

MPL Experiment 8 (PWA)

Name: Devansh Wadhwani - 64
Kunal Punjabi - 43
Manav Punjabi - 44
Hitesh Rohra - 46

Class: D15A

Aim: To code and register a service worker, and complete the install and activation process for a new service worker for the Food App PWA.

Theory:

Service Worker

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

- A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
- Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
- The service worker becomes idle when not in use and restarts when it's next needed. You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.

What can we do with Service Workers?

- You can manage all network traffic of the page and do any manipulations. For example, when the page requests a CSS file, you can send plain text as a response or when the page requests an HTML file, you can send a png file as a response. You can also send a true response too.
- You can Cache
You can cache any request/response pair with Service Worker and Cache API and you can access these offline content anytime.
- You can manage Push Notifications
You can manage push notifications with Service Worker and show any information message to the user.
- You can Continue
Although Internet connection is broken, you can start any process with Background Sync of Service Worker.

What can't we do with Service Workers?

- You can't access the Window
You can't access the window, therefore, You can't manipulate DOM elements. But, you can communicate to the window through post Message and manage processes that you want.
- You can't work it on 80 Port
Service Worker just can work on HTTPS protocol. But you can work on localhost during development.

Codes:

//Serviceworker.js

```
const CACHE_NAME = 'delice-cache-v1';
const urlsToCache = [
  './',
  './index.html',
  './styles.css',
  './script.js',
  './icon-192x192.png',
  './icon-512x512.png'
];

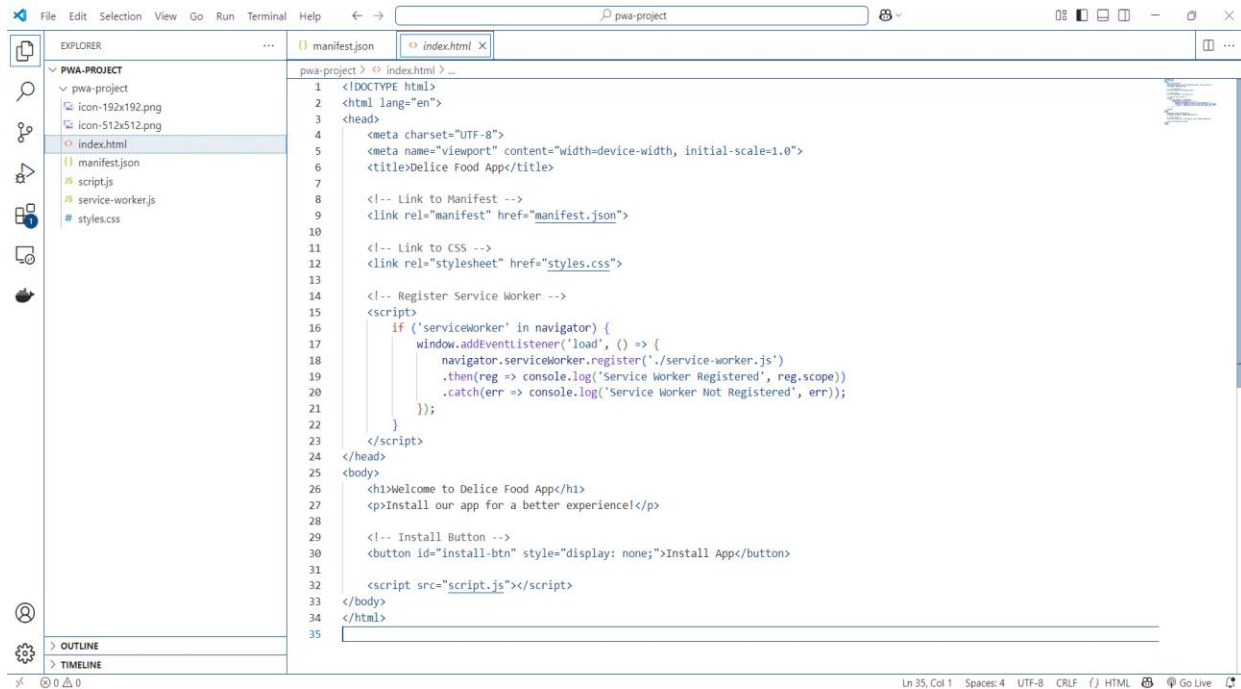
// Install Service Worker
self.addEventListener('install', event => {
  event.waitUntil(
    caches.open(CACHE_NAME).then(cache => {
      console.log('Opened cache');
      return cache.addAll(urlsToCache);
    })
  );
});

// Activate Service Worker (Old cache cleanup)
self.addEventListener('activate', event => {
  event.waitUntil(
    caches.keys().then(cacheNames => {
      return Promise.all(
        cacheNames.filter(cache => cache !==
CACHE_NAME).map(cache => caches.delete(cache))
      );
    })
  );
});

// Fetch Requests with Network Fallback
self.addEventListener('fetch', event => {
  event.respondWith(
    caches.match(event.request).then(response => {
      return response || fetch(event.request)
        .then(networkResponse => {
          return caches.open(CACHE_NAME).then(cache => {
            cache.put(event.request,
networkResponse.clone());
          });
        })
    })
  );
});
```

```
        return networkResponse;
    });
    });
}).catch(() => {
    return caches.match('./index.html'); // Offline fallback
})
);
});
```

Output:



The screenshot shows the VS Code editor with a project named 'pwa-project'. The Explorer sidebar on the left shows the file structure: 'pwa-project' folder containing 'icon-192x192.png', 'icon-512x512.png', 'index.html', 'manifest.json', 'script.js', 'service-worker.js', and 'styles.css'. The 'index.html' file is selected and its content is displayed in the main editor. The code is a standard HTML5 boilerplate for a Progressive Web App, including a meta viewport tag, a title 'Delice Food App', links to 'manifest.json' and 'styles.css', and a script to register a service worker. The body contains a welcome message, an installation prompt, and an 'Install App' button.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Delice Food App</title>
7
8   <!-- Link to Manifest -->
9   <link rel="manifest" href="manifest.json">
10
11   <!-- Link to CSS -->
12   <link rel="stylesheet" href="styles.css">
13
14   <!-- Register Service Worker -->
15   <script>
16     if ('serviceWorker' in navigator) {
17       window.addEventListener('load', () => {
18         navigator.serviceWorker.register('./service-worker.js')
19           .then(reg => console.log('Service Worker Registered', reg.scope))
20           .catch(err => console.log('Service Worker Not Registered', err));
21       });
22     }
23   </script>
24 </head>
25 <body>
26   <h1>Welcome to Delice Food App</h1>
27   <p>Install our app for a better experience!</p>
28
29   <!-- Install Button -->
30   <button id="install-btn" style="display: none;">Install App</button>
31
32   <script src="script.js"></script>
33 </body>
34 </html>
35
```

