MPL Practical 11

Aim: To use google Lighthouse PWA Analysis Tool to test the PWA functioning.

Theory:

Google Lighthouse is a powerful open-source tool built into Chrome DevTools that helps test the quality of a Progressive Web App (PWA). It evaluates key aspects like performance, accessibility, SEO, and best practices, and provides a score along with suggestions for improvement.

In this experiment, we tested the CampusCraze PWA using Lighthouse at my github deployed website. The results showed excellent performance with a Performance score of 99, Accessibility 95, Best Practices 96, and SEO 100. The service worker was detected, manifest was valid, and offline support was working correctly. Minor suggestions like improving contrast and caching policies were noted for future enhancement.

This analysis confirmed that our PWA implementation is highly optimized and provides a reliable, fast, and installable user experience.

---

sw.js

```
// CampusCraze Service Worker for PWA functionality
const CACHE_NAME = 'campuscraze-v1';
const ASSETS = [
  '/PWA-CAMPUS-CRAZE/',
  '/PWA-CAMPUS-CRAZE/index.html',
  '/PWA-CAMPUS-CRAZE/styles.css',
  '/PWA-CAMPUS-CRAZE/main.js',
  '/PWA-CAMPUS-CRAZE/manifest.json',
  '/PWA-CAMPUS-CRAZE/images/logo.svg',
  '/PWA-CAMPUS-CRAZE/images/college-vibes.webp',
  '/PWA-CAMPUS-CRAZE/images/stationery.webp',
  '/PWA-CAMPUS-CRAZE/images/decor.webp',
  '/PWA-CAMPUS-CRAZE/images/fashion.webp',
  '/PWA-CAMPUS-CRAZE/images/tech.webp',
  '/PWA-CAMPUS-CRAZE/images/study-bundle.webp',
  '/PWA-CAMPUS-CRAZE/images/decor-bundle.webp',
  '/PWA-CAMPUS-CRAZE/images/app-mockup.webp',
  '/PWA-CAMPUS-CRAZE/images/google-play.svg',
  '/PWA-CAMPUS-CRAZE/images/app-store.svg',
  '/PWA-CAMPUS-CRAZE/images/avatar-simran.webp',
  '/PWA-CAMPUS-CRAZE/images/avatar-rishi.webp',
  '/PWA-CAMPUS-CRAZE/images/avatar-priya.webp',
  '/PWA-CAMPUS-CRAZE/images/instagram.svg',
  '/PWA-CAMPUS-CRAZE/images/youtube.svg',
```

```
  '/PWA-CAMPUS-CRAZE/images/twitter.svg',
  '/PWA-CAMPUS-CRAZE/icons/icon-192x192.png',
  '/PWA-CAMPUS-CRAZE/icons/icon-512x512.png',
  '/PWA-CAMPUS-CRAZE/offline.html',
  'https://fonts.googleapis.com/css2?family=Montserrat:wght@400;500;700&display=swap'
];

// 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(ASSETS);
    })
  );
});

// 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"))
```

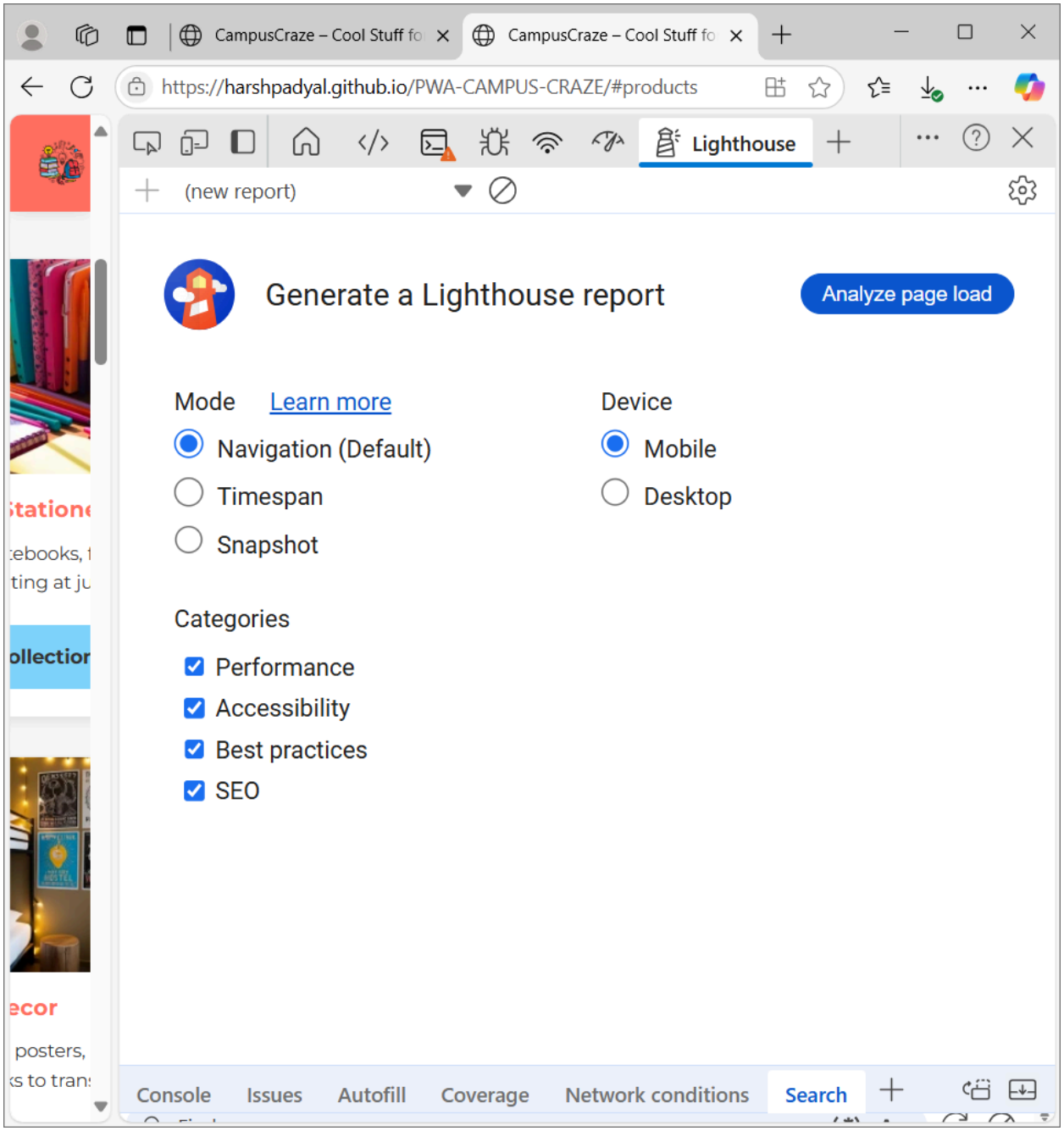
```
      );
    })
  );
 } else {
  // Else, use Network First
  event.respondWith(
   fetch(event.request)
    .then((response) => {
     return response;
    })
    .catch(() =>
     caches.match(event.request).then((res) => {
      return res || caches.match("offline.html");
     })
    )
  );
 }
});

// 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("CampusCraze - Cool Stuff for College Life", {
        body: data.message,
      })
    );
  }
 }
});
```



Conclusion

In this experiment, we used Google Lighthouse to analyze the PWA performance of CampusCraze and achieved near-perfect scores, confirming our implementation was correct. Initially, the service worker wasn't detected because of a wrong file path during GitHub Pages deployment, which we fixed by updating all asset links with the /PWA-CAMPUS-CRAZE/ prefix.