Practical 11

Name: Sharvari Kishor More
Class: D15B
Roll No.: 35

Aim:

To use google Lighthouse PWA Analysis Tool to test the PWA functioning

Theory:

Lighthouse is a tool provided by Google that helps us check how well our web app performs as a Progressive Web App (PWA). It analyzes important factors like performance, accessibility, SEO, best practices, and PWA features.

In this experiment, we used Lighthouse in Chrome DevTools to test our deployed PWA. Lighthouse runs a series of audits and gives a score out of 100 for each category. These scores show how well our app performs and where we can improve.

What we implemented:

- We deployed our PWA Festival-Combo on GitHub Pages.
- Then, we ran Lighthouse from Chrome DevTools on the live site.
- Lighthouse gave us excellent scores:
  Performance: 92
  Accessibility: 90
  Best Practices: 89
  SEO: 90

These high scores show that our app is fast, user-friendly, and optimized for search engines. Some minor suggestions were also given, like improving color contrast and adding a meta description, which can help make the app even better.

Code:
Manifest.json

```
{
  "name": "Festival Combo Shop",
  "short_name": "Festival Combos",
  "start_url": "./index.html",
  "display": "standalone",
  "background_color": "#f9fafb",
  "theme_color": "#ec4899",
  "description": "Celebrate with perfect festival combos for Diwali, Holi, Christmas, and more!",
  "icons": [
   {
```

```
      "src": "/icon-192.png",
      "sizes": "192x192",
      "type": "image/png",
      "purpose": "any-maskable"
    },
    {
      "src": "/icon-512.png",
      "sizes": "512x512",
      "type": "image/png",
      "purpose": "any-maskable"
    }
  ],
  "screenshots": [
    {
      "src": "/diwalisweets.webp",
      "type": "image/webp",
      "sizes": "1600x1000"
    },
    {
      "src": "/holicolors.webp",
      "type": "image/webp",
      "sizes": "1280x720"
    },
    {
      "src": "/christmas.webp",
      "type": "image/webp",
      "sizes": "1600x1069"
    }
  ]
}
```

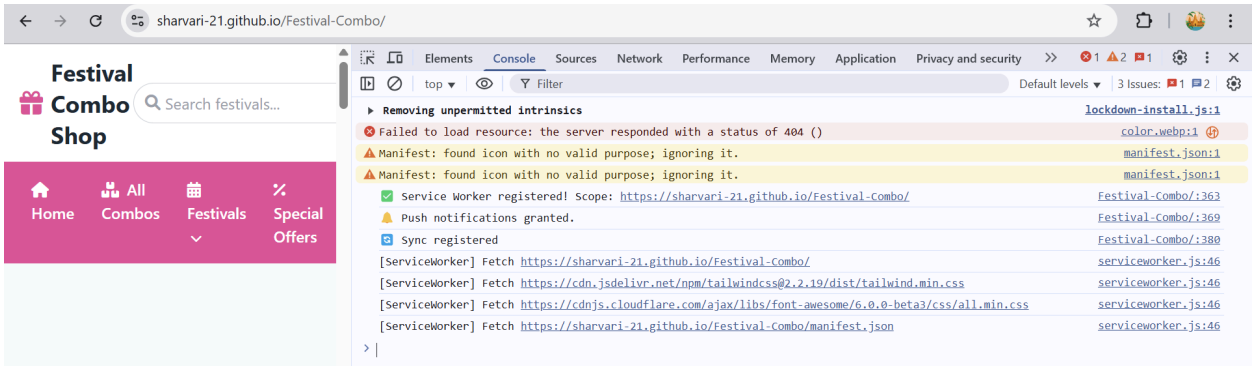
serverworker.js

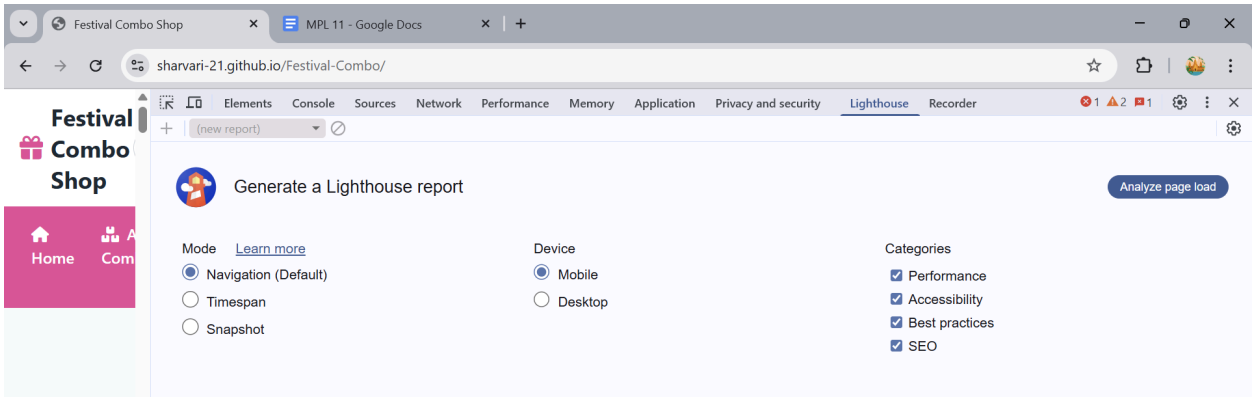
```
const CACHE_NAME = 'festival-combo-shop-v1';
const FILES_TO_CACHE = [
  '/Festival-Combo/',
  '/Festival-Combo/index.html',
  '/Festival-Combo/styles.css',
  '/Festival-Combo/manifest.json',
  '/Festival-Combo/serviceworker.js',
```

```
  '/Festival-Combo/color.webp',
  '/Festival-Combo/diwalisweets.webp',
  '/Festival-Combo/holicolors.webp',
  '/Festival-Combo/christmas.webp',
  ];
// 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(FILES_TO_CACHE);
      })
    );
  });

  // 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("Maharashtrian Handloom", {
          body: data.message,
        })
      );
    }
  }
});
```





Conclusion:

In this experiment, we tested the Festival Combo PWA using Google Lighthouse and achieved high scores in performance, accessibility, best practices, and SEO. Initially, we faced an error where all images weren't visible, which was resolved by adding the repository name into the serviceworker.js file before the image file.