Name : Shreyas Naik
Roll no : 38
Div : D15B

# MPL Experiment 8

AIM : To code and register a service worker, and complete the install and activation process for a new service worker

Code :

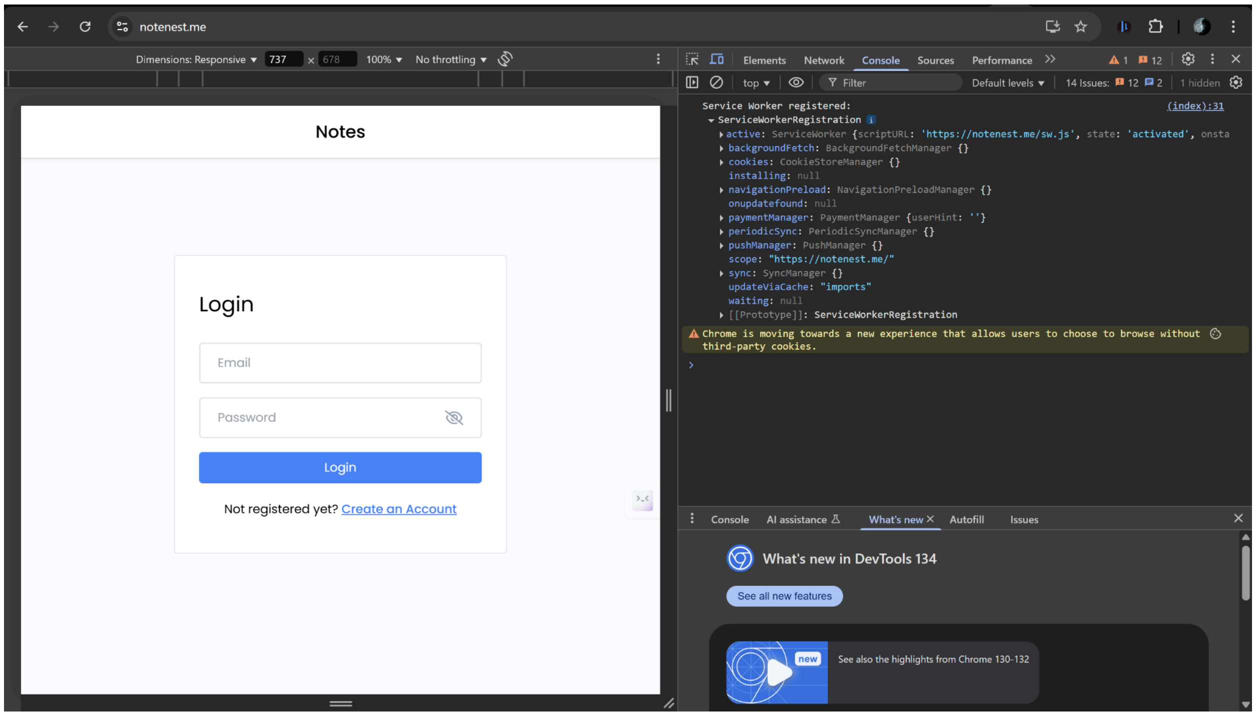
Add this in your ***index.html*** file

```
  <!-- Register Service Worker -->
  <script>
    if ('serviceWorker' in navigator) {
      navigator.serviceWorker.register('/sw.js').then((registration) => {
        console.log('Service Worker registered:', registration);
      }).catch((error) => {
        console.log('Service Worker registration failed:', error);
      });
    }
  </script>
```

***sw.js***

```
if (!self.define) {
  let registry = {};

  // Used for `eval` and `importScripts` where we can't get script URL by other means.
  // In both cases, it's safe to use a global var because those functions are synchronous.
  let nextDefineUri;

  const singleRequire = (uri, parentUri) => {
    uri = new URL(uri + ".js", parentUri).href;
    return registry[uri] || (

        new Promise(resolve => {
          if ("document" in self) {
            const script = document.createElement("script");
            script.src = uri;
            script.onload = resolve;
            document.head.appendChild(script);
          } else {
```

```
          nextDefineUri = uri;
          importScripts(uri);
          resolve();
        }
      })

    .then(() => {
      let promise = registry[uri];
      if (!promise) {
        throw new Error(`Module ${uri} didn’t register its module`);
      }
      return promise;
    })
  );
};

self.define = (depsNames, factory) => {
  const uri = nextDefineUri || ("document" in self ? document.currentScript.src : "") ||
location.href;
  if (registry[uri]) {
    // Module is already loading or loaded.
    return;
  }
  let exports = {};
  const require = depUri => singleRequire(depUri, uri);
  const specialDeps = {
    module: { uri },
    exports,
    require
  };
  registry[uri] = Promise.all(depsNames.map(
    depName => specialDeps[depName] || require(depName)
  )).then(deps => {
    factory(...deps);
    return exports;
  });
};
}
define(['./workbox-54d0af47'], (function (workbox) { 'use strict';

  self.skipWaiting();
  workbox.clientsClaim();

  /**
  * The precacheAndRoute() method efficiently caches and responds to
  * requests for URLs in the manifest.
  * See https://goo.gl/S9QRab
  */
 workbox.precacheAndRoute([{
   "url": "registerSW.js",
   "revision": "3ca0b8505b4bec776b69afdba2768812"
 }, {
   "url": "index.html",
   "revision": "0.dtaucfr1o8"
 }], {});
 workbox.cleanupOutdatedCaches();
 workbox.registerRoute(new
workbox.NavigationRoute(workbox.createHandlerBoundToURL("index.html"), {
   allowlist: [/^\/$/]
 }));

}));
```


notenest.me
Dimensions: Responsive 737 x 678 100% No throttling
Elements Network Console Sources Performance
top Filter Default levels 14 Issues: 12 2 1 hidden
Notes
Login
Email
Password
Login
Not registered yet? Create an Account
Console AI assistance What's new Autofill Issues
What's new in DevTools 134
See all new features
new
See also the highlights from Chrome 130-132

```
Service Worker registered:                                              (index):31
 ▾ ServiceWorkerRegistration
   ▸ active: ServiceWorker {scriptURL: 'https://notenest.me/sw.js', state: 'activated', onsta
   ▸ backgroundFetch: BackgroundFetchManager {}
   ▸ cookies: CookieStoreManager {}
     installing: null
   ▸ navigationPreload: NavigationPreloadManager {}
     onupdatefound: null
   ▸ paymentManager: PaymentManager {userHint: ''}
   ▸ periodicSync: PeriodicSyncManager {}
   ▸ pushManager: PushManager {}
     scope: "https://notenest.me/"
   ▸ sync: SyncManager {}
     updateViaCache: "imports"
     waiting: null
   ▸ [[Prototype]]: ServiceWorkerRegistration
⚠ Chrome is moving towards a new experience that allows users to choose to browse without
  third-party cookies.
>
```