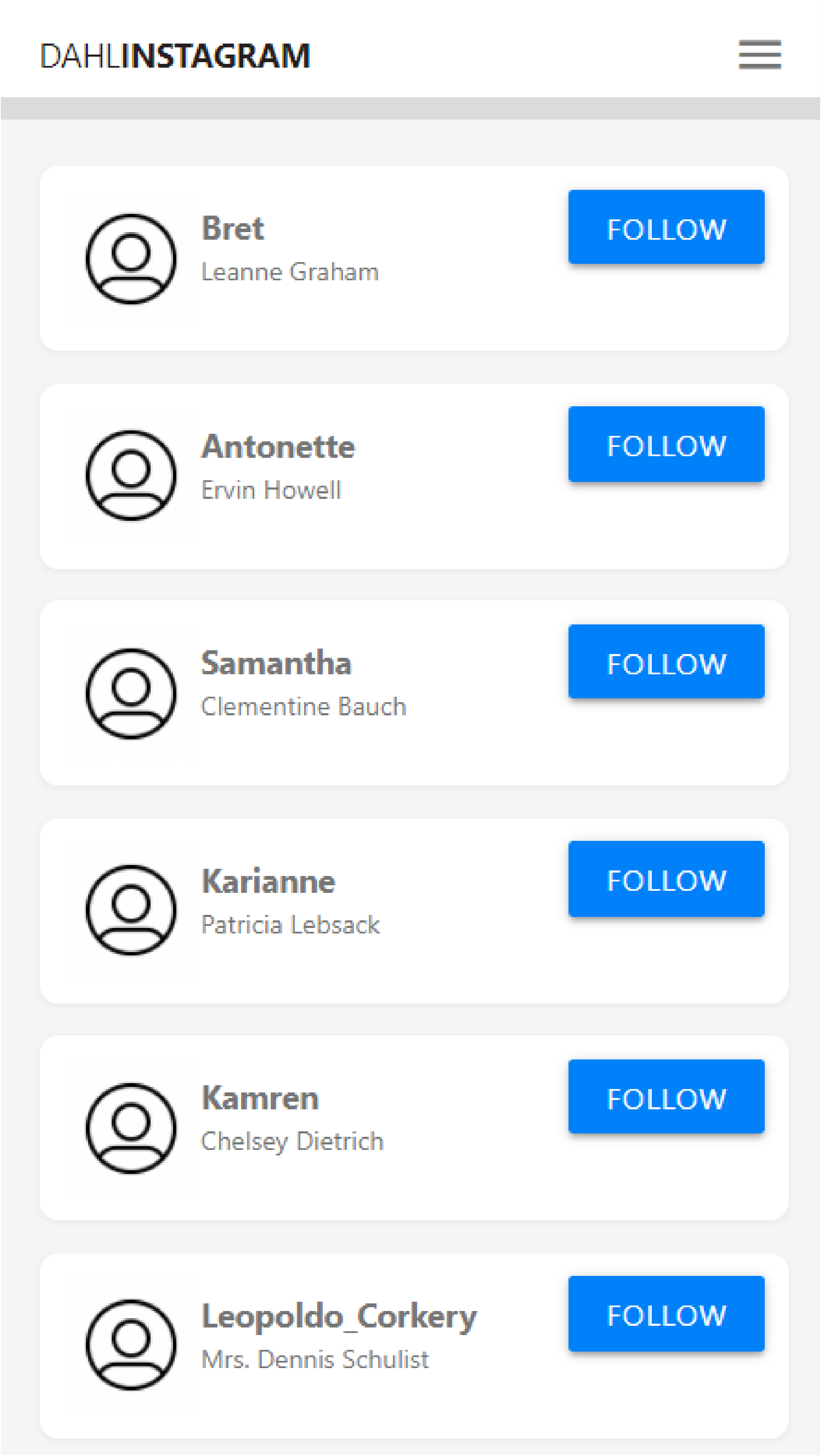
PROGRESSIVE WEB APP

DEMO



DAHLINSTAGRAM

Bret
Leanne Graham

Antonette
Ervin Howell

Samantha
Clementine Bauch

Karianne
Patricia Lebsack

Kamren
Chelsey Dietrich

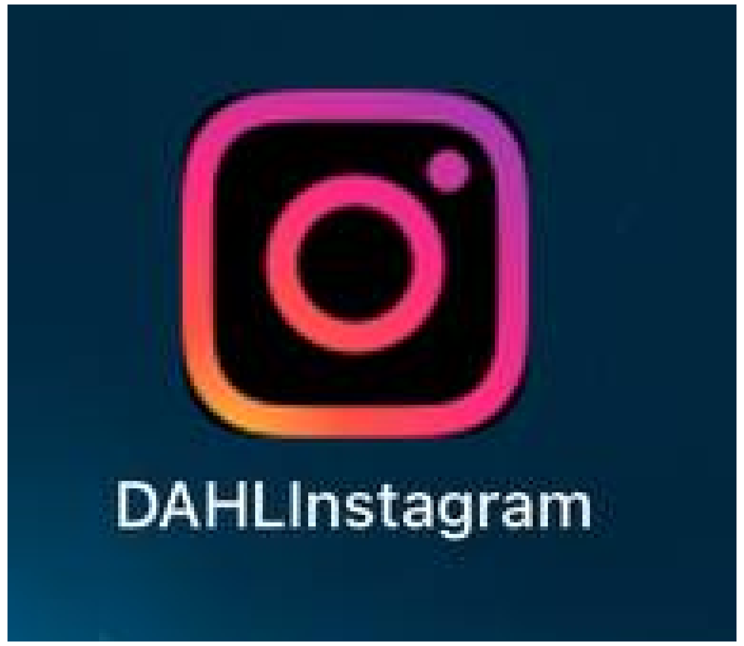
Leopoldo_Corkery
Mrs. Dennis Schulist

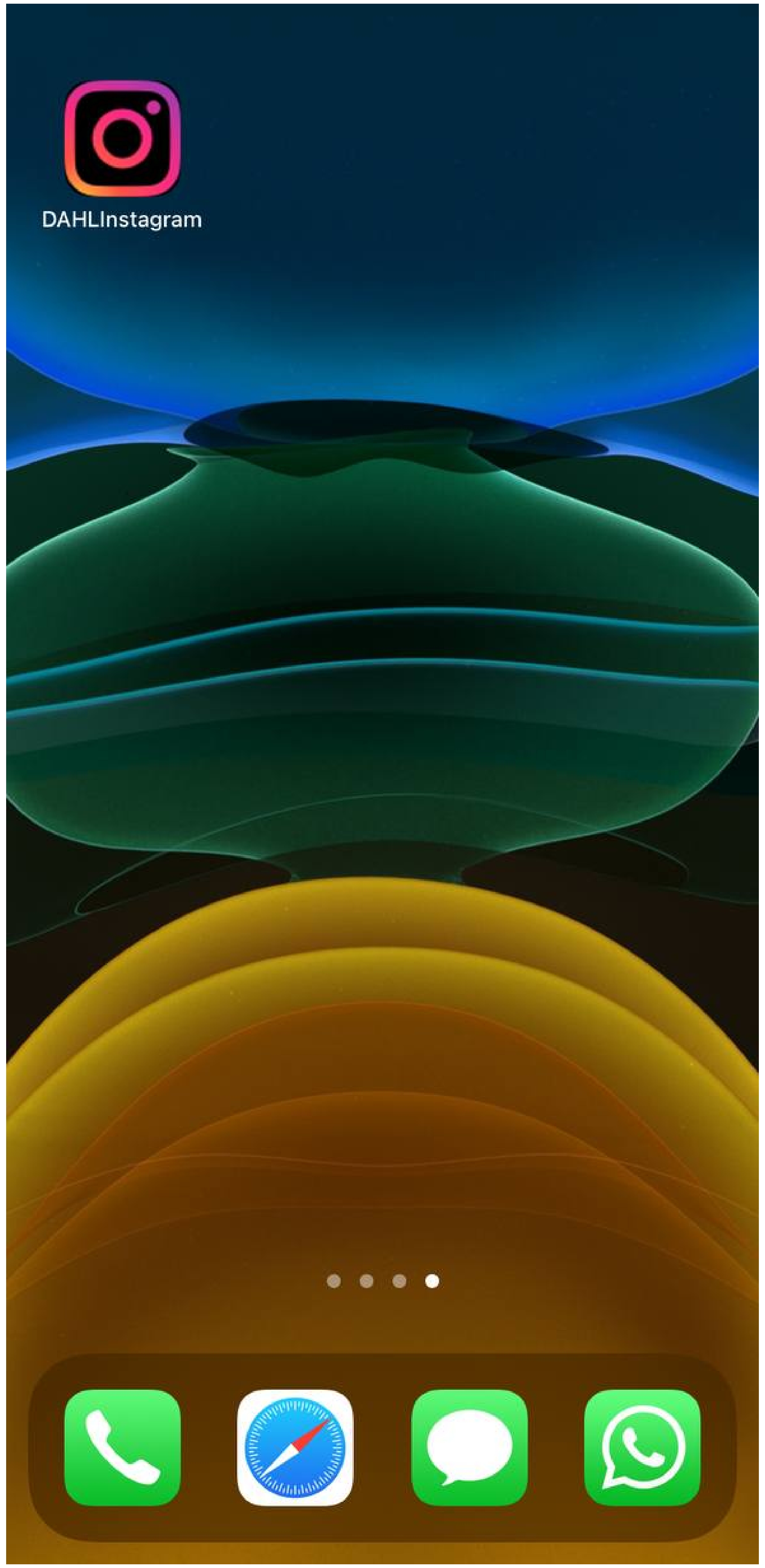
Objetivo

Construir una aplicación que sea capaz de funcionar offline, maneje notificaciones push y sea instalable en homescreen de un dispositivo.

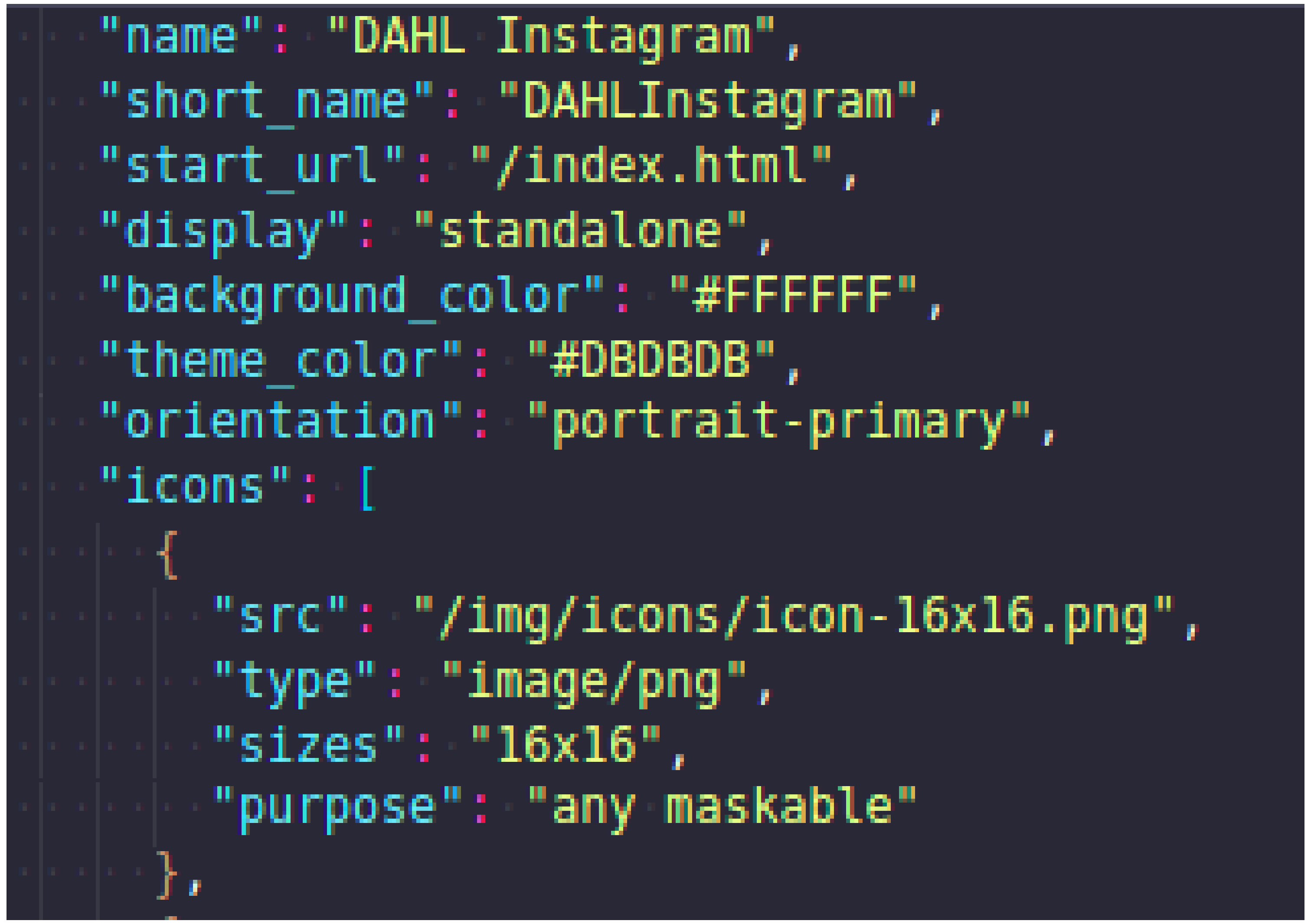
En este caso replicamos una aplicación con el estilo de Instagram.

Instalable en dispositivo





manifest.json

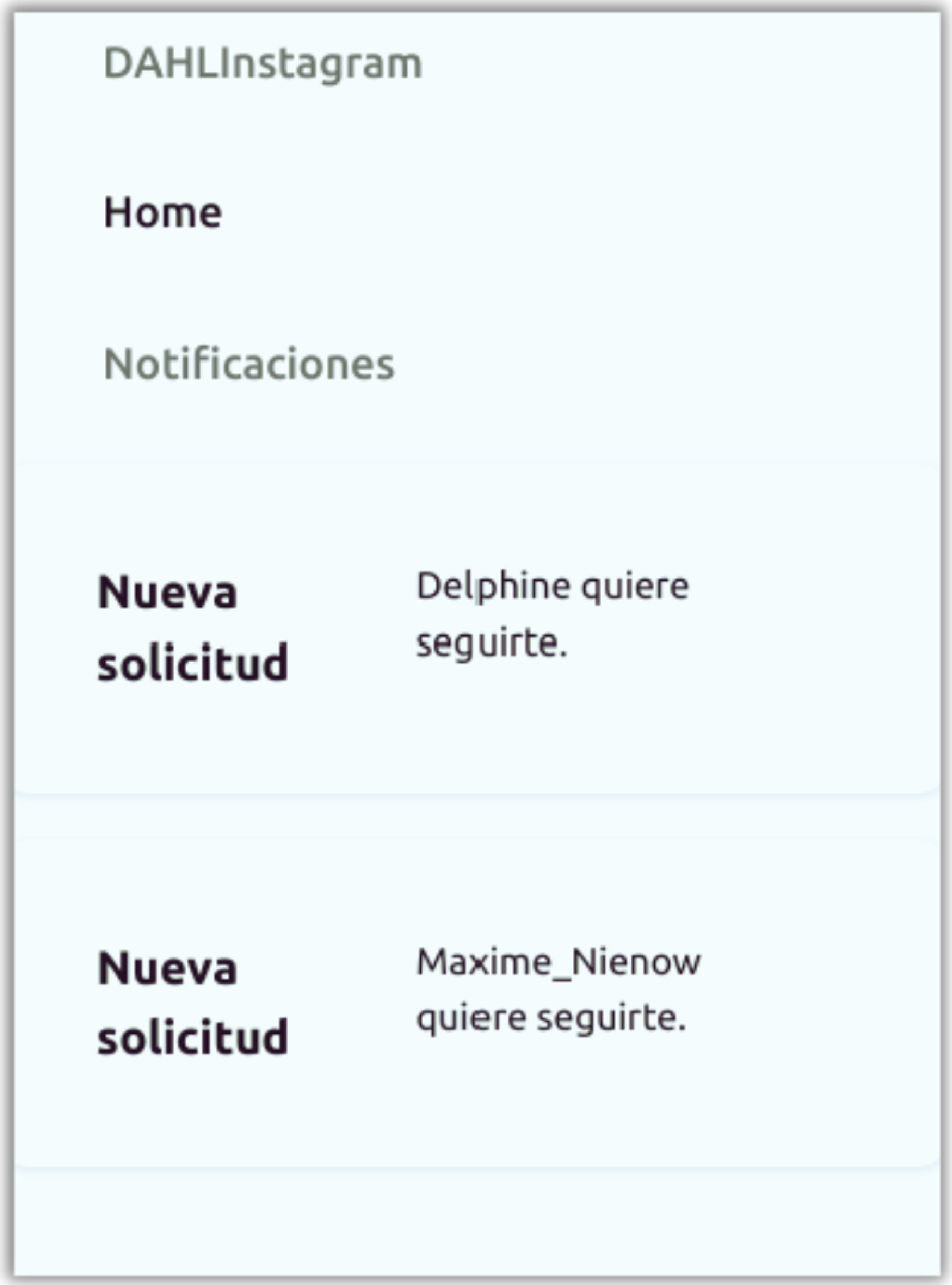


```
  "name": "DAHL Instagram",
  "short_name": "DAHLInstagram",
  "start_url": "/index.html",
  "display": "standalone",
  "background_color": "#FFFFFF",
  "theme_color": "#DBDBDB",
  "orientation": "portrait-primary",
  "icons": [
    {
      "src": "/img/icons/icon-16x16.png",
      "type": "image/png",
      "sizes": "16x16",
      "purpose": "any maskable"
    },
```

Notificaciones push

Notificaciones push

Firebase



DAHLInstagram

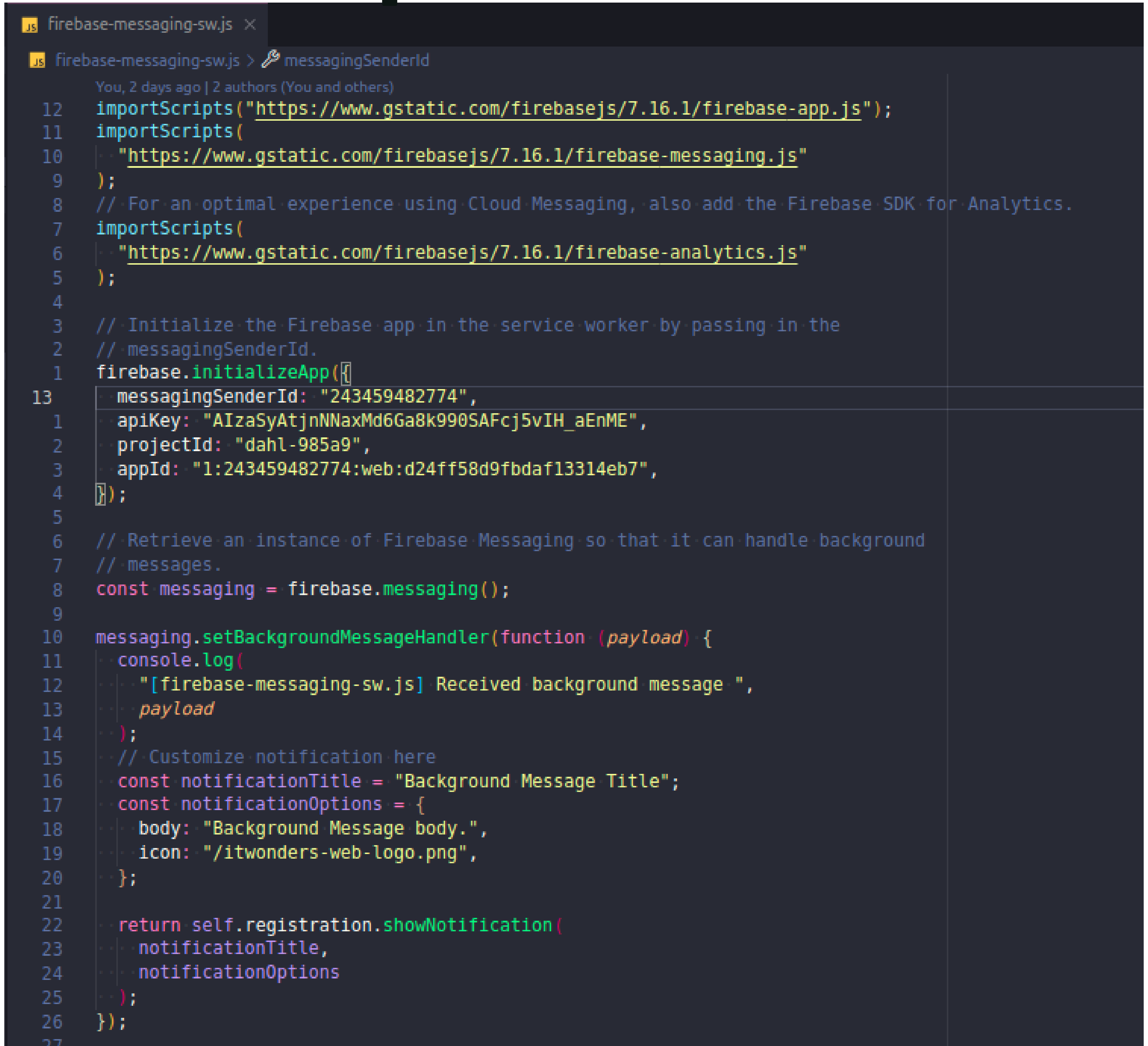
Home

Notificaciones

Nueva solicitud Delphine quiere seguirte.

Nueva solicitud Maxime_Nienow quiere seguirte.

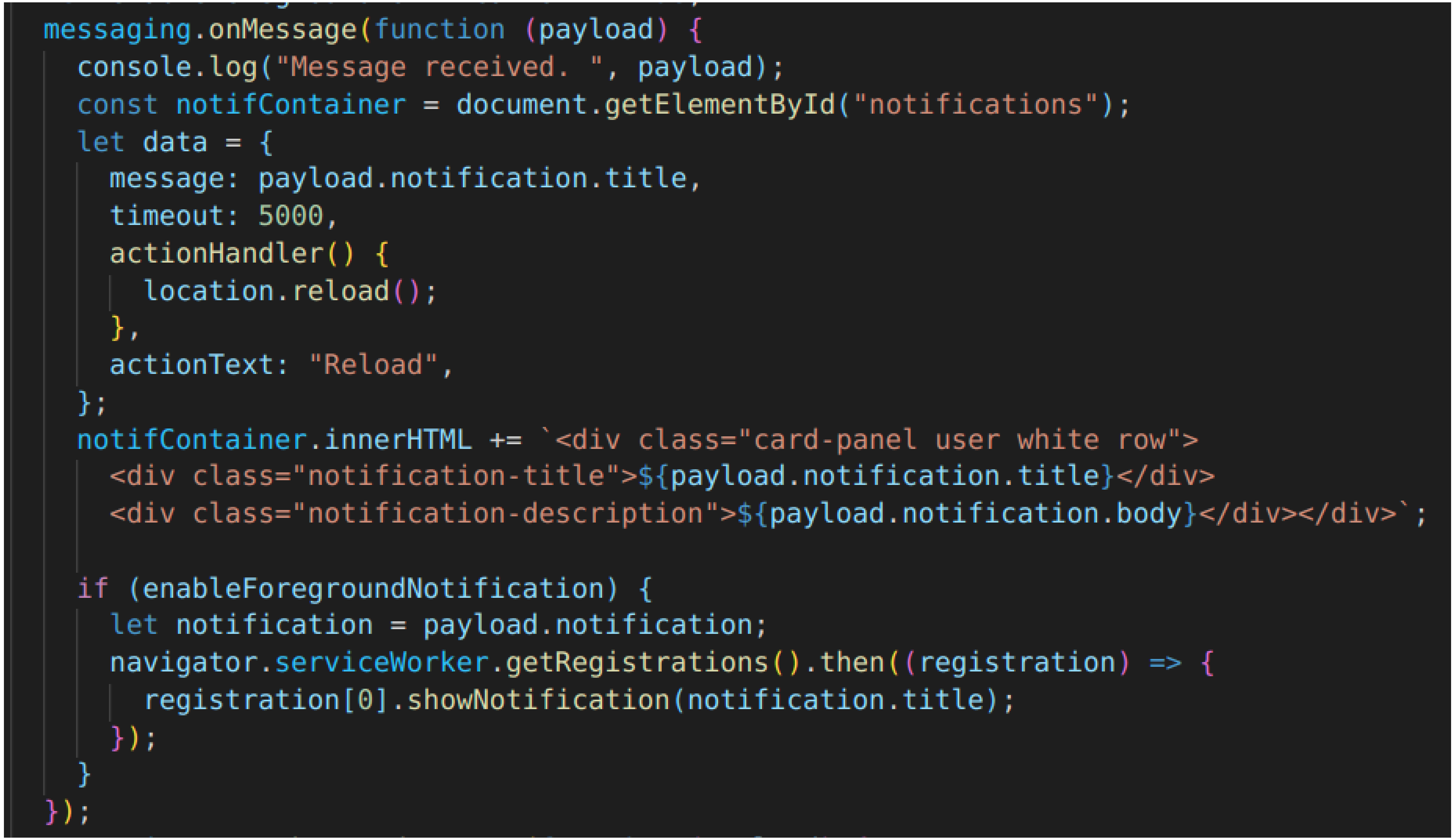
Notificaciones push



firebase-messaging-sw.js ×
firebase-messaging-sw.js > messagingSenderId
You, 2 days ago | 2 authors (You and others)

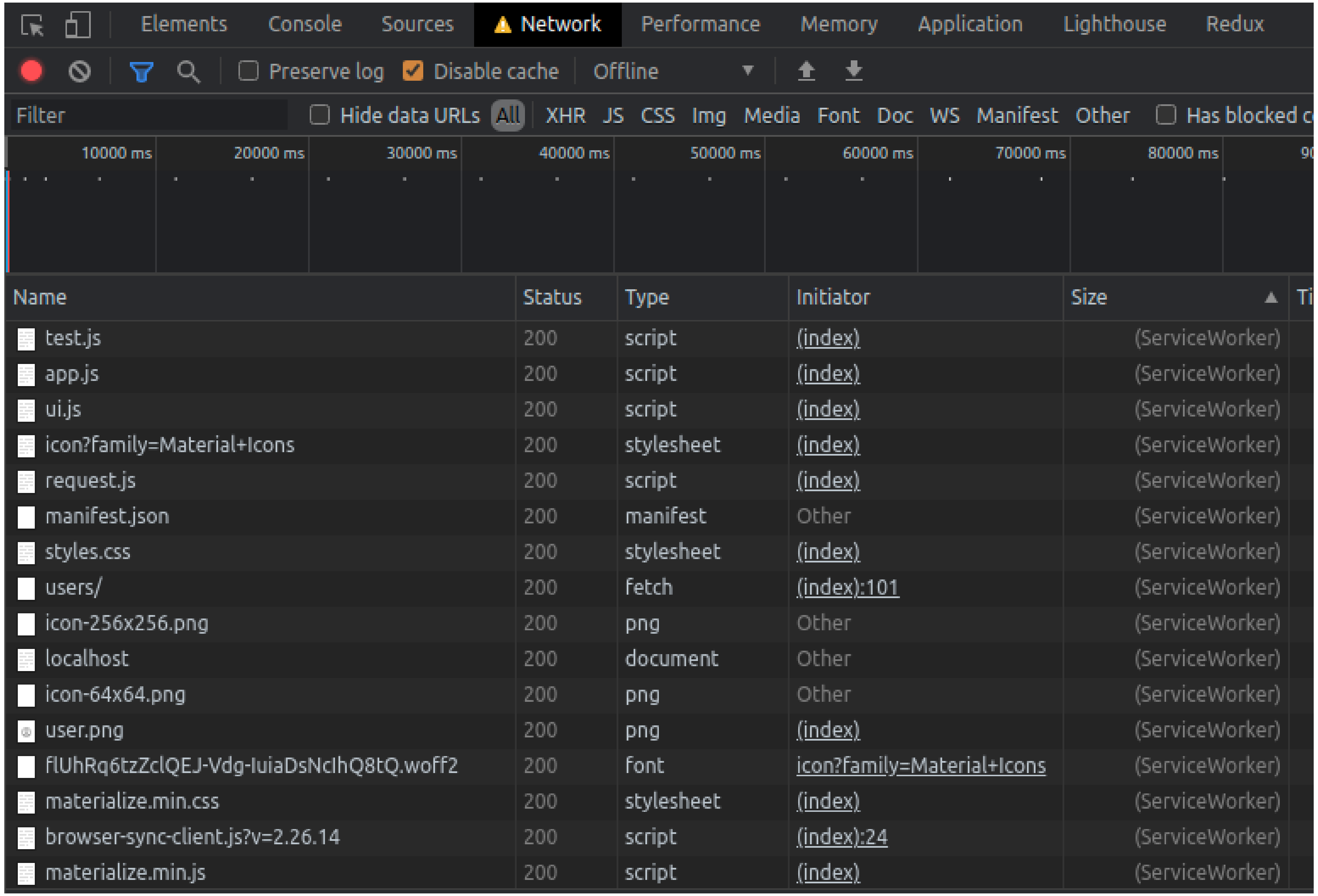
```javascript
importScripts("https://www.gstatic.com/firebasejs/7.16.1/firebase-app.js");
importScripts(
  "https://www.gstatic.com/firebasejs/7.16.1/firebase-messaging.js"
);
// For an optimal experience using Cloud Messaging, also add the Firebase SDK for Analytics.
importScripts(
  "https://www.gstatic.com/firebasejs/7.16.1/firebase-analytics.js"
);

// Initialize the Firebase app in the service worker by passing in the
// messagingSenderId.
firebase.initializeApp({
  messagingSenderId: "243459482774",
  apiKey: "AIzaSyAtjnNNaxMd6Ga8k990SAFcj5vIH_aEnME",
  projectId: "dahl-985a9",
  appId: "1:243459482774:web:d24ff58d9fbdaf13314eb7",
});

// Retrieve an instance of Firebase Messaging so that it can handle background
// messages.
const messaging = firebase.messaging();

messaging.setBackgroundMessageHandler(function (payload) {
  console.log(
    "[firebase-messaging-sw.js] Received background message ",
    payload
  );
  // Customize notification here
  const notificationTitle = "Background Message Title";
  const notificationOptions = {
    body: "Background Message body.",
    icon: "/itwonders-web-logo.png",
  };

  return self.registration.showNotification(
    notificationTitle,
    notificationOptions
  );
});
```

Notificaciones push



```
messaging.onMessage(function (payload) {
  console.log("Message received. ", payload);
  const notifContainer = document.getElementById("notifications");
  let data = {
    message: payload.notification.title,
    timeout: 5000,
    actionHandler() {
      location.reload();
    },
    actionText: "Reload",
  };
  notifContainer.innerHTML += `<div class="card-panel user white row">
    <div class="notification-title">${payload.notification.title}</div>
    <div class="notification-description">${payload.notification.body}</div></div>`;

  if (enableForegroundNotification) {
    let notification = payload.notification;
    navigator.serviceWorker.getRegistrations().then((registration) => {
      registration[0].showNotification(notification.title);
    });
  }
});
```

Funcionamiento offline

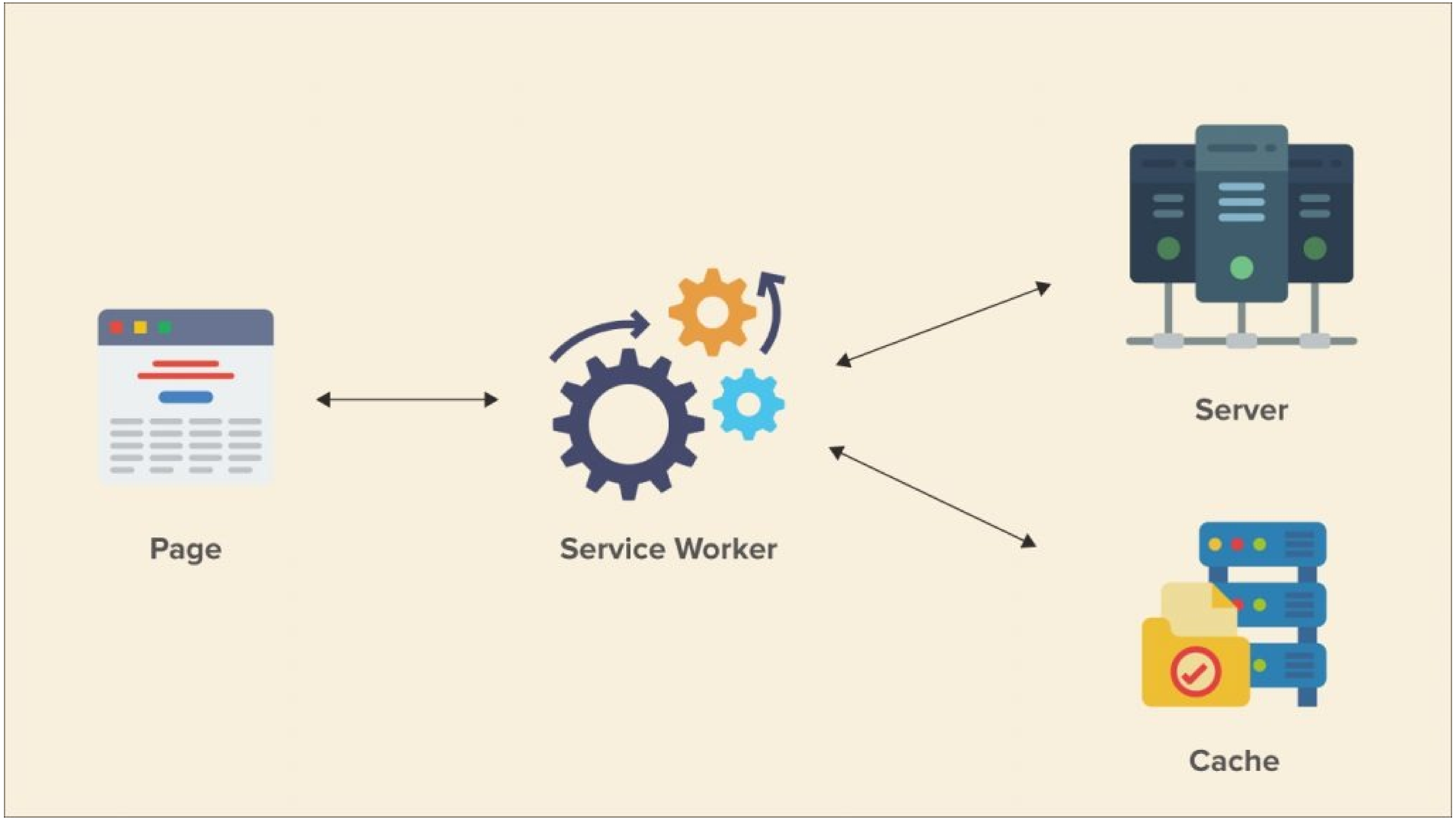


☐ Preserve log ☑ Disable cache Offline

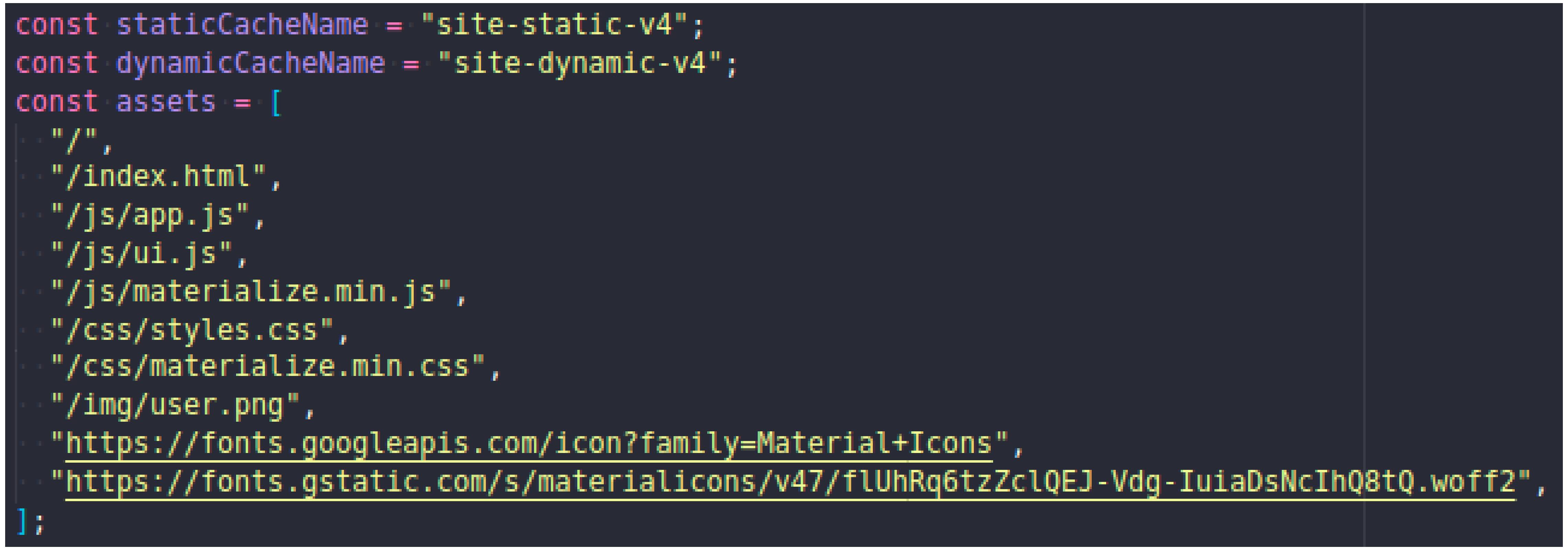
☐ Hide data URLs ☐ Has blocked c

Name	Status	Type	Initiator	Size	Ti
test.js	200	script	(index)	(ServiceWorker)	
app.js	200	script	(index)	(ServiceWorker)	
ui.js	200	script	(index)	(ServiceWorker)	
icon?family=Material+Icons	200	stylesheet	(index)	(ServiceWorker)	
request.js	200	script	(index)	(ServiceWorker)	
manifest.json	200	manifest	Other	(ServiceWorker)	
styles.css	200	stylesheet	(index)	(ServiceWorker)	
users/	200	fetch	(index):101	(ServiceWorker)	
icon-256x256.png	200	png	Other	(ServiceWorker)	
localhost	200	document	Other	(ServiceWorker)	
icon-64x64.png	200	png	Other	(ServiceWorker)	
user.png	200	png	(index)	(ServiceWorker)	
flUhRq6tzZclQEJ-Vdg-IuiaDsNcIhQ8tQ.woff2	200	font	icon?family=Material+Icons	(ServiceWorker)	
materialize.min.css	200	stylesheet	(index)	(ServiceWorker)	
browser-sync-client.js?v=2.26.14	200	script	(index):24	(ServiceWorker)	
materialize.min.js	200	script	(index)	(ServiceWorker)	

Service worker

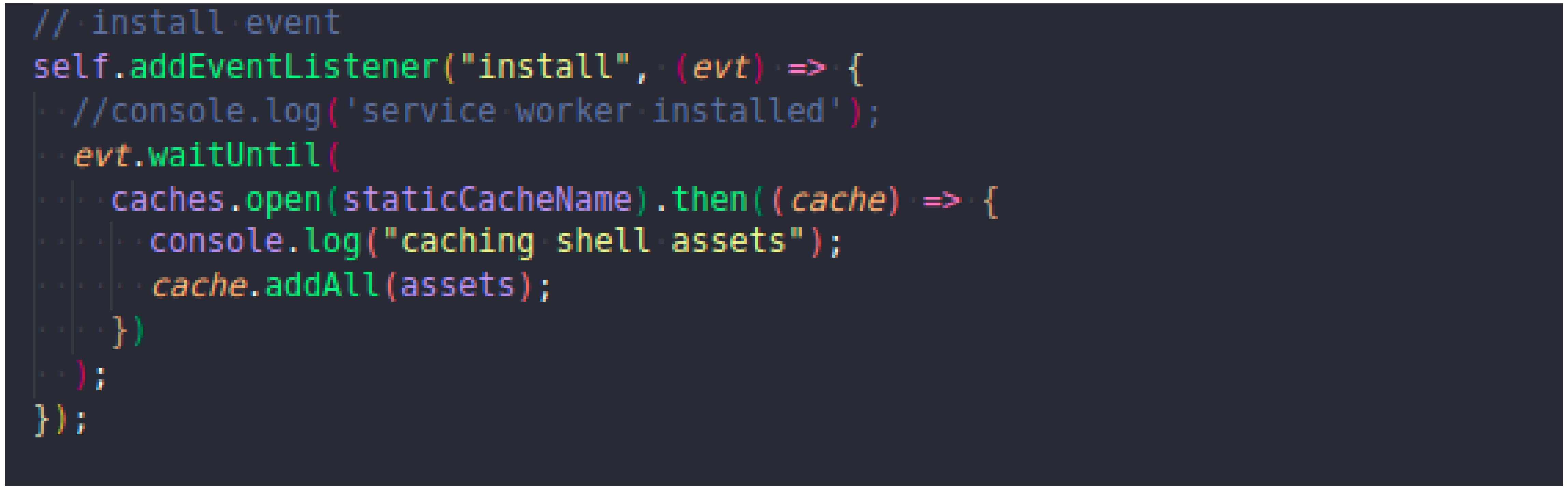


Service worker



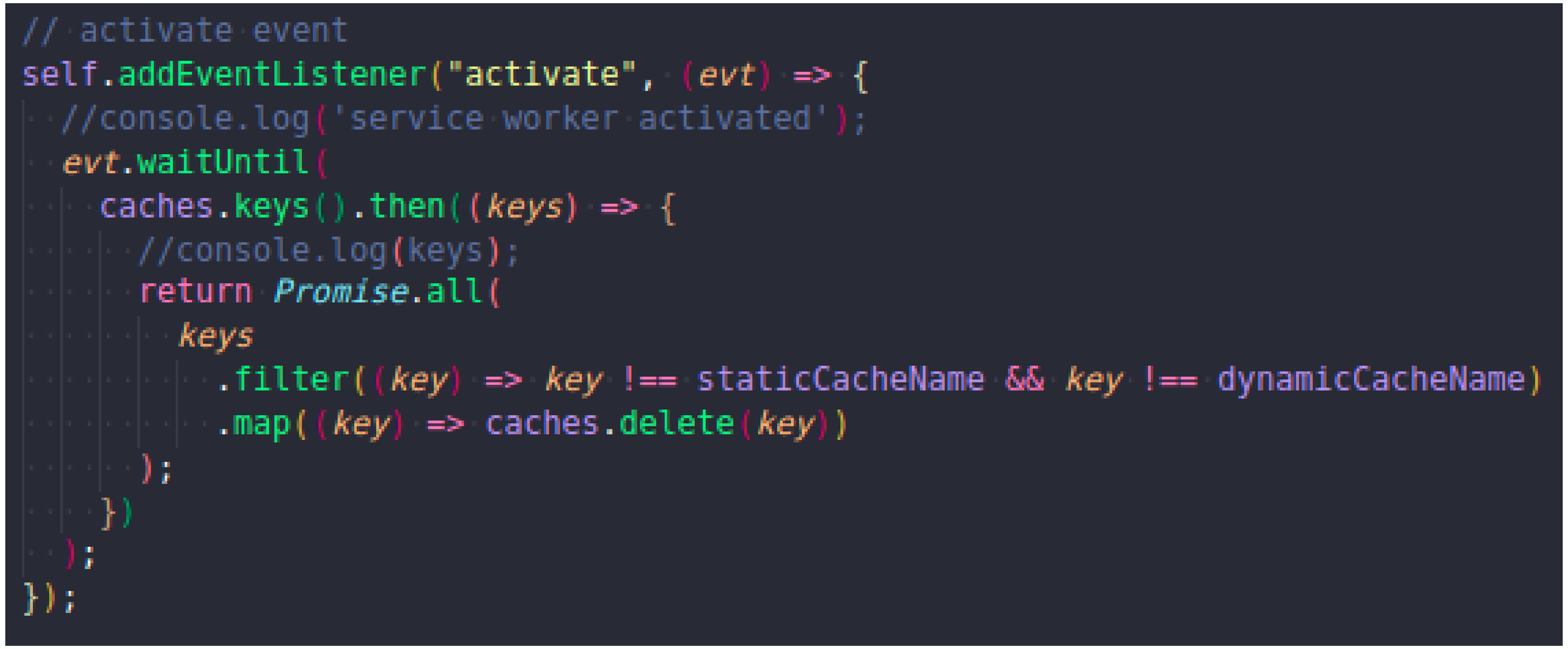
```
const staticCacheName = "site-static-v4";
const dynamicCacheName = "site-dynamic-v4";
const assets = [
  "/",
  "/index.html",
  "/js/app.js",
  "/js/ui.js",
  "/js/materialize.min.js",
  "/css/styles.css",
  "/css/materialize.min.css",
  "/img/user.png",
  "https://fonts.googleapis.com/icon?family=Material+Icons",
  "https://fonts.gstatic.com/s/materialicons/v47/flUhRq6tzZclQEJ-Vdg-IuiaDsNcIhQ8tQ.woff2",
];
```

Service worker



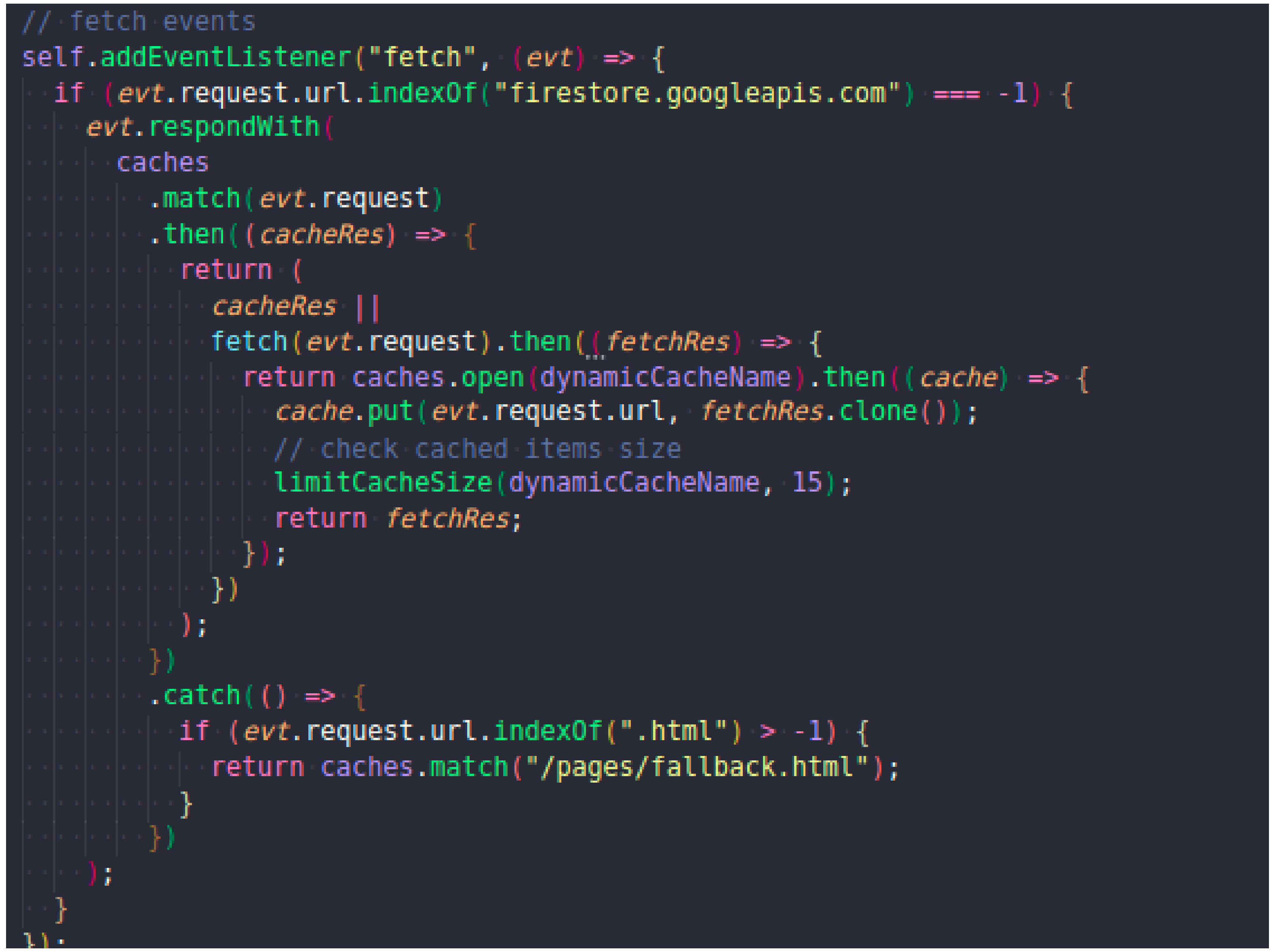
```
// install event
self.addEventListener("install", (evt) => {
  //console.log('service worker installed');
  evt.waitUntil(
    caches.open(staticCacheName).then((cache) => {
      console.log("caching shell assets");
      cache.addAll(assets);
    })
  );
});
```

Service worker



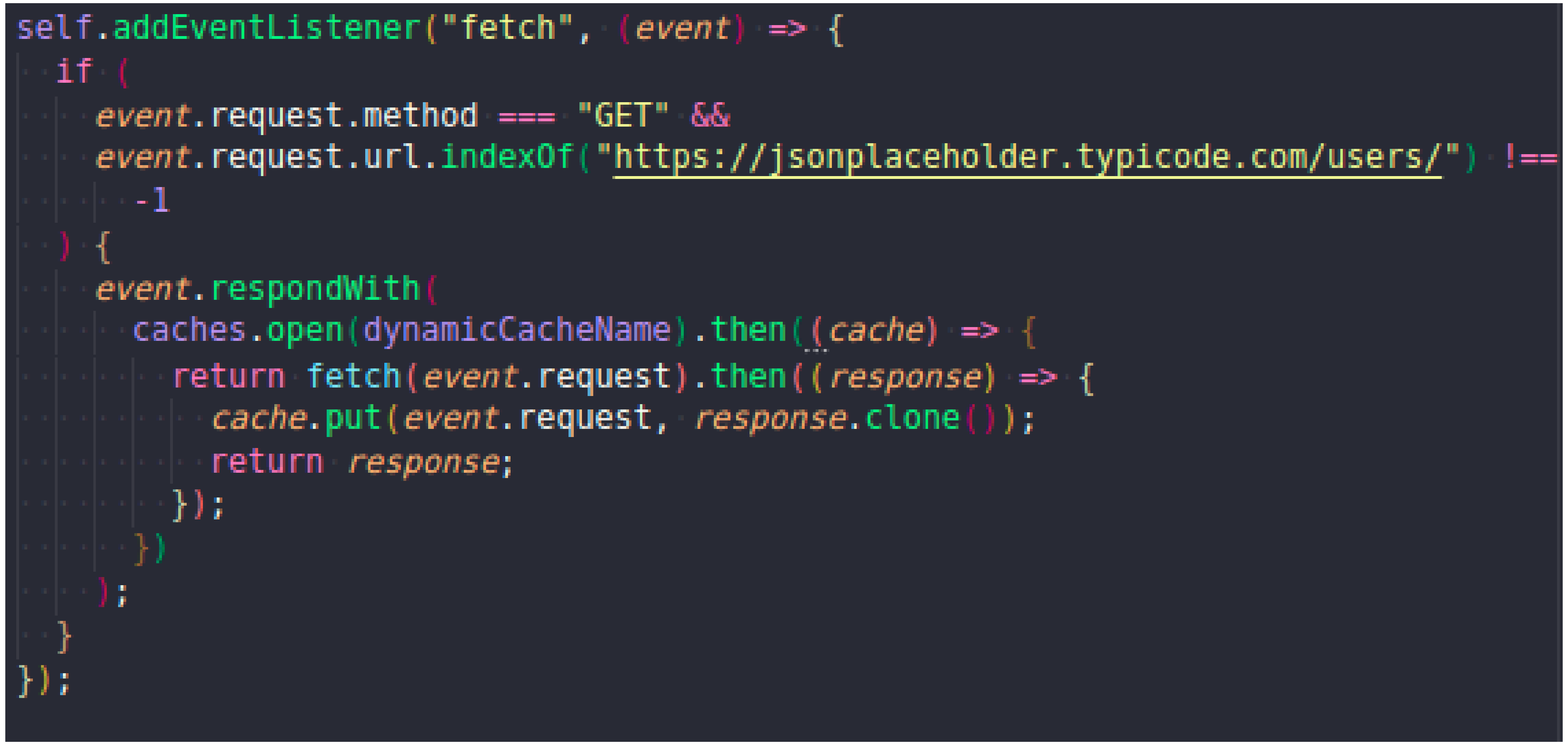
```
// activate event
self.addEventListener("activate", (evt) => {
  //console.log('service worker activated');
  evt.waitUntil(
    caches.keys().then((keys) => {
      //console.log(keys);
      return Promise.all(
        keys
          .filter((key) => key !== staticCacheName && key !== dynamicCacheName)
          .map((key) => caches.delete(key))
      );
    })
  );
});
```

Service worker



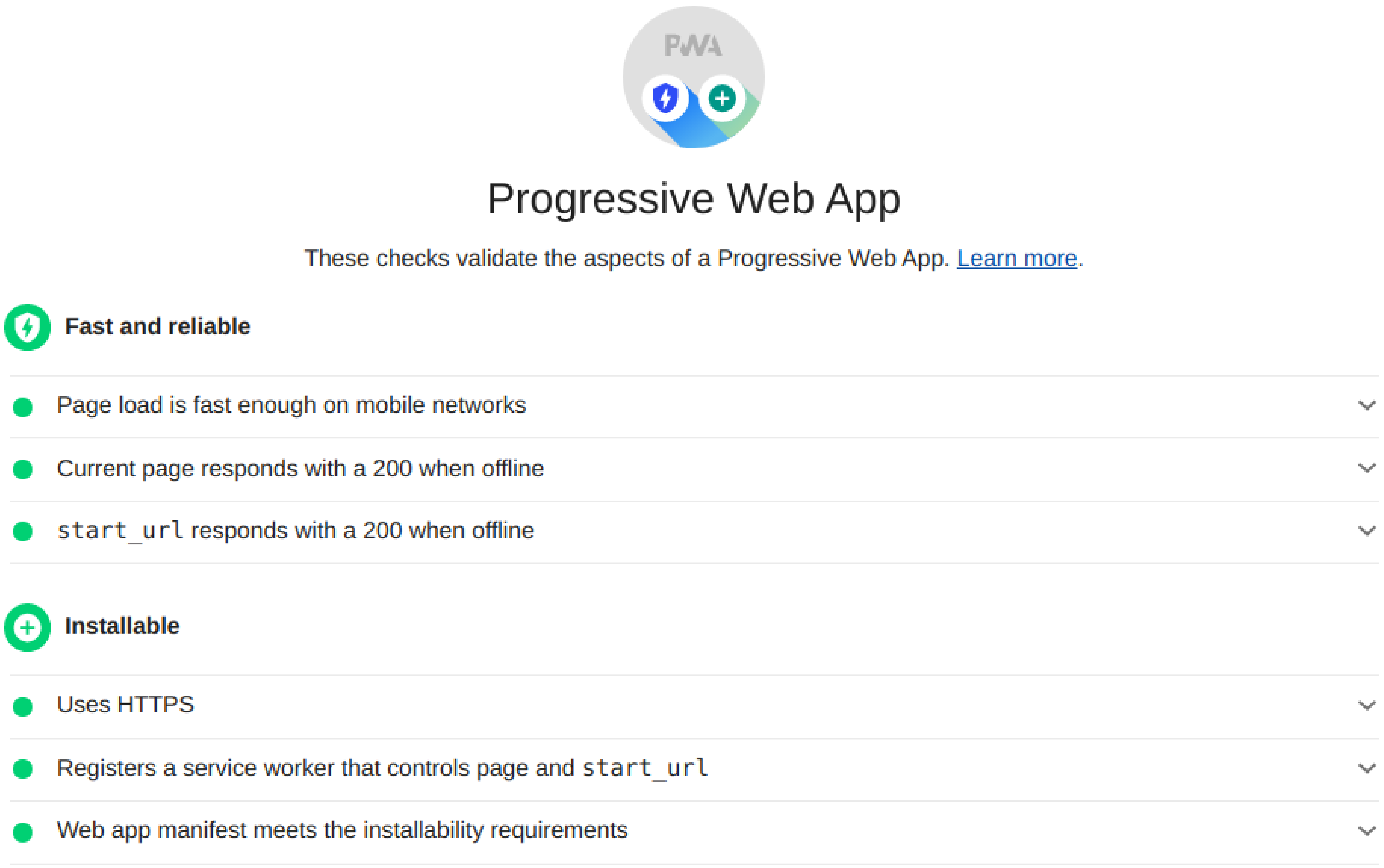
```
// fetch events
self.addEventListener("fetch", (evt) => {
  if (evt.request.url.indexOf("firestore.googleapis.com") === -1) {
    evt.respondWith(
      caches
        .match(evt.request)
        .then((cacheRes) => {
          return (
            cacheRes ||
            fetch(evt.request).then((fetchRes) => {
              return caches.open(dynamicCacheName).then((cache) => {
                cache.put(evt.request.url, fetchRes.clone());
                // check cached items size
                limitCacheSize(dynamicCacheName, 15);
                return fetchRes;
              });
            })
          );
        })
        .catch(() => {
          if (evt.request.url.indexOf(".html") > -1) {
            return caches.match("/pages/fallback.html");
          }
        })
    );
  }
});
```

Service worker



```
self.addEventListener("fetch", (event) => {
  if (
    event.request.method === "GET" &&
    event.request.url.indexOf("https://jsonplaceholder.typicode.com/users/") !==
      -1
  ) {
    event.respondWith(
      caches.open(dynamicCacheName).then((cache) => {
        return fetch(event.request).then((response) => {
          cache.put(event.request, response.clone());
          return response;
        });
      })
    );
  }
});
```

Lighthouse report



Progressive Web App

These checks validate the aspects of a Progressive Web App. Learn more.

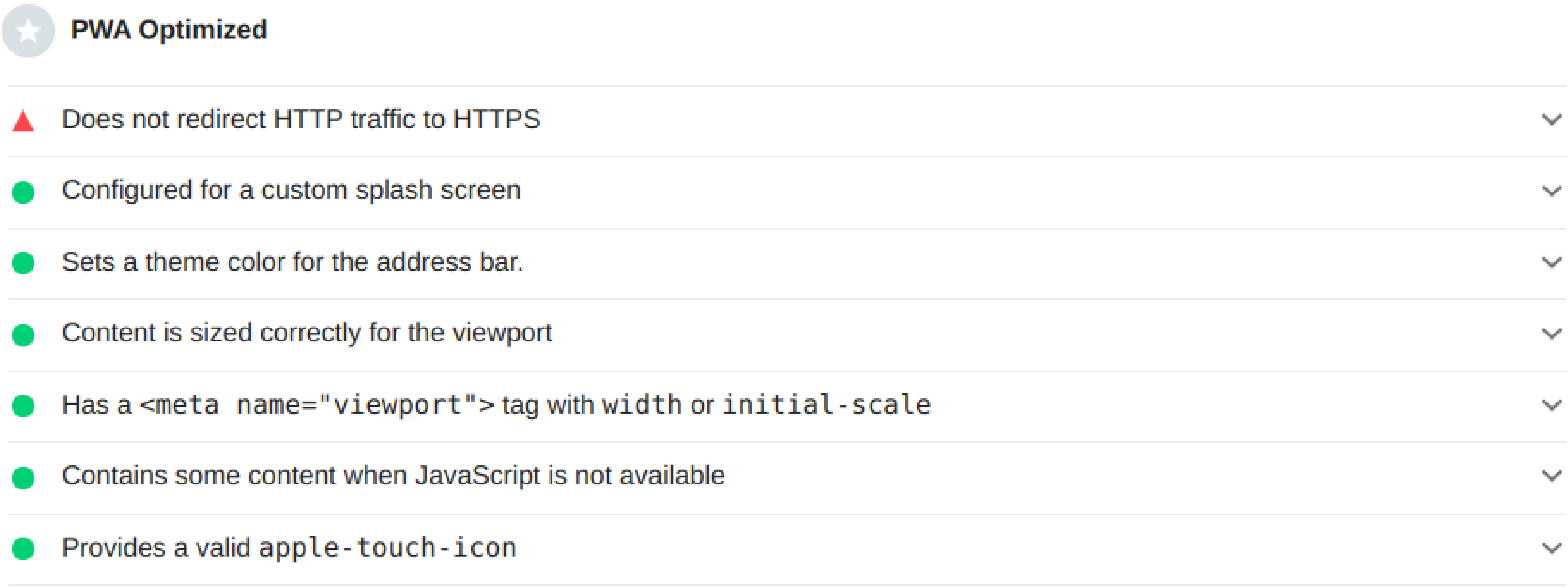
Fast and reliable

- ● Page load is fast enough on mobile networks
- ● Current page responds with a 200 when offline
- ● `start_url` responds with a 200 when offline

Installable

- ● Uses HTTPS
- ● Registers a service worker that controls page and `start_url`
- ● Web app manifest meets the installability requirements

Lighthouse report



PWA Optimized

- ▲ Does not redirect HTTP traffic to HTTPS
- ● Configured for a custom splash screen
- ● Sets a theme color for the address bar.
- ● Content is sized correctly for the viewport
- ● Has a `<meta name="viewport">` tag with `width` or `initial-scale`
- ● Contains some content when JavaScript is not available
- ● Provides a valid `apple-touch-icon`

Aprendizajes & Dificultades

- El navegador de Chrome almacenaba versiones anteriores de la aplicación, por lo que era complicado el revisar las diferentes versiones y aplicar los cambios.
- Service workers.
- Aplicación offline.
- Notificaciones push con Firebase.
- Lighthouse audit
- Funcionalidades de Devtools.

MUCHAS GRACIAS