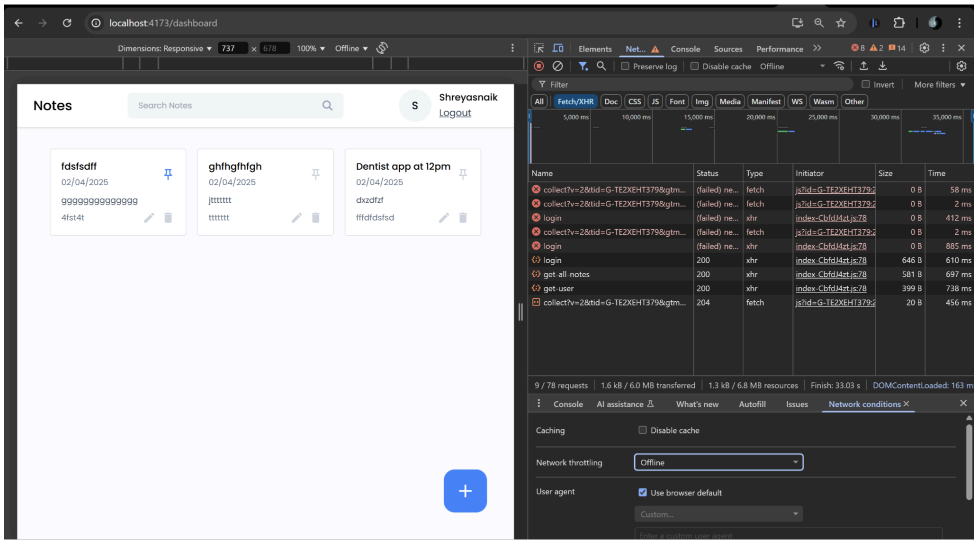
Service Workers are powerful scripts that run in the background and enable rich offline experiences, background sync, and push notifications. In this project, we implemented three core events: **fetch**, **sync**, and **push**.

1. `fetch` Event – For Offline Caching

What it does:
Intercepts network requests made by the app and allows the Service Worker to serve cached content when offline.

Use case in app:
Ensures that notes and assets like images or scripts can still load when the user is offline.

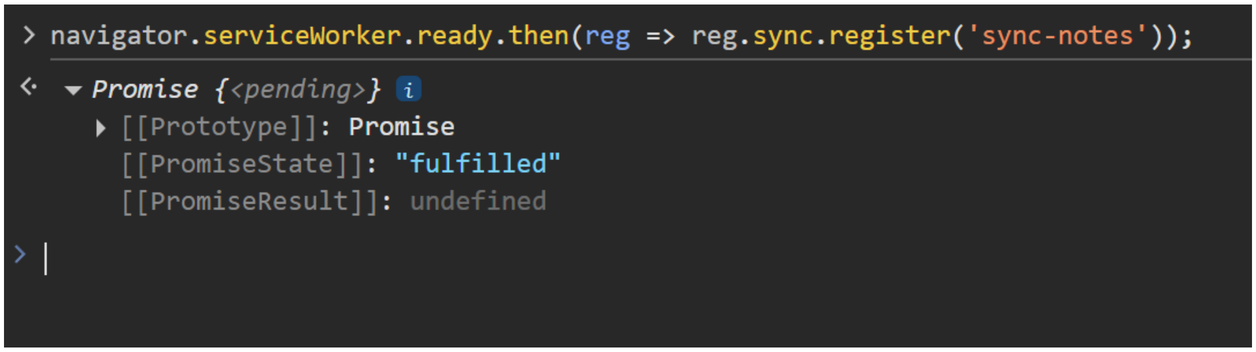
```
self.addEventListener('fetch', (event) => {
  console.log('[SW] Fetching:', event.request.url);
  event.respondWith(
    caches.match(event.request).then((cachedResponse) => {
      return cachedResponse || fetch(event.request);
    })
```



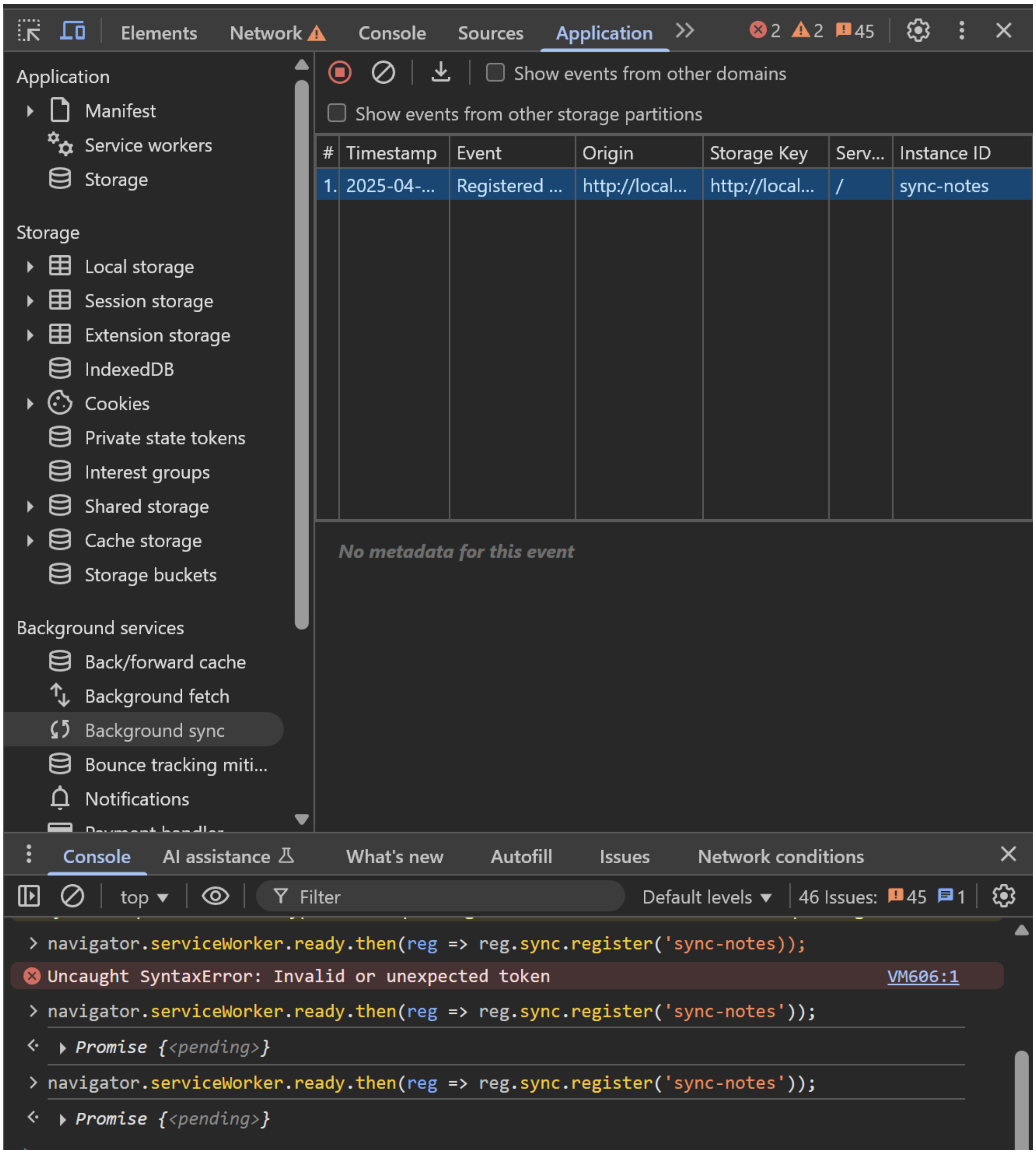
```
Name                                   Status        Type   Initiator                 Size    Time
collect?v=2&tid=G-TE2XEHT379&gtm...    (failed) ne...  fetch  js?id=G-TE2XEHT379:2      0 B   58 ms
collect?v=2&tid=G-TE2XEHT379&gtm...    (failed) ne...  fetch  js?id=G-TE2XEHT379:2      0 B    2 ms
login                                  (failed) ne...  xhr    index-CbfdJ4zt.js:78      0 B  412 ms
collect?v=2&tid=G-TE2XEHT379&gtm...    (failed) ne...  fetch  js?id=G-TE2XEHT379:2      0 B    2 ms
login                                  (failed) ne...  xhr    index-CbfdJ4zt.js:78      0 B  885 ms
login                                  200           xhr    index-CbfdJ4zt.js:78    646 B  610 ms
get-all-notes                          200           xhr    index-CbfdJ4zt.js:78    581 B  697 ms
get-user                               200           xhr    index-CbfdJ4zt.js:78    399 B  738 ms
collect?v=2&tid=G-TE2XEHT379&gtm...    204           fetch  js?id=G-TE2XEHT379:2     20 B  456 ms

9 / 78 requests | 1.6 kB / 6.0 MB transferred | 1.3 kB / 6.8 MB resources | Finish: 33.03 s | DOMContentLoaded: 163 m
```

Caching: Disable cache
Network throttling: Offline
User agent: [✓ Use browser default]



```
> navigator.serviceWorker.ready.then(reg => reg.sync.register('sync-notes'));
<· ▾ Promise {<pending>} i
    ▸ [[Prototype]]: Promise
      [[PromiseState]]: "fulfilled"
      [[PromiseResult]]: undefined
> |
```



#	Timestamp	Event	Origin	Storage Key	Serv...	Instance ID
1.	2025-04-...	Registered ...	http://local...	http://local...	/	sync-notes

No metadata for this event

```
> navigator.serviceWorker.ready.then(reg => reg.sync.register('sync-notes));
✖ Uncaught SyntaxError: Invalid or unexpected token                VM606:1
> navigator.serviceWorker.ready.then(reg => reg.sync.register('sync-notes'));
<  ▸ Promise {<pending>}
> navigator.serviceWorker.ready.then(reg => reg.sync.register('sync-notes'));
<  ▸ Promise {<pending>}
```

```
);
});
```

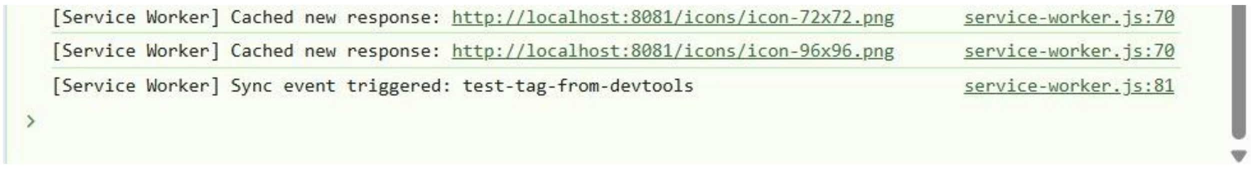
2. sync Event – Background Sync

What it does:
 Runs tasks in the background (when internet reconnects), like syncing offline-created data with the server.

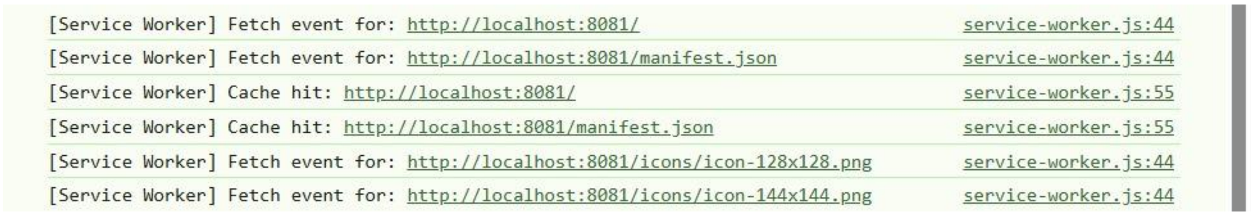
Use case in app:
 After creating or editing a note offline, sync it with the backend when the user is online again.

```
self.addEventListener('sync', (event) => {
  if (event.tag === 'sync-notes') {
    console.log('[SW] Background sync triggered!');
    event.waitUntil(syncNotesWithServer());
  }
});

async function syncNotesWithServer() {
  console.log('[SW] Syncing notes to server...');
  // Your sync logic here
}
```



```
[Service Worker] Cached new response: http://localhost:8081/icons/icon-72x72.png      service-worker.js:70
[Service Worker] Cached new response: http://localhost:8081/icons/icon-96x96.png      service-worker.js:70
[Service Worker] Sync event triggered: test-tag-from-devtools                          service-worker.js:81
>
```



```
[Service Worker] Fetch event for: http://localhost:8081/                              service-worker.js:44
[Service Worker] Fetch event for: http://localhost:8081/manifest.json                 service-worker.js:44
[Service Worker] Cache hit: http://localhost:8081/                                    service-worker.js:55
[Service Worker] Cache hit: http://localhost:8081/manifest.json                       service-worker.js:55
[Service Worker] Fetch event for: http://localhost:8081/icons/icon-128x128.png        service-worker.js:44
[Service Worker] Fetch event for: http://localhost:8081/icons/icon-144x144.png        service-worker.js:44
```

3. push Event – Push Notifications

What it does:
 Handles incoming push messages from the server and shows notifications to the user.

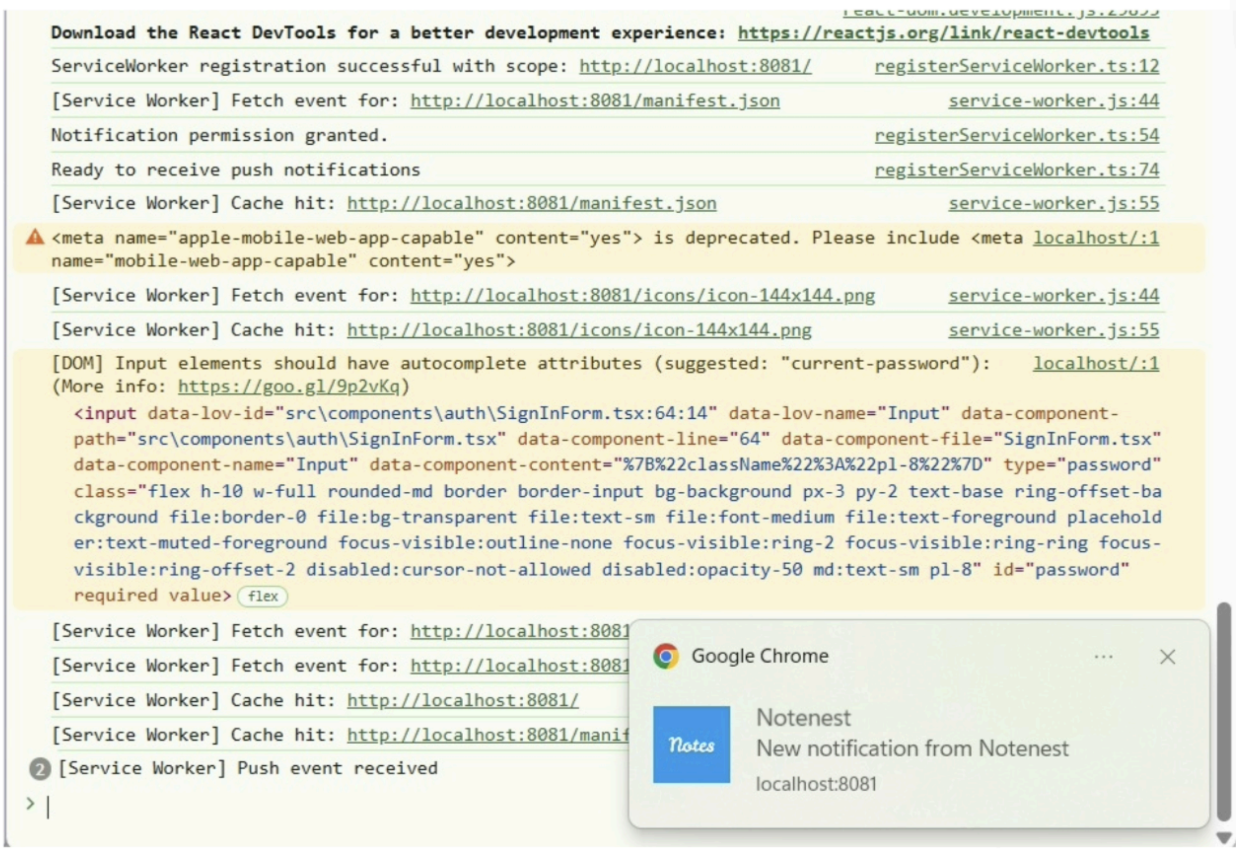
Use case in app:
 Send a push notification when a new note is shared or updated.

```
self.addEventListener('push', (event) => {
  console.log('[SW] Push received:', event);

  const data = event.data?.json() || {
    title: 'NoteNest',
    message: 'You got a new note update!',
  };

  const options = {
    body: data.message,
    icon: '/android-chrome-192x192.png',
    badge: '/android-chrome-192x192.png',
  };

  event.waitUntil(
    self.registration.showNotification(data.title, options)
  );
});
```



```
Download the React DevTools for a better development experience: https://reactjs.org/link/react-devtools
ServiceWorker registration successful with scope: http://localhost:8081/        registerServiceWorker.ts:12
[Service Worker] Fetch event for: http://localhost:8081/manifest.json                  service-worker.js:44
Notification permission granted.                                                registerServiceWorker.ts:54
Ready to receive push notifications                                             registerServiceWorker.ts:74
[Service Worker] Cache hit: http://localhost:8081/manifest.json                        service-worker.js:55
<meta name="apple-mobile-web-app-capable" content="yes"> is deprecated. Please include <meta localhost/:1
name="mobile-web-app-capable" content="yes">
[Service Worker] Fetch event for: http://localhost:8081/icons/icon-144x144.png         service-worker.js:44
[Service Worker] Cache hit: http://localhost:8081/icons/icon-144x144.png               service-worker.js:55
[DOM] Input elements should have autocomplete attributes (suggested: "current-password"):     localhost/:1
(More info: https://goo.gl/9p2vKq)
  <input data-lov-id="src\components\auth\SignInForm.tsx:64:14" data-lov-name="Input" data-component-
  path="src\components\auth\SignInForm.tsx" data-component-line="64" data-component-file="SignInForm.tsx"
  data-component-name="Input" data-component-content="%7B%22className%22%3A%22pl-8%22%7D" type="password"
  class="flex h-10 w-full rounded-md border border-input bg-background px-3 py-2 text-base ring-offset-ba
  ckground file:border-0 file:bg-transparent file:text-sm file:font-medium file:text-foreground placehold
  er:text-muted-foreground focus-visible:outline-none focus-visible:ring-2 focus-visible:ring-ring focus-
  visible:ring-offset-2 disabled:cursor-not-allowed disabled:opacity-50 md:text-sm pl-8" id="password"
  required value> flex
[Service Worker] Fetch event for: http://localhost:8081
[Service Worker] Fetch event for: http://localhost:8081
[Service Worker] Cache hit: http://localhost:8081/
[Service Worker] Cache hit: http://localhost:8081/manif
2 [Service Worker] Push event received
>
```