- Aim :To implement Service Worker events like fetch, sync, and push for the E-commerce PWA.

- Theory:Service workers provide powerful features that allow web applications to operate reliably, even under unreliable network conditions. Key events such as fetch, sync, and push enhance the performance, offline capability, and user engagement of a Progressive Web App (PWA).
- Fetch Event The fetch event allows the service worker to intercept network requests made by the PWA. This is commonly used for caching strategies, such as:
    - Cache-first: Serve content from cache, then update in the background.
    - Network-first: Try to fetch from the network, fall back to cache if offline.

  This is useful for delivering product pages, images, or static assets quickly and reliably.
- Sync Event : The sync event, especially background sync, helps the app manage tasks when connectivity is restored. For instance, if a user submits an order or review while offline, the service worker can save it locally and send it once the connection is re-established.
    - Requires registering a sync task using registration.sync.register('tag-name').
- Push Event : The push event enables the PWA to receive push notifications from a server, even when the app is not open. This is ideal for eCommerce apps to send:
    - Order updates
    - Promotional offers
    - Cart reminders

  It requires integration with a push service and permission from the user.

Code :
Serviceworker.js

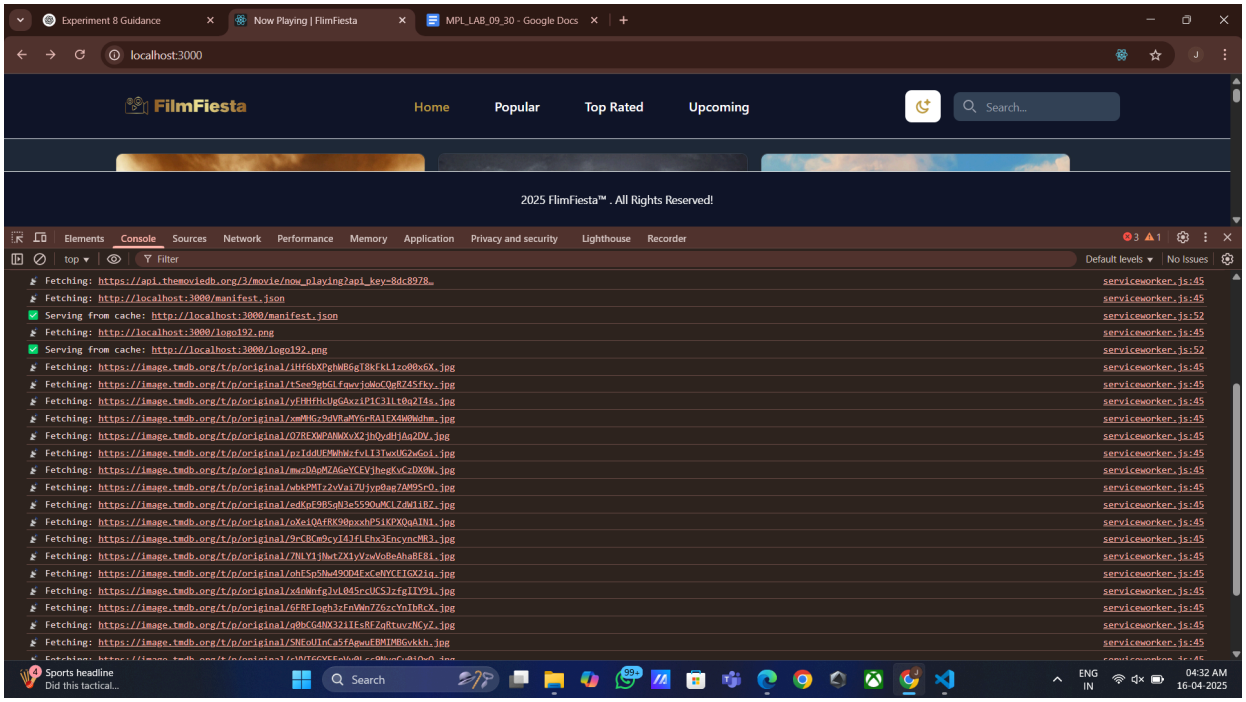
```
/* eslint-disable no-restricted-globals */
const CACHE_NAME = "filmfiesta-cache-v1";

// List of files to cache initially
const urlsToCache = [
  "/",
  "/index.html",
  "/manifest.json",
  "/favicon.ico",
  "/logo192.png",
];

// Install event: cache app shell
self.addEventListener("install", (event) => {
  console.log("🛠️ Service Worker installing...");
  event.waitUntil(
    caches.open(CACHE_NAME).then((cache) => {
```

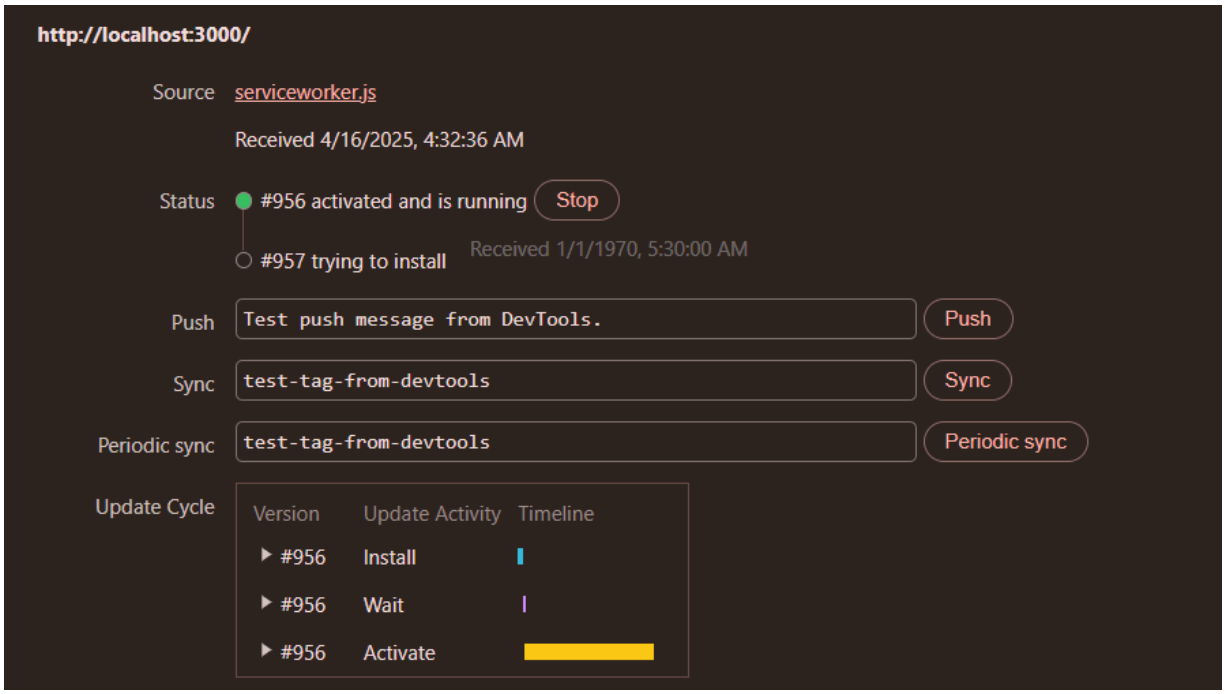
```
      console.log("📦 Caching app shell assets...");
      return cache.addAll(urlsToCache);
    })
  );
  self.skipWaiting();
});

// Activate event: clear old caches
self.addEventListener("activate", (event) => {
  console.log("⚡ Service Worker activating...");
  event.waitUntil(
    caches.keys().then((cacheNames) =>
      Promise.all(
        cacheNames.map((cache) => {
          if (cache !== CACHE_NAME) {
            console.log("🧹 Deleting old cache:", cache);
            return caches.delete(cache);
          }
        })
      )
    )
  );
  self.clients.claim();
});

// Fetch event: cache-first strategy with dynamic caching
self.addEventListener("fetch", (event) => {
  console.log("📡 Fetching:", event.request.url);

  if (event.request.method !== "GET") return;

  event.respondWith(
    caches.match(event.request).then((cachedResponse) => {
      if (cachedResponse) {
        console.log("✅ Serving from cache:", event.request.url);
        return cachedResponse;
      }

      return fetch(event.request)
        .then((networkResponse) => {
```

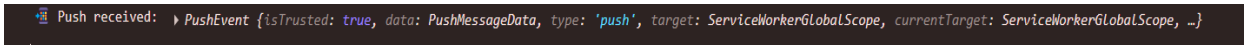
```
          // Only cache successful responses
          if (
            !event.request.url.startsWith("http") ||
            networkResponse.status !== 200 ||
            networkResponse.type !== "basic"
          ) {
            return networkResponse;
          }

          // Clone & cache the response
          const clonedResponse = networkResponse.clone();
          caches.open(CACHE_NAME).then((cache) => {
            cache.put(event.request, clonedResponse);
            console.log("➕ Added to cache:", event.request.url);
          });

          return networkResponse;
        })
        .catch(() => {
          // Optional: return fallback HTML or image if needed
          return caches.match("/index.html");
        });
    })
  );
});
```

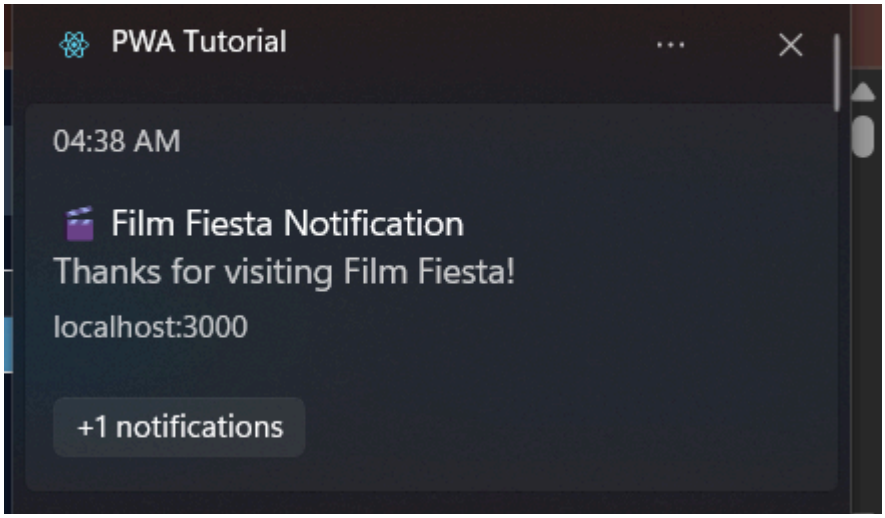
- Output :

```
Fetching: http://localhost:3000/
Fetching: http://localhost:3000/static/css/main.edc88386.css
Fetching: http://localhost:3000/manifest.json
Serving from cache: http://localhost:3000/static/css/main.edc88386.css
Serving from cache: http://localhost:3000/
Serving from cache: http://localhost:3000/manifest.json
```


Experiment 8 Guidance
Now Playing | FlimFiesta
MPL_LAB_09_30 - Google Docs
localhost:3000
FilmFiesta
Home
Popular
Top Rated
Upcoming
Search...
2025 FlimFiesta™ . All Rights Reserved!
Elements
Console
Sources
Network
Performance
Memory
Application
Privacy and security
Lighthouse
Recorder
Filter
Default levels
No Issues
Fetching: https://api.themoviedb.org/3/movie/now_playing?api_key=8dc8978…
Fetching: http://localhost:3000/manifest.json
Serving from cache: http://localhost:3000/manifest.json
Fetching: http://localhost:3000/logo192.png
Serving from cache: http://localhost:3000/logo192.png
Fetching: https://image.tmdb.org/t/p/original/iHf6bXPghWB6gT8kFkl1zo00x6X.jpg
Fetching: https://image.tmdb.org/t/p/original/tSee9gbGLfqwvjoWoCQpRZ45fky.jpg
Fetching: https://image.tmdb.org/t/p/original/yFHHfHcUgGAxziP1C3lLt0q2T4s.jpg
Fetching: https://image.tmdb.org/t/p/original/xmMHGz9dVRaMY6rRA1EX4WOWdhm.jpg
Fetching: https://image.tmdb.org/t/p/original/O7REXWPANWXvX2jbQydHjAq2DV.jpg
Fetching: https://image.tmdb.org/t/p/original/pzIddUEMWhWzfvLI3TwxUG2wGoi.jpg
Fetching: https://image.tmdb.org/t/p/original/mwzDApMZAGeYCEVjhegKvCzDX0W.jpg
Fetching: https://image.tmdb.org/t/p/original/wbkPMTz2vVai7Ujyp0ag7AM9SrO.jpg
Fetching: https://image.tmdb.org/t/p/original/edKpE9B5qN3e559OuMCLZdW1iBZ.jpg
Fetching: https://image.tmdb.org/t/p/original/oXeiQAfRK90pxxhP5iKPXQqAIN1.jpg
Fetching: https://image.tmdb.org/t/p/original/9rCBCm9cyI4lfLEhx3EncyncMR3.jpg
Fetching: https://image.tmdb.org/t/p/original/7NLY1jNwtZX1yVzwVoBeAhaBE8i.jpg
Fetching: https://image.tmdb.org/t/p/original/obESp5Nw49OD4ExCeNYCEIGX2iq.jpg
Fetching: https://image.tmdb.org/t/p/original/x4nWnfgJvL045rcUCSJzfgIIY9i.jpg
Fetching: https://image.tmdb.org/t/p/original/6FRFIogh3zFnVWn7Z6zcYnIbRcX.jpg
Fetching: https://image.tmdb.org/t/p/original/q0bCG4NX32iIEsRFZqRtuvzNCyZ.jpg
Fetching: https://image.tmdb.org/t/p/original/SNEoUInCa5fAgwuEBMIM8Gvkkh.jpg
serviceworker.js:45
serviceworker.js:52
Sports headline
Did this tactical...
Search
ENG
IN
04:32 AM
16-04-2025


http://localhost:3000/
Source
serviceworker.js
Received 4/16/2025, 4:32:36 AM
Status
#956 activated and is running
Stop
#957 trying to install
Received 1/1/1970, 5:30:00 AM
Push
Test push message from DevTools.
Push
Sync
test-tag-from-devtools
Sync
Periodic sync
test-tag-from-devtools
Periodic sync
Update Cycle
Version
Update Activity
Timeline
#956
Install
#956
Wait
#956
Activate


Push received:
PushEvent {isTrusted: true, data: PushMessageData, type: 'push', target: ServiceWorkerGlobalScope, currentTarget: ServiceWorkerGlobalScope, …}



- Conclusion : Implementing fetch, sync, and push events in the service worker of an E-commerce PWA significantly enhances the app’s usability, resilience, and user engagement. These features enable the PWA to function offline, perform background tasks, and communicate with users proactively, providing a seamless and efficient shopping experience.