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

Theory

Progressive Web Apps (PWAs) use service workers to enhance user experience by enabling features like offline support, background syncing, and push notifications. In this experiment, we focused on implementing three major service worker events:

1. Fetch Event (Offline Support)
The fetch event allows the service worker to intercept network requests made by the app. In our implementation:

- We first try to load files from the cache.
- If the file is not in cache, we fetch it from the network.
- If the network is unavailable, we show a custom offline.html page. This improves the user experience during poor or no internet connection.

2. Sync Event (Background Sync)
The sync event is used to handle background tasks when the device is back online. In our code:

- We register a background sync with the tag "sync-data".
- When triggered, it can be used to sync data like user cart, orders, etc., to the server. This helps in saving user actions even when they're offline and syncing them later.

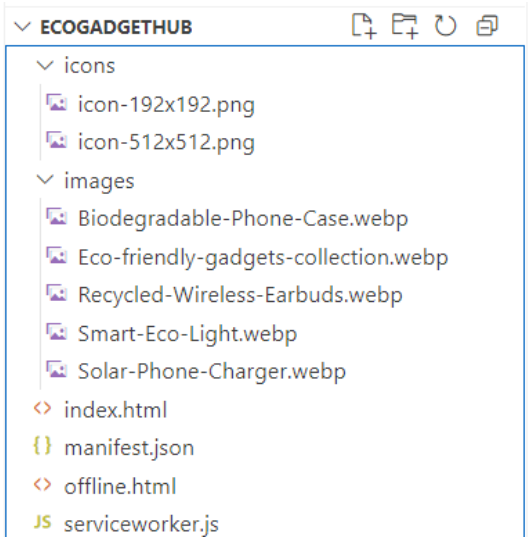
3. Push Event (Push Notifications)
The push event allows the PWA to receive messages even when the app is not open. In our project:

- We ask the user for notification permission.
- When a push message arrives, a notification is displayed to the user with custom content. This keeps users engaged with updates like offers, product arrivals, etc.

What We Did in Our E-commerce PWA:

- Registered a service worker in index.html.
- Handled caching and fallback in fetch for offline use.
- Registered sync-data to simulate background sync.
- Managed push events to show product-related notifications.
- Created a user-friendly offline.html page for no-network cases.

Folder Structure



index.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>EcoGadgetHub – Sustainable Tech Gadgets</title>
  <link rel="manifest" href="manifest.json">
  </style>
 </head>
 <body>
  <script>
   if ("serviceWorker" in navigator) {
    window.addEventListener("load", () => {
     navigator.serviceWorker
      .register("/serviceworker.js")
      .then((registration) => {
       console.log(
        "✅ Service Worker registered! Scope:",
        registration.scope
       );

       // 🔔 Request Push Notification Permission
       if ("PushManager" in window) {
        Notification.requestPermission().then((permission) => {
         if (permission === "granted") {
          console.log("🔔 Push notifications granted.");
         } else {
```

```
            console.log("🔕 Push notifications denied.");
          }
        });
      }

      // 🔄 Register Background Sync
      if ("SyncManager" in window) {
        navigator.serviceWorker.ready.then((swReg) => {
          swReg.sync
            .register("sync-data")
            .then(() => {
              console.log("🔄 Sync registered");
            })
            .catch((err) => {
              console.log("❌ Sync registration failed:", err);
            });
        });
      }
    })
    .catch((error) => {
      console.log("❌ Service Worker registration failed:", error);
    });
  });
}
</script>
</body>
</html>
```

offline.html

```
<!-- offline.html -->
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8" />
 <meta name="viewport" content="width=device-width, initial-scale=1.0" />
 <title>Offline</title>
 <style>
  body {
   font-family: Arial;
   text-align: center;
```

```
    padding: 50px;
    background: #f5f5f5;
   }
  </style>
</head>
<body>
  <h2>You are offline</h2>
  <p>Please check your internet connection and try again.</p>
</body>
</html>
```

serviceworker.js

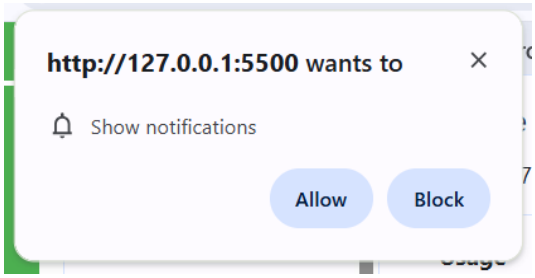
```
const CACHE_NAME = 'eco-gadget-hub-v1';
const urlsToCache = [
  '/',
  '/index.html',
  'icons/icon-192x192.png',
  'icons/icon-512x512.png',
  'images/Biodegradable-Phone-Case.webp',
  'images/Eco-friendly-gadgets-collection.webp',
  'images/Recycled-Wireless-Earbuds.webp',
  'images/Smart-Eco-Light.webp',
  'images/Solar-Phone-Charger.webp',
  'offline.html'
];

// Install Event
self.addEventListener("install", (event) => {
  console.log("[ServiceWorker] Install");
  event.waitUntil(
    caches.open(CACHE_NAME).then((cache) => {
      console.log("[ServiceWorker] Caching files");
      return cache.addAll(urlsToCache);
    })
  );
});

// Activate Event
self.addEventListener("activate", (event) => {
  console.log("[ServiceWorker] Activate");
  event.waitUntil(
```

```
  caches.keys().then((keyList) =>
   Promise.all(
    keyList.map((key) => {
     if (key !== CACHE_NAME) {
      console.log("[ServiceWorker] Removing old cache", key);
      return caches.delete(key);
     }
    })
   )
  )
 );
 return self.clients.claim();
});


// Enhanced Fetch Event
self.addEventListener("fetch", (event) => {
 console.log("[ServiceWorker] Fetch", event.request.url);
 const requestURL = new URL(event.request.url);


 // If request is same-origin, use Cache First
 if (requestURL.origin === location.origin) {
  event.respondWith(
   caches.match(event.request).then((cachedResponse) => {
    return ( cachedResponse ||
     fetch(event.request).catch(() => caches.match("offline.html"))
    );
   })
  );
 } else {
  // Else, use Network First
  event.respondWith(
   fetch(event.request)
    .then((response) => {
     return response;
    })
    .catch(() =>
     caches.match(event.request).then((res) => {
      return res || caches.match("offline.html");
     })
```

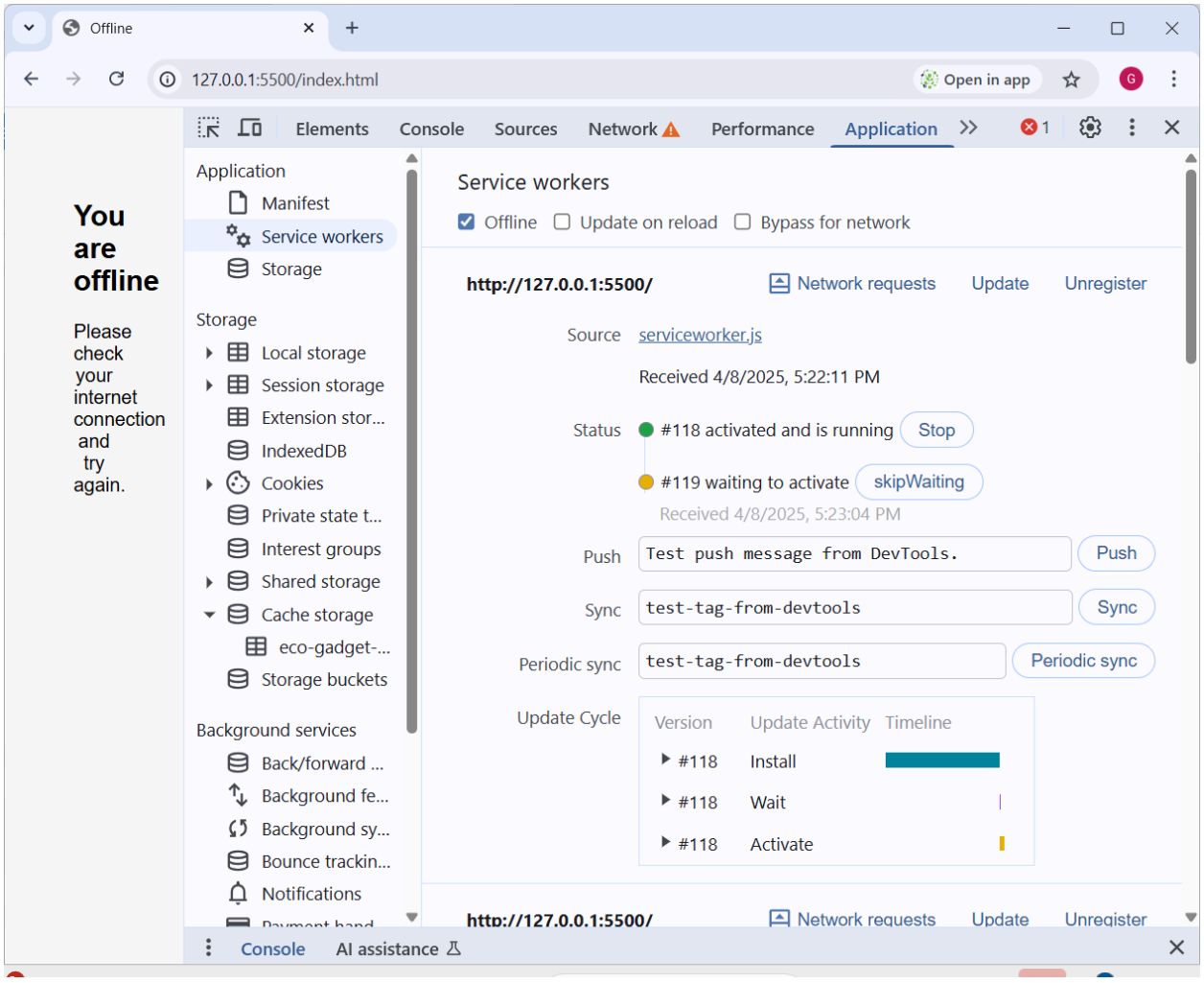
```
      )
    );
  }
});


// Sync Event (simulation)
self.addEventListener("sync", (event) => {
  if (event.tag === "sync-data") {
    event.waitUntil(
      (async () => {
        console.log("Sync event triggered: 'sync-data'");
        // Here you can sync data with server when online
      })()
    );
  }
});


// Push Event
self.addEventListener("push", function (event) {
  if (event && event.data) {
    let data = {};
    try {
      data = event.data.json();
    } catch (e) {
      data = {
        method: "pushMessage",
        message: event.data.text(),
      };
    }


    if (data.method === "pushMessage") {
      console.log("Push notification sent");
      event.waitUntil(
        self.registration.showNotification("Eco Friendly Gadgets Collection", {
          body: data.message,
        })
      );
    }
```

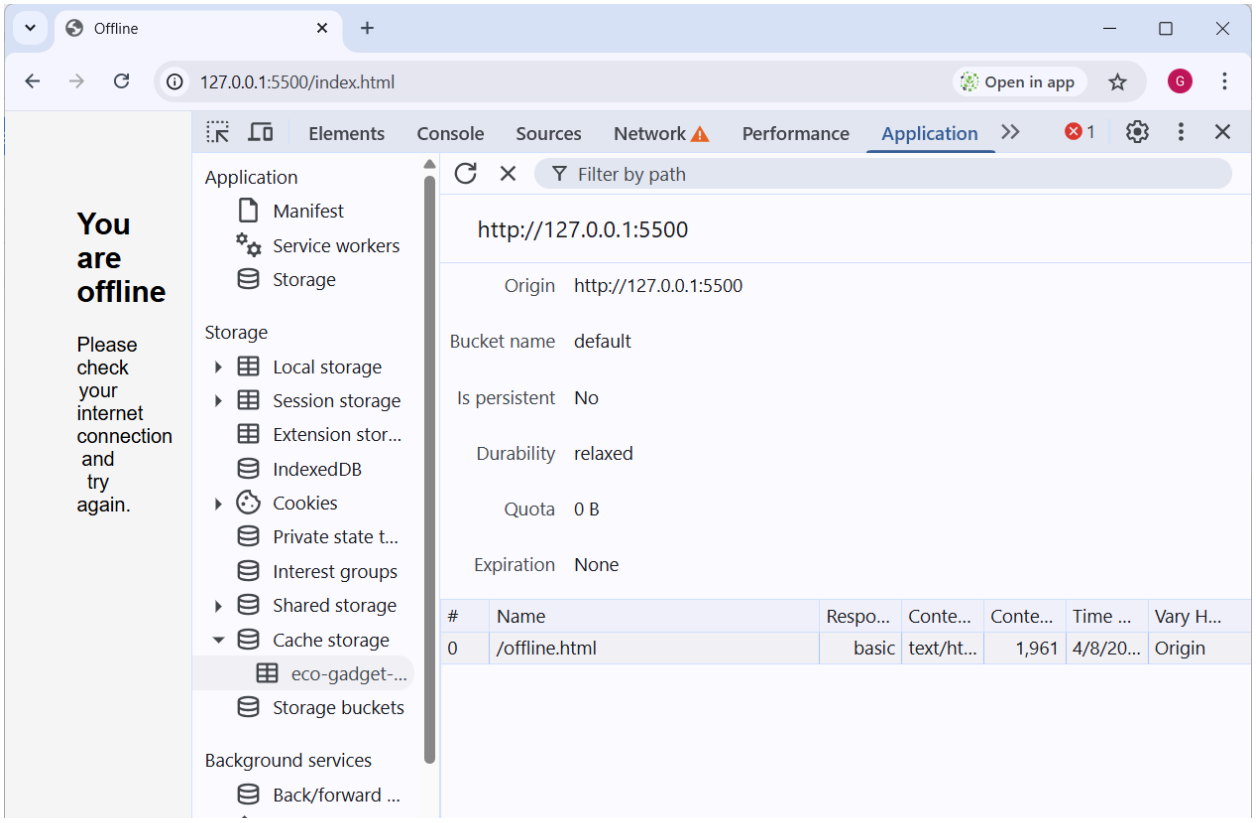
```
  }
});
```

Screenshot

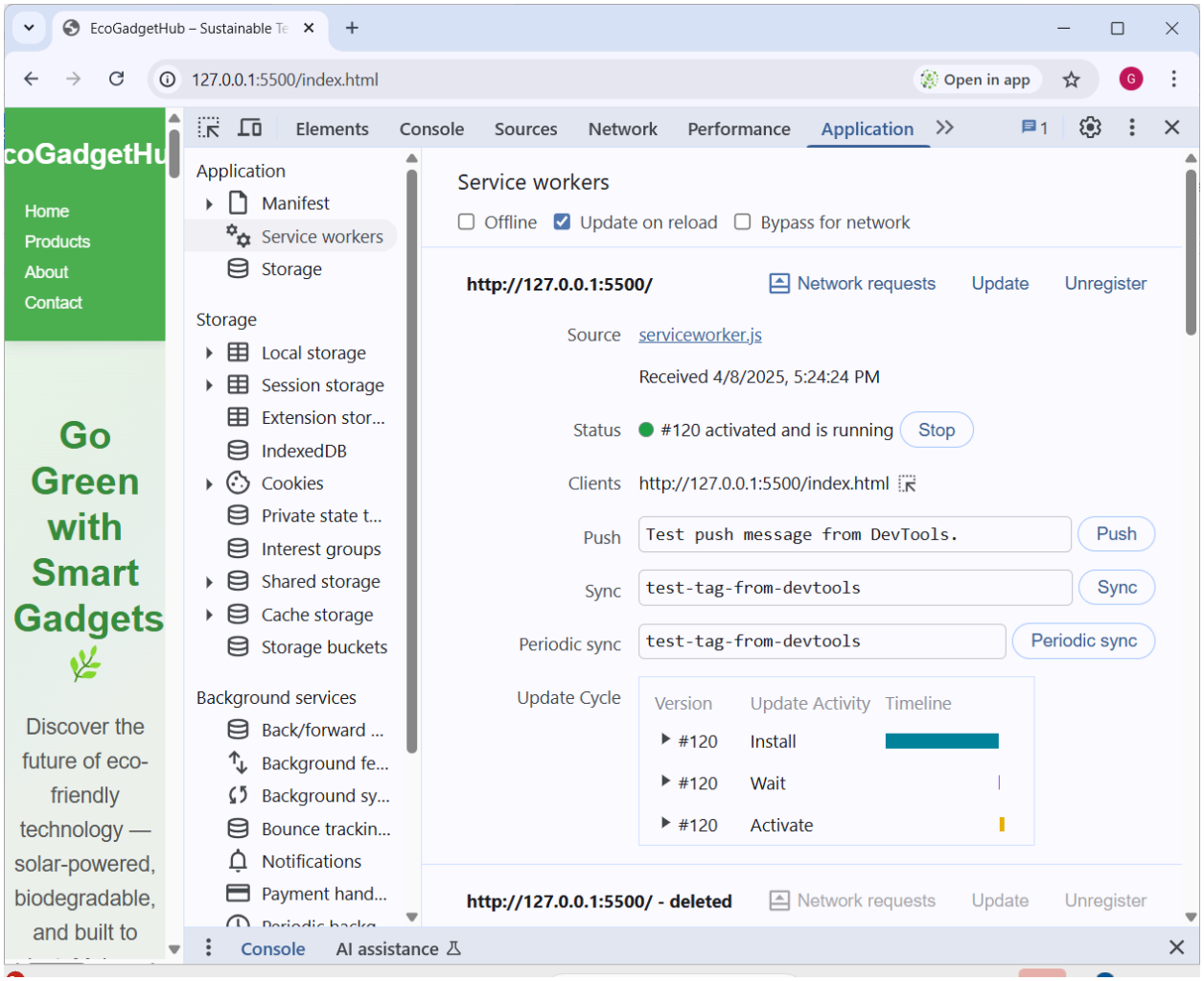


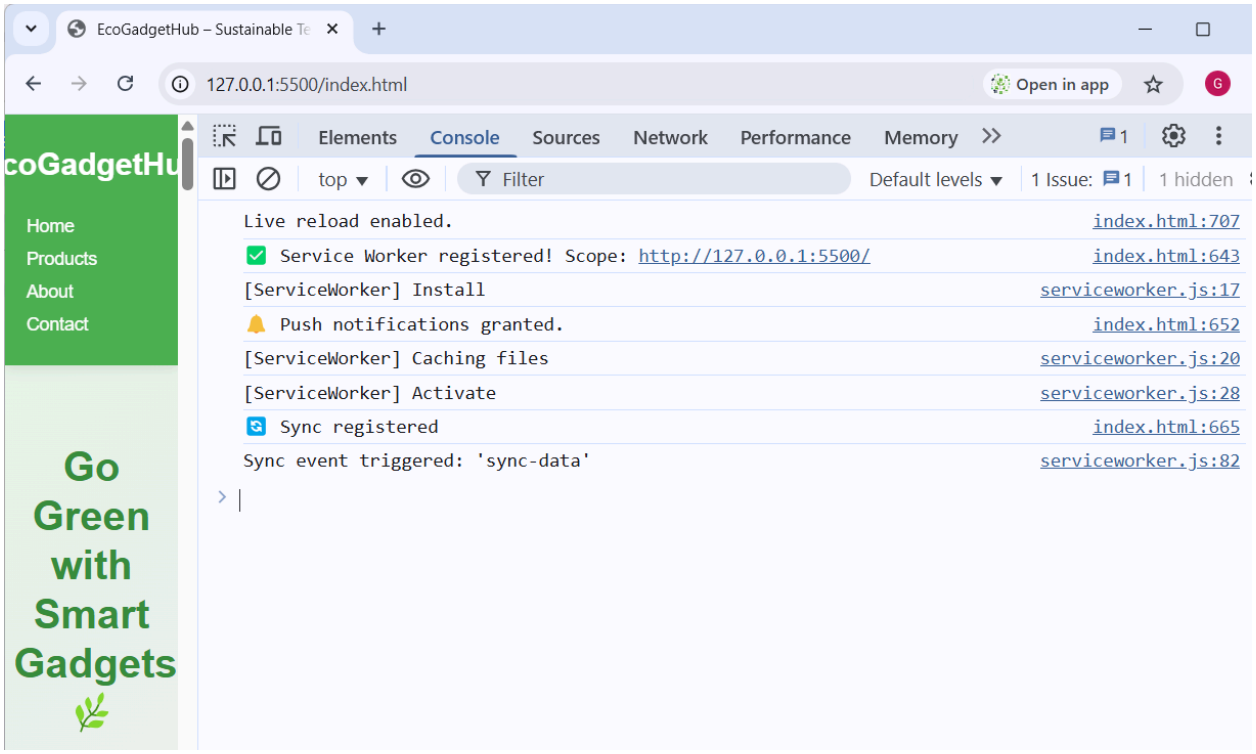
1. Test: Fetch Event (Offline Support)



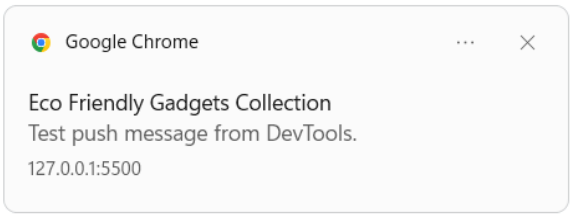


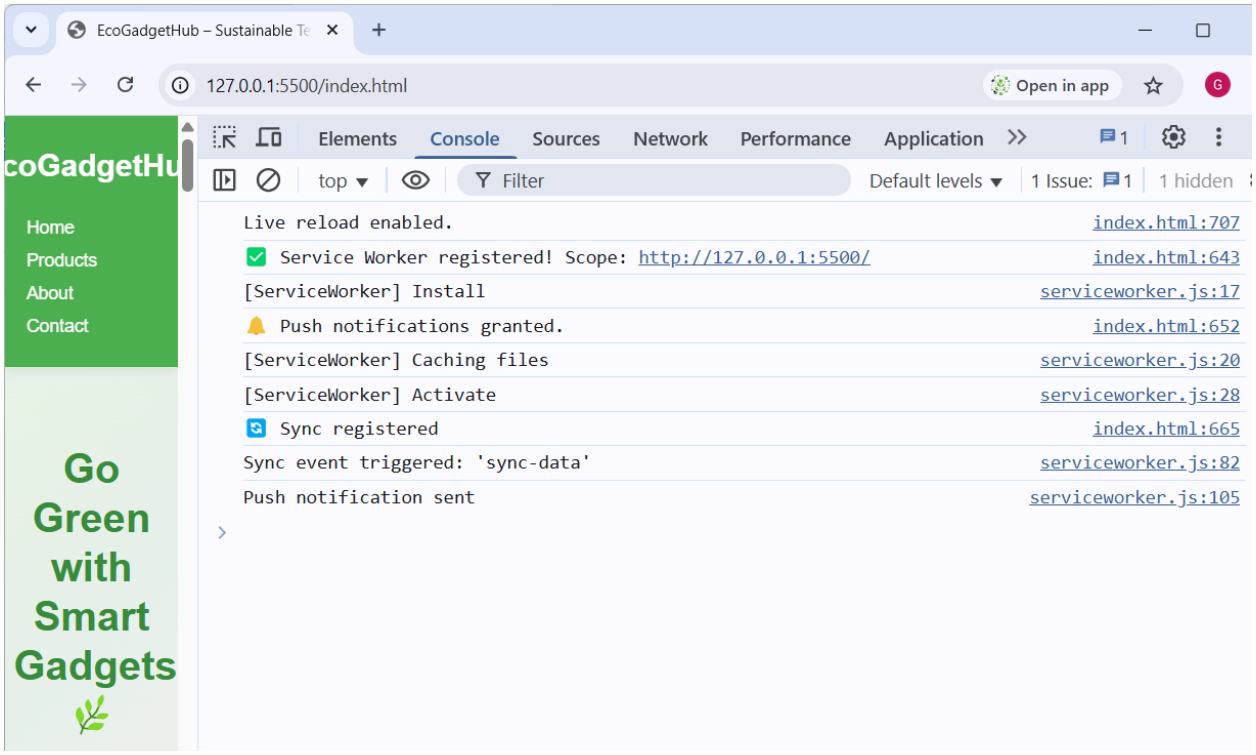
2. Test: Background Sync Event





3. Test: Push Notification Event





Conclusion

In this experiment, we successfully implemented fetch, sync, and push events using service workers to enhance offline functionality, background syncing, and notifications for EcoGadget Hub. Initially, we faced a cache loading issue due to incorrect file paths, which we resolved by carefully updating the paths and ensuring all required files were listed in the cache.