Implementing Service Worker Events (Fetch, Sync, Push) for E-Commerce PWA

Progressive Web Apps (PWAs) are web applications that offer app-like experiences through modern web capabilities. One of the key components of a PWA is the service worker, which enables features like offline access, background sync, and push notifications. In this document, we will explore how to implement service worker events such as `fetch`, `sync`, and `push` in the context of an e-commerce application. Below is a sample implementation.

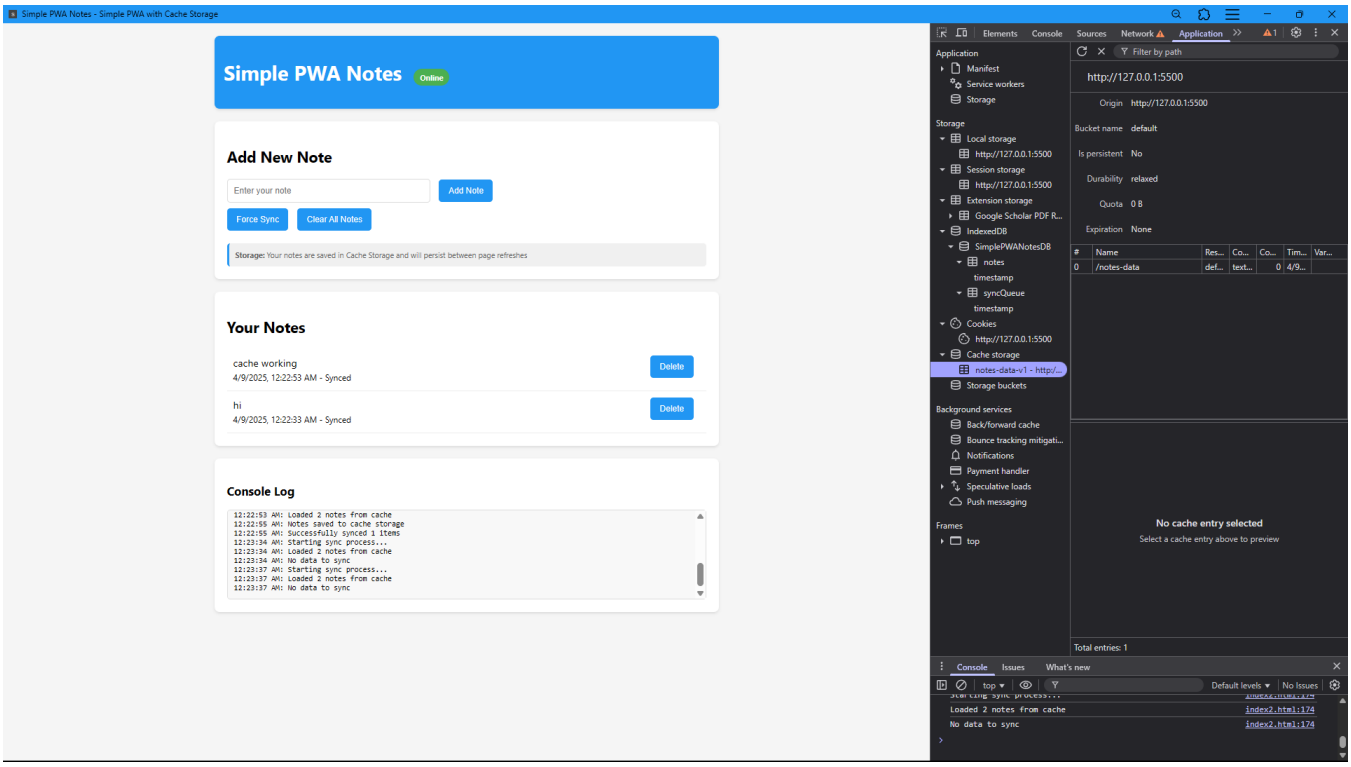
---

1. Caching Static Assets Using Install Event

The `install` event is triggered when the service worker is installed. During this phase, essential files are cached to enable offline access.

```
const CACHE_NAME = "campquest-v1"; const
ASSETS_TO_CACHE = [
  "/",
  "/index.html",
  "/src/main.jsx",
  "/CampQuest.svg",
  "/manifest.json",
];

self.addEventListener("install", (event) => {
event.waitUntil(
caches.open(CACHE_NAME).then((cache) => {
return cache.addAll(ASSETS_TO_CACHE);
    })
  );
});
```



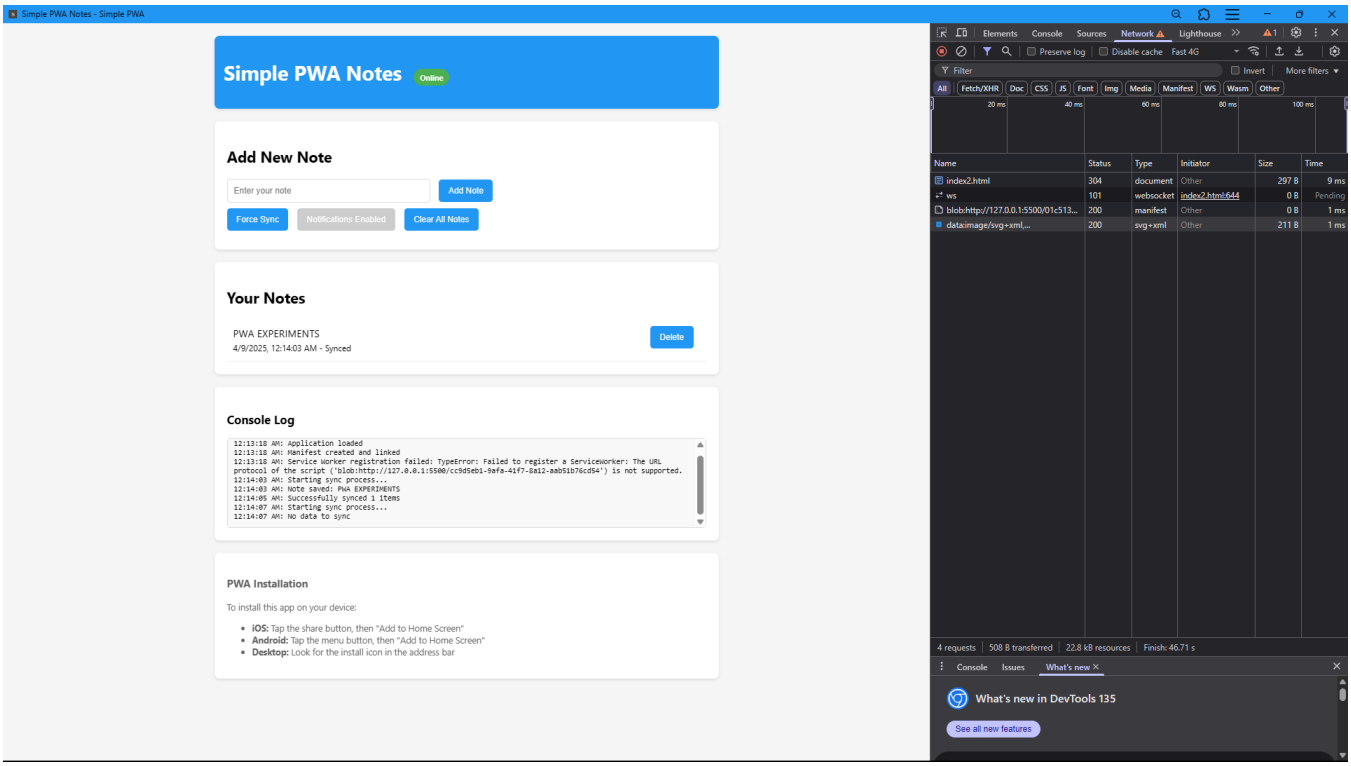
---

2. Handling Fetch Requests

The `fetch` event intercepts network requests. We use this event to implement a cache-first or network-first strategy depending on the URL path.

```
self.addEventListener("fetch", (event) => {   const url =
new URL(event.request.url);   if
(url.pathname.startsWith("/campgrounds")) {
event.respondWith(       fetch(event.request)
.then((response) => {          const responseClone =
response.clone();
caches.open(CACHE_NAME).then((cache) => {
cache.put(event.request, responseClone);
        });
        return response;
      })
      .catch(() => {          return
caches.match(event.request);
```

```
      })
    );
  } else {     event.respondWith(
caches.match(event.request).then((response) => {
return response || fetch(event.request);
      })
    );
  }
});
```



3. Background Sync (Conceptual Example)

The `sync` event is used to defer actions until the user has stable connectivity. For an e-commerce app, you could use this to sync cart data or orders.

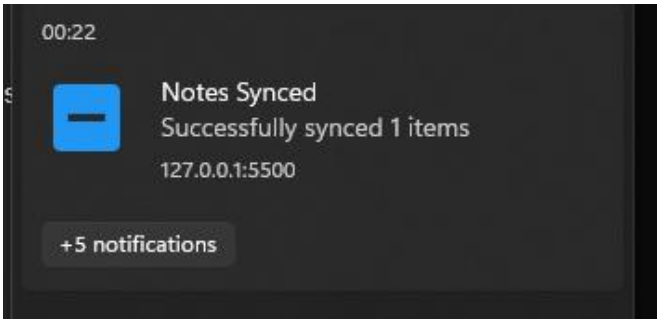
```
self.addEventListener("sync", (event) => {
if (event.tag === "sync-cart") {
event.waitUntil(
    // Logic to sync cart data with server
  );
}
```

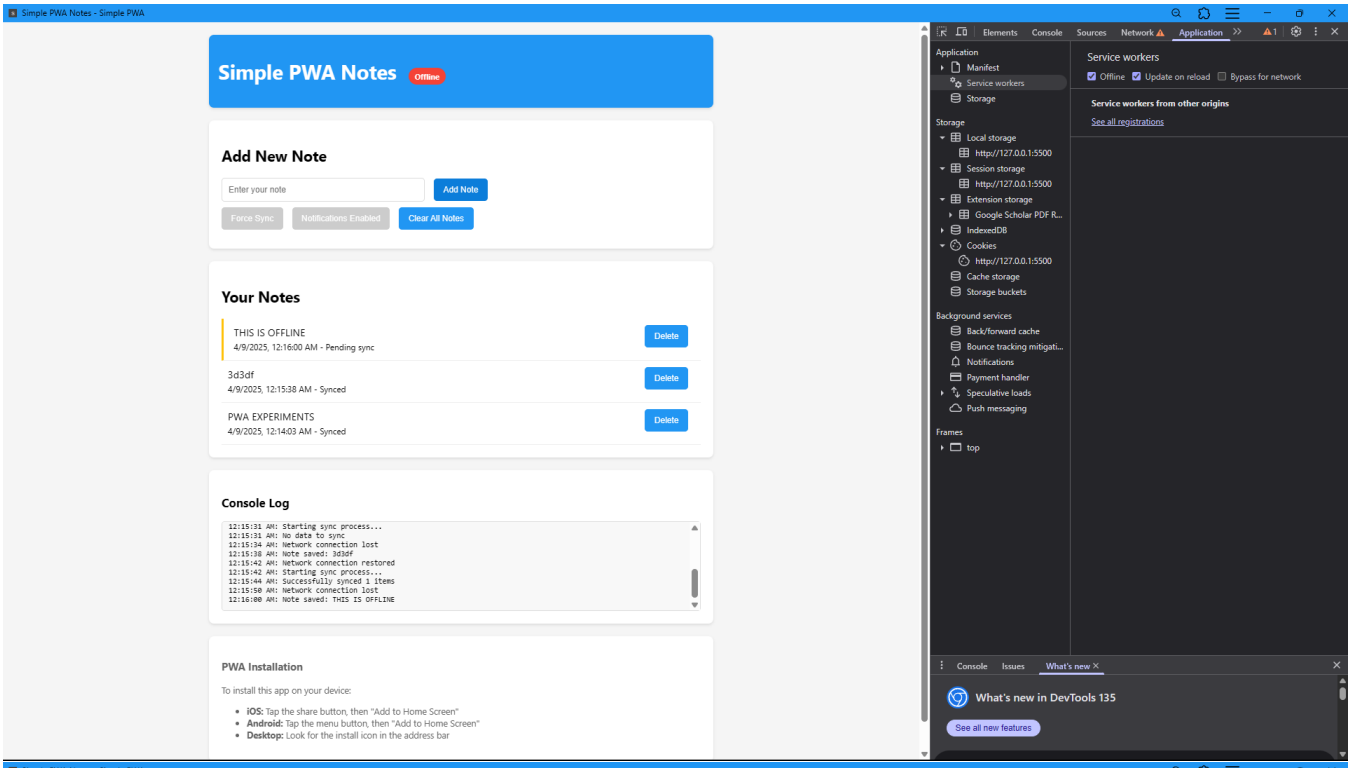
});

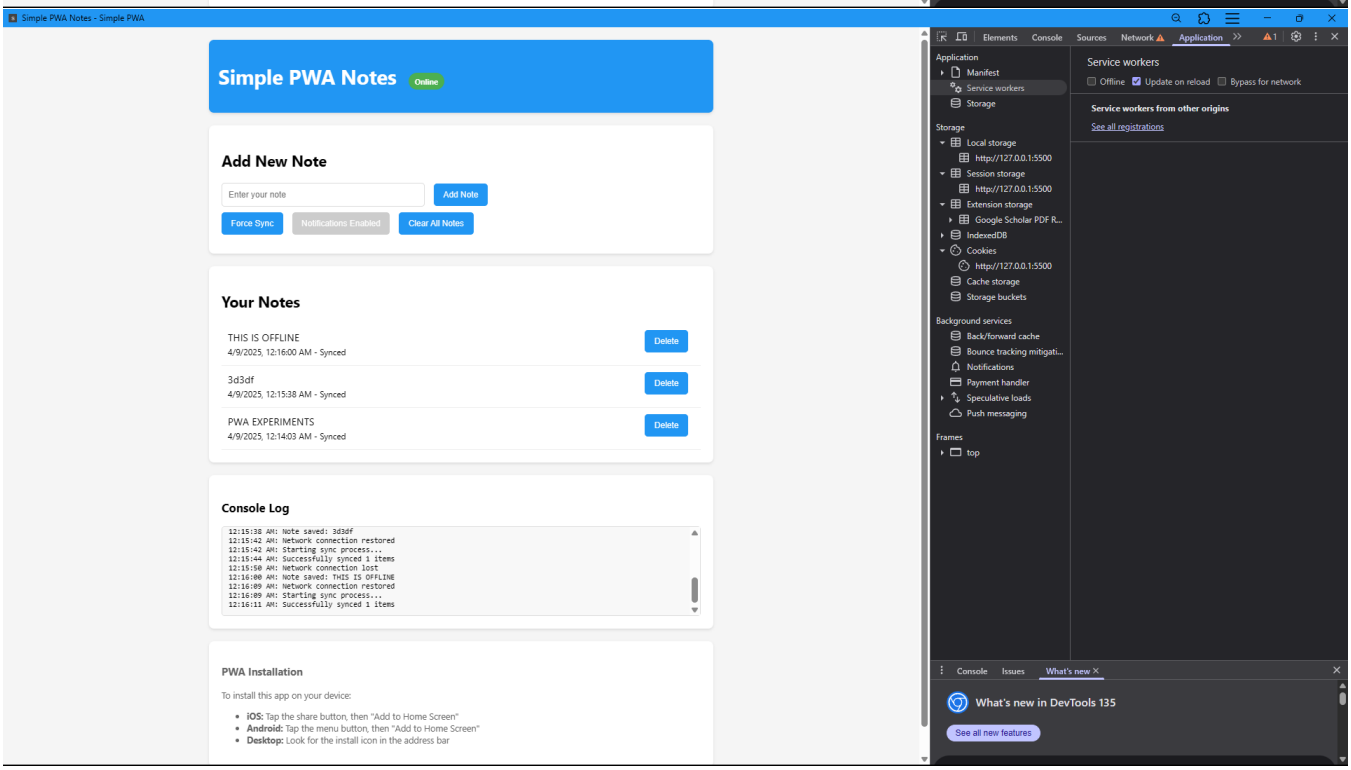
4. Push Notifications (Conceptual Example)

The `push` event is triggered when a push message is received. This could be used to notify users of new deals or order status updates.

```
self.addEventListener("push", (event) => {
const data = event.data.json();   const
options = {     body: data.body,     icon:
"/icon.png",
  };
  event.waitUntil(
self.registration.showNotification(data.title, options)
  );
});
```




Simple PWA Notes - Simple PWA
Simple PWA Notes
Offline
Add New Note
Enter your note
Add Note
Force Sync
Notifications Enabled
Clear All Notes
Your Notes
THIS IS OFFLINE
4/9/2025, 12:16:00 AM - Pending sync
Delete
3d3df
4/9/2025, 12:15:38 AM - Synced
Delete
PWA EXPERIMENTS
4/9/2025, 12:14:03 AM - Synced
Delete
Console Log
12:15:31 AM: Starting sync process...
12:15:31 AM: No data to sync
12:15:34 AM: Network connection lost
12:15:38 AM: Note saved: 3d3df
12:15:42 AM: Network connection restored
12:15:42 AM: Starting sync process...
12:15:44 AM: Successfully synced 1 items
12:15:50 AM: Network connection lost
12:16:00 AM: Note saved: THIS IS OFFLINE
PWA Installation
To install this app on your device:
iOS: Tap the share button, then "Add to Home Screen"
Android: Tap the menu button, then "Add to Home Screen"
Desktop: Look for the install icon in the address bar
Elements
Console
Sources
Network
Application
Service workers
Offline
Update on reload
Bypass for network
Service workers from other origins
See all registrations
Manifest
Storage
Local storage
Session storage
Extension storage
IndexedDB
Cookies
Cache storage
Storage buckets
Background services
Back/forward cache
Notifications
Payment handler
Speculative loads
Push messaging
Frames
top
Issues
What's new
What's new in DevTools 135
See all new features


Simple PWA Notes - Simple PWA
Simple PWA Notes
Online
Add New Note
Enter your note
Add Note
Force Sync
Notifications Enabled
Clear All Notes
Your Notes
THIS IS OFFLINE
4/9/2025, 12:16:00 AM - Synced
Delete
3d3df
4/9/2025, 12:15:38 AM - Synced
Delete
PWA EXPERIMENTS
4/9/2025, 12:14:03 AM - Synced
Delete
Console Log
12:15:38 AM: Note saved: 3d3df
12:15:42 AM: Network connection restored
12:15:42 AM: Starting sync process...
12:15:44 AM: Successfully synced 1 items
12:15:50 AM: Network connection lost
12:16:00 AM: Note saved: THIS IS OFFLINE
12:16:09 AM: Network connection restored
12:16:09 AM: Starting sync process...
12:16:11 AM: Successfully synced 1 items
PWA Installation
To install this app on your device:
iOS: Tap the share button, then "Add to Home Screen"
Android: Tap the menu button, then "Add to Home Screen"
Desktop: Look for the install icon in the address bar
Elements
Console
Sources
Network
Application
Service workers
Offline
Update on reload
Bypass for network
Service workers from other origins
See all registrations
Manifest
Storage
Local storage
Session storage
Extension storage
IndexedDB
Cookies
Cache storage
Storage buckets
Background services
Back/forward cache
Notifications
Payment handler
Speculative loads
Push messaging
Frames
top
Issues
What's new
What's new in DevTools 135
See all new features

Conclusion

Service workers are powerful tools in building resilient and engaging e-commerce PWAs. By handling `install`, `fetch`, `sync`, and `push` events effectively, you can create a seamless experience for users, even in offline or low-connectivity scenarios.