PWA

Grupo II

Cache

STATIC_CACHE → archivos estáticos

DYNAMIC_CACHE → requests que no pudieron ser cacheados

INDEXEDDB → requests a la api

```
self.addEventListener('install', (event) => {
  event.waitUntil((async () => {
    const cache = await caches.open(STATIC_CACHE);
    cache.addAll(STATIC_FILES);
  })());
});
```

```
self.addEventListener('fetch', (event) => {
  const url = 'https://igpwa-3d0a9-default-rtdb.firebaseio.com/images.json'
  if (event.request.url.indexOf(url) > -1) {
    event.respondWith((async () => {
      const response = await fetch(event.request);
      const clonedResponse = response.clone();
      const data = await clonedResponse.json();
      Object.keys(data).forEach((key) => {
        writeData('images', {
          id: key,
          url: data[key].url,
        });
      });
      return response;
    })())
  } else {
    .
    .
    .
  }
});
```

```
self.addEventListener('fetch', (event) => {
  const url = 'https://igpwa-3d0a9-default-rtdb.firebaseio.com/images.json'
  if (event.request.url.indexOf(url) > -1) {
    .
    .
    .
  } else {
    event.respondWith((async () => {
      const cachedResponse = await caches.match(event.request);
      if (cachedResponse) {
        return cachedResponse;
      }
      const response = await fetch(event.request);
      const cache = await caches.open(DYNAMIC_CACHE);
      cache.put(event.request.url, response.clone());
      return response;
    })());
  }
});
```

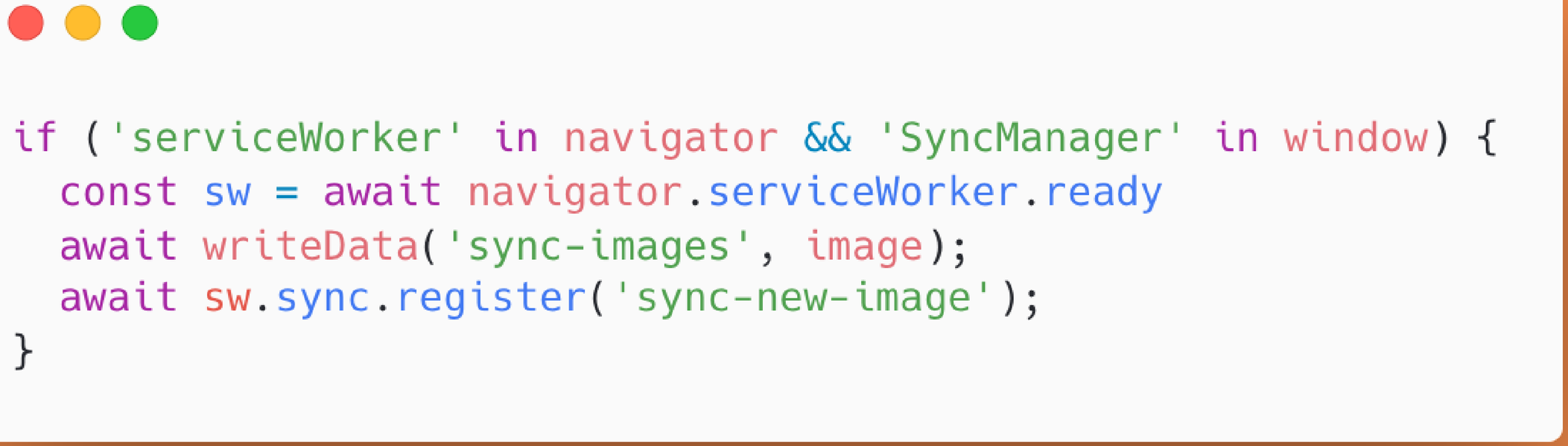
```
try {
    const response = await fetch('https://igpwa-3d0a9-default-rtdb.firebaseio.com/images.json');
    const data = await response.json();
    Object.keys(data).forEach((key) => {
      addImage(data[key].url);
    });
  } catch (error) {
    if ('indexedDB' in window) {
      const data = await readAllData('images');
      data.forEach(({ url }) => {
        addImage(url);
      });
    }
  }
```

Cache versioning

```
const STATIC_CACHE = 'static-cache-v1';
const DYNAMIC_CACHE = 'dynamic-cache-v1';
```

```
self.addEventListener('activate', (event) =>  {
  event.waitUntil((async () => {
    const keyList = await caches.keys();
    return Promise.all(keyList.map(function(key) {
      if (key !== STATIC_CACHE && key !== DYNAMIC_CACHE) {
        return caches.delete(key);
      }
    }));
  })())
  return self.clients.claim();
});
```

Background sync



```
if ('serviceWorker' in navigator && 'SyncManager' in window) {
  const sw = await navigator.serviceWorker.ready
  await writeData('sync-images', image);
  await sw.sync.register('sync-new-image');
}
```

```
self.addEventListener('sync', (event) => {
  if (event.tag === 'sync-new-image') {
    event.waitUntil((async () => {
      const data = await readAllData('sync-images');
      await Promise.all(data.map(({ url }) => (
        fetch(`https://igpwa-3d0a9-default-rtdb.firebaseio.com/images.json`, {
          method: 'POST',
          headers: {
            'Content-Type': 'application/json',
            'Accept': 'application/json',
          },
          body: JSON.stringify({
            url,
          }),
        })
      )));
      const clients = await self.clients.matchAll();
      clients.forEach((client) => {
        data.forEach(({ url }) => client.postMessage({
          type: 'load-image',
          image: url,
        }));
      });
      await clearAllData('sync-images');
    })());
  }
});
```

```
navigator.serviceWorker.addEventListener('message', (event) => {
  if (event.data.type === 'load-image') {
    addImage(event.data.image);
  }
});
```