173/src/components/Sidebar.jsx:21:20)
    at div
    at http://localhost:5173/node_modules/.vite/deps/chunk-KZDO5ZGS.js?v=fc3a2cde:2351:46
    at Box3 (http://localhost:5173/node_modules/.vite/deps/chunk-KZDO5ZGS.js?v=fc3a2cde:6048:19)
    at App (http://localhost:5173/src/App.jsx:26:37)
  Service Worker registered: http://localhost:5173/                main.jsx:9
>
```



Inventory Management System

Search

SORT A-Z

Inventory List

No items in inventory.

Identity

Name Inventory Management

Short name Inventory

Description An application to manage inventory efficiently.

Computed App ID http://localhost:5173/ Learn more

Note: id is not specified in the manifest, start_url is used instead. To specify an App ID that matches the current identity, set the id field to / .

Presentation

Start URL /

Theme color 🟦 #317EFB

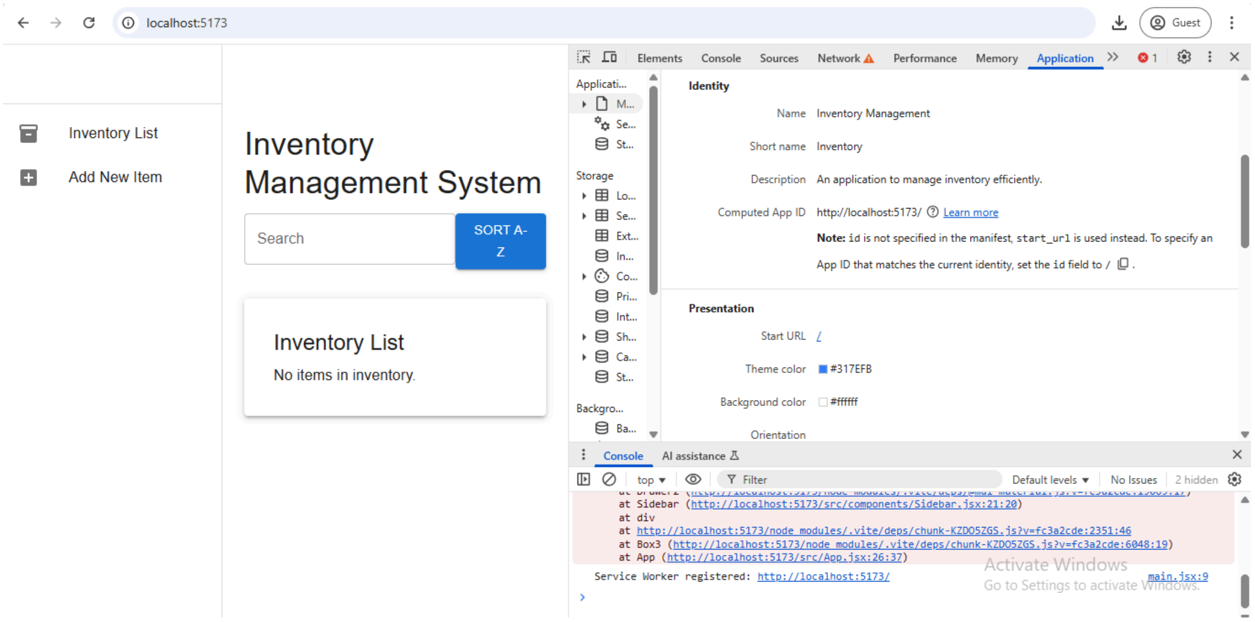
Background color ⬜ #ffffff

Orientation

Console AI assistance

top Filter Default levels No Issues 2 hidden

```
    at Sidebar (http://localhost:5173/src/components/Sidebar.jsx:21:20)
    at div
    at http://localhost:5173/node_modules/.vite/deps/chunk-KZDO5ZGS.js?v=fc3a2cde:2351:46
    at Box3 (http://localhost:5173/node_modules/.vite/deps/chunk-KZDO5ZGS.js?v=fc3a2cde:6048:19)
    at App (http://localhost:5173/src/App.jsx:26:37)
  Service Worker registered: http://localhost:5173/                main.jsx:9
>
```


localhost:5173

Inventory List
Add New Item

Inventory Management System

Search | SORT A-Z

Inventory List
No items in inventory.

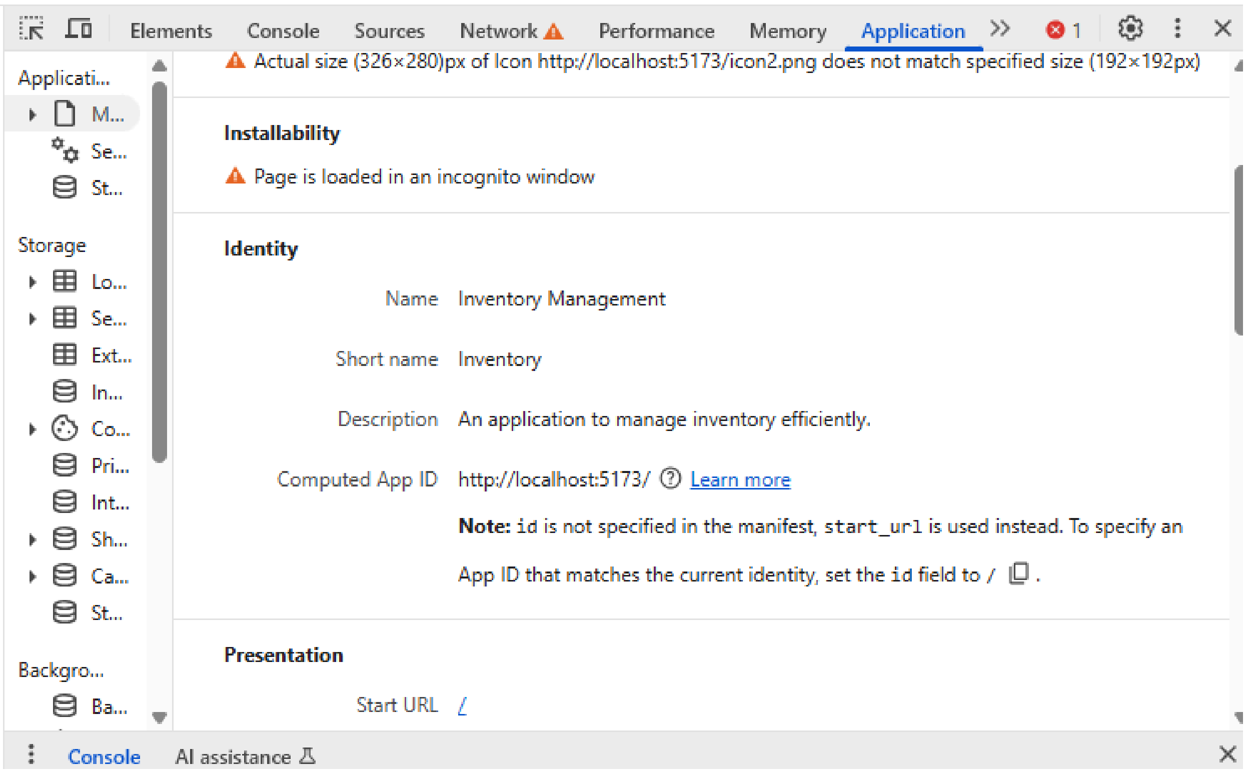
Identity

Name	Inventory Management
Short name	Inventory
Description	An application to manage inventory efficiently.
Computed App ID	http://localhost:5173/ Learn more
	Note: id is not specified in the manifest, start_url is used instead. To specify an App ID that matches the current identity, set the id field to / .

Presentation

Start URL	/
Theme color	🟦 #317EFB
Background color	⬜ #ffffff
Orientation	

```
    at Sidebar (http://localhost:5173/src/components/Sidebar.jsx:21:20)
    at div
    at http://localhost:5173/node_modules/.vite/deps/chunk-KZDO5ZGS.js?v=fc3a2cde:2351:46
    at Box3 (http://localhost:5173/node_modules/.vite/deps/chunk-KZDO5ZGS.js?v=fc3a2cde:6048:19)
    at App (http://localhost:5173/src/App.jsx:26:37)
  Service Worker registered: http://localhost:5173/                          main.jsx:9
```


⚠ Actual size (326×280)px of Icon http://localhost:5173/icon2.png does not match specified size (192×192px)

Installability

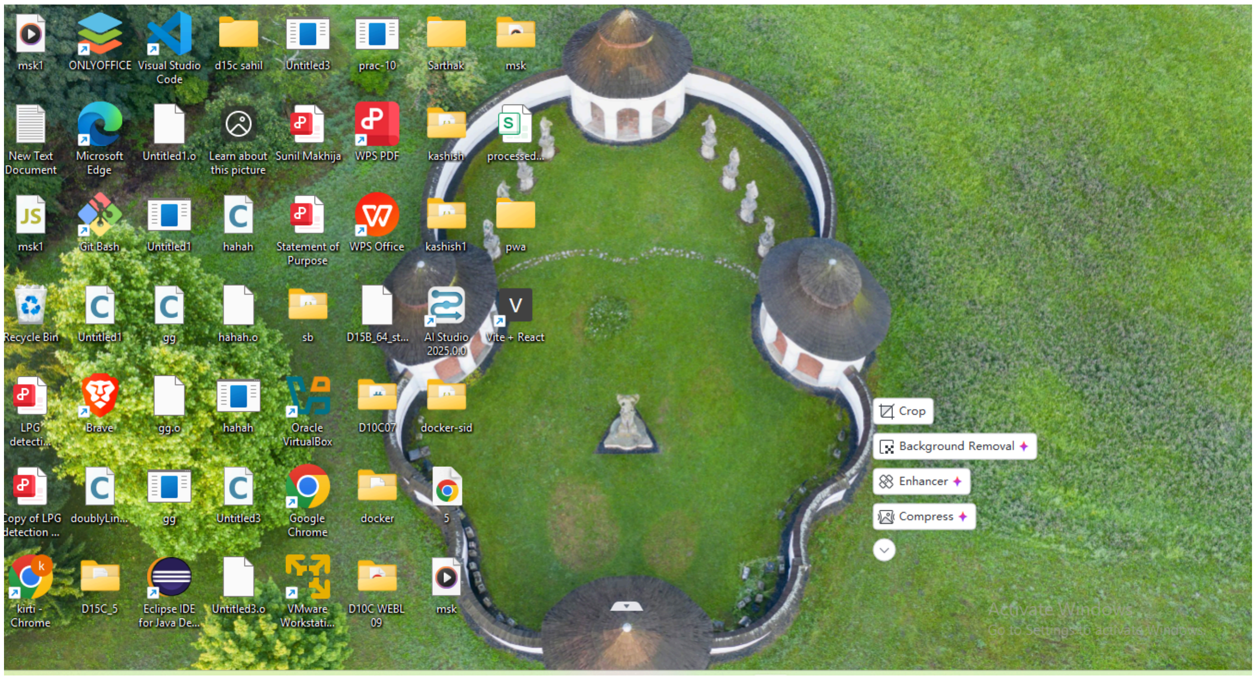
⚠ Page is loaded in an incognito window

Identity

Name	Inventory Management
Short name	Inventory
Description	An application to manage inventory efficiently.
Computed App ID	http://localhost:5173/ Learn more
	Note: id is not specified in the manifest, start_url is used instead. To specify an App ID that matches the current identity, set the id field to / .

Presentation

Start URL	/


msk1
ONLYOFFICE
Visual Studio Code
d15c sahil
Untitled3
prac-10
Sarthak
msk
New Text Document
Microsoft Edge
Untitled1.o
Learn about this picture
Sunil Makhija
WPS PDF
kashish
processed...
msk1
Git Bash
Untitled1
hahah
Statement of Purpose
WPS Office
kashish1
pwa
Recycle Bin
Untitled1
gg
hahah.o
sb
D15B_64_st...
AI Studio 2025.0.0
Vite + React
LPG detecti...
Brave
gg.o
hahah
Oracle VirtualBox
D10C07
docker-sid
Copy of LPG detection ...
doublyLin...
gg
Untitled3
Google Chrome
docker
5
kirti - Chrome
D15C_5
Eclipse IDE for Java De...
Untitled3.o
VMware Workstati...
D10C WEBL 09
msk
Crop
Background Removal
Enhancer
Compress
Activate Windows
Go to Settings to activate Windows.