Siddhant Bhoi D15B_07

Experiment No.9
MAD & PWA LAB

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

II. Theory:

Service-Worker:

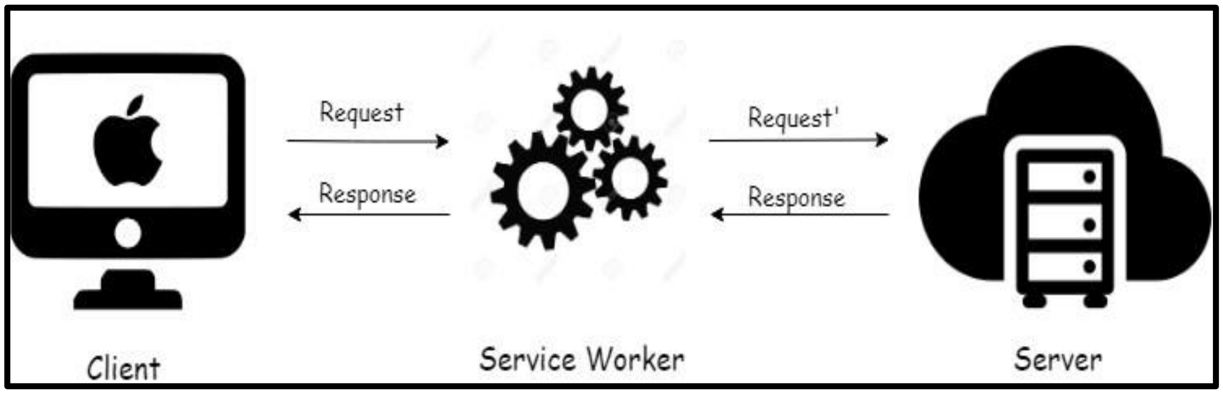
Service workers essentially act as proxy servers that sit between web applications, the browser, and the network (when available). They are intended, among other things, to enable the creation of effective offline experiences, intercept network requests and take appropriate action based on whether the network is available, and update assets residing on the server. They will also allow access to push notifications and background sync APIs.

1. Fetch Event:

• The fetch event is fired every time the service worker intercepts a network request made by the web page. It allows the service worker to intercept and modify network requests, including caching strategies and offline support.

• Fetch event is commonly used to implement caching strategies like Cache First or Network First, allowing the service worker to serve assets from the cache when offline or when the network is slow.

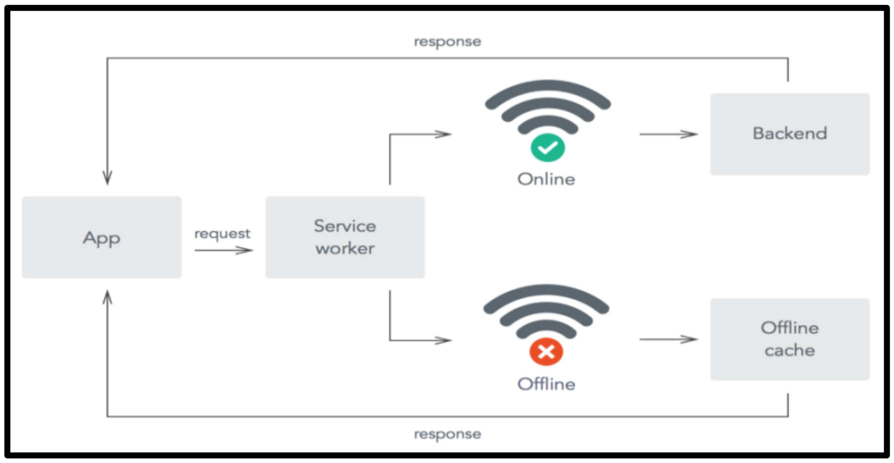
• Example: In the provided service worker code, the fetch event is used to cache fetched files in the 'v1' cache and serve them from the cache if available.



2. Sync Event:

• The sync event is fired when the browser requests a background sync with the service worker. Background sync allows the service worker to perform tasks, such as sending data to a server, even when the app is not actively in use or when the device is offline.

• Sync event is useful for tasks that require periodic updates or data synchronization, such as syncing offline data with a server once the network is available.

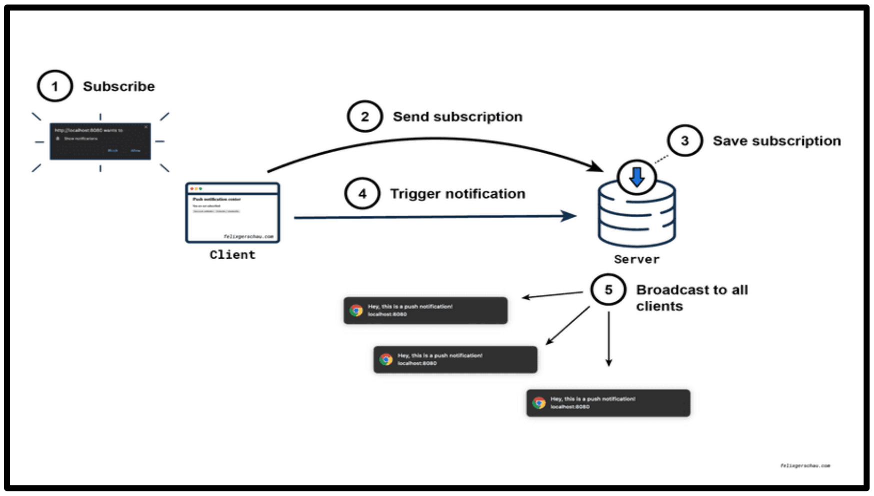


3. Push Event:

• The push event is fired when a push notification is received by the service worker from a push service. It allows the service worker to handle the push notification, including displaying a notification to the user or performing background tasks.

• Push event is commonly used to display notifications to users based on certain triggers, such as new messages or updates.

• In the provided service worker code, the push event is included to handle push notifications. It checks if the push message method is "pushMessage" and then displays a notification with the message content.


1 Subscribe
2 Send subscription
3 Save subscription
4 Trigger notification
Client
Server
5 Broadcast to all clients
Hey, this is a push notification! localhost:8080
Hey, this is a push notification! localhost:8080
Hey, this is a push notification! localhost:8080
felixgerschau.com

Service-Worker.js:

```
self.addEventListener('install', function(event) {
  console.log('Service Worker: Installed');
});

self.addEventListener('activate', function(event) {
  console.log('Service Worker: Activated');
});
```

Fetch Event

```
self.addEventListener('fetch', function(event) {
  console.log('Service Worker: Fetching');
  event.respondWith(
    fetch(event.request)
      .then(function(response) {
        if (!response || response.status !== 200 || response.type !== 'basic') {
          return response;
        }
        var responseToCache = response.clone();
        caches.open('v1').then(function(cache) {
          cache.put(event.request, responseToCache);
        });
        return response;
      })
      .catch(function(err) {
        console.error('Service Worker: Error fetching and caching new data:', err);
      })
  );
});
```

Push Event

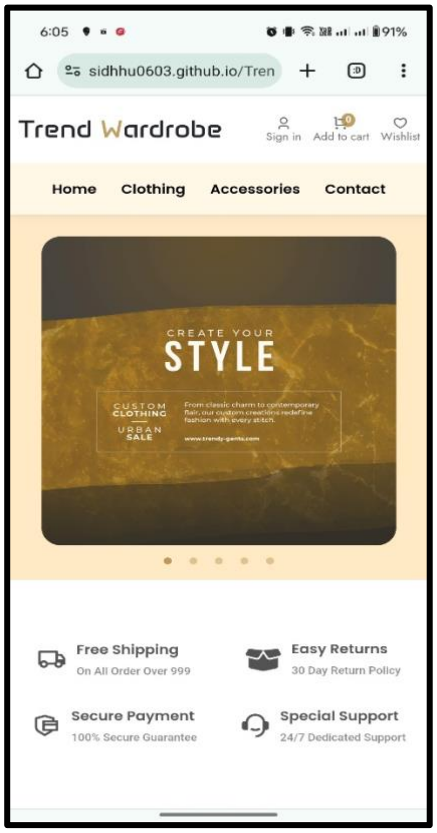
```
self.addEventListener('push', function(event) {
   console.log('Service Worker: Push received');
   // Handle push notification
   const title = 'Push Notification';
   const options = {
      body: 'This is a push notification',
      icon: 'path_to_icon.png'
   };
   event.waitUntil(
      self.registration.showNotification(title, options)
   );
});
```

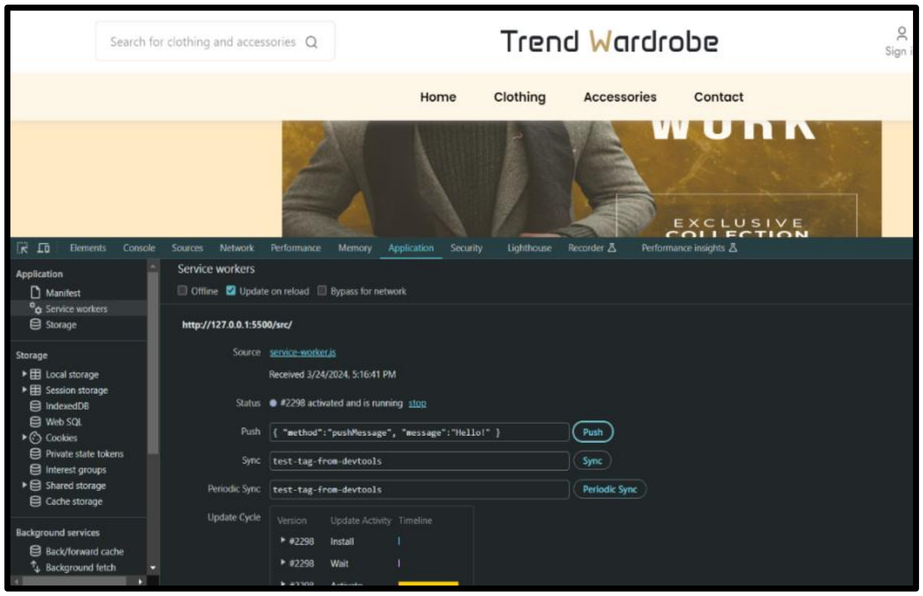
Sync event:

```
self.addEventListener('sync', function(event) {
   console.log('Service Worker: Sync event triggered');
   // Handle sync event
   if (event.tag === 'sync-example') {
      event.waitUntil(doSync());
   }
});
function doSync() {
   // Perform sync operation
   console.log('Service Worker: Syncing data');
}
```

Siddhant Bhoi D15B_07

III. OUTPUT:





Service workers

Offline [✓ Update on reload] Bypass for network

http://127.0.0.1:5500/src/

Source service-worker.js

Received 3/24/2024, 5:16:41 PM

Status ● #2298 activated and is running stop

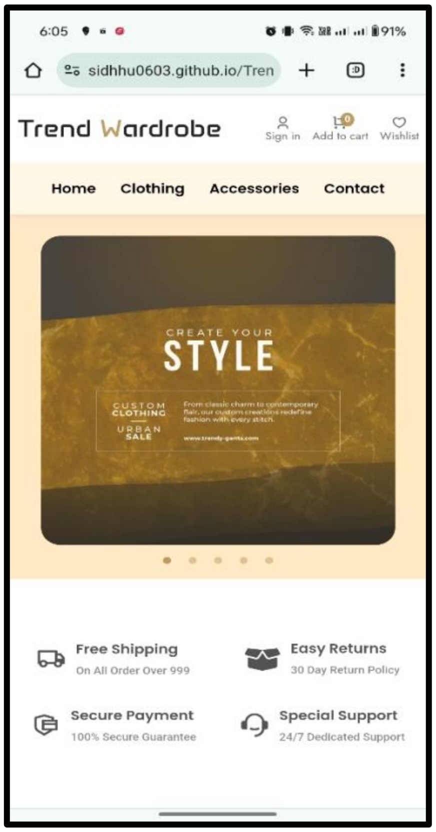
Push `{ "method":"pushMessage", "message":"Hello!" }` Push

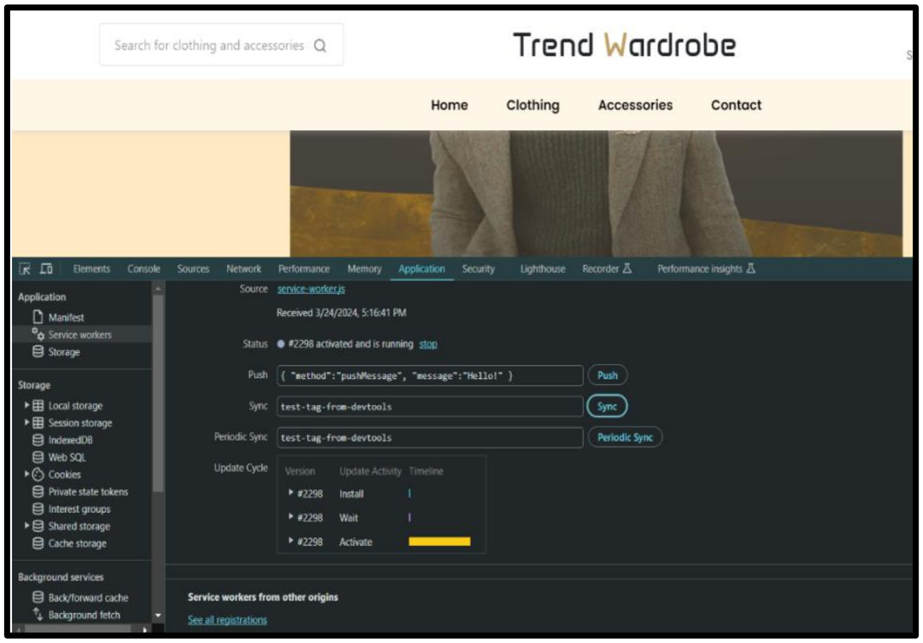
Sync `test-tag-from-devtools` Sync

Periodic Sync `test-tag-from-devtools` Periodic Sync

Update Cycle

Version	Update Activity	Timeline
#2298	Install	
#2298	Wait	





Source service-worker.js

Received 3/24/2024, 5:16:41 PM

Status ● #2298 activated and is running stop

Push `{ "method":"pushMessage", "message":"Hello!" }` Push

Sync `test-tag-from-devtools` Sync

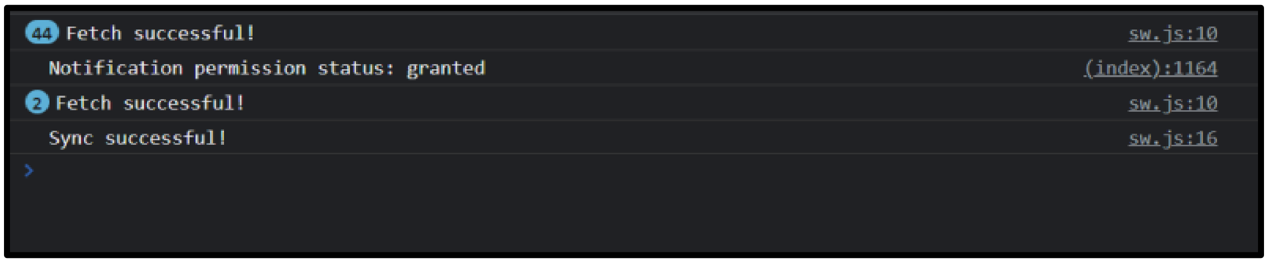
Periodic Sync `test-tag-from-devtools` Periodic Sync

Update Cycle

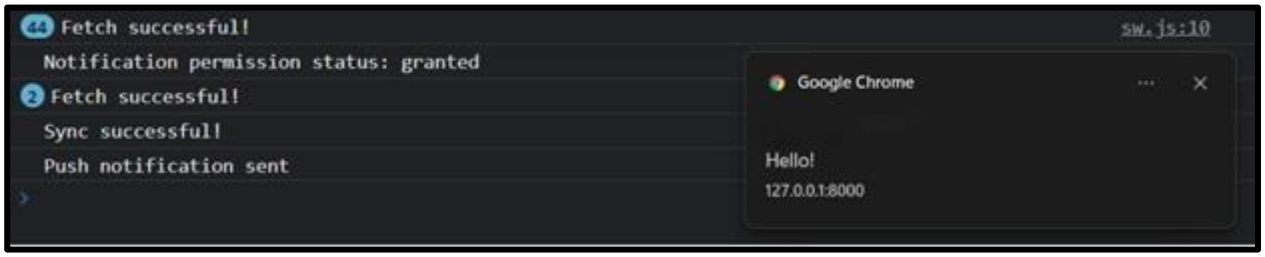
Version	Update Activity	Timeline
#2298	Install	
#2298	Wait	
#2298	Activate	

Service workers from other origins

See all registrations



```
44 Fetch successful!                                         sw.js:10
   Notification permission status: granted               (index):1164
 2 Fetch successful!                                         sw.js:10
   Sync successful!                                          sw.js:16
>
```



```
44 Fetch successful!                                         sw.js:10
   Notification permission status: granted
 2 Fetch successful!
   Sync successful!
   Push notification sent
>
```

IV. <u>CONCLUSION:</u>

Hence, successfully implemented service worker events like fetch, sync, and push for the E-commerce Progressive Web App (PWA), enabling advanced features such as offline caching, background synchronization, and push notifications. These enhancements enhance user experience and provide robust functionality, even in challenging network conditions.