

# MAD & PWA Lab

## Journal

Experiment No.	09
Experiment Title.	To implement Service worker events like fetch, sync and push for E-commerce PWA
Roll No.	56
Name	Mahvish Siddiqui
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

## EXPERIMENT NO. 9

**Aim:** To implement Service worker events like fetch, sync and push for E-commerce PWA.

### **Theory:**

#### **Service Worker**

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

- A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
- Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
- The service worker becomes idle when not in use and restarts when it's next needed. You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.
- Service workers make extensive use of promises, so if you're new to promises, then you should stop reading this and check out Promises, an introduction.

#### **Fetch Event**

You can track and manage page network traffic with this event. You can check existing cache, manage “cache first” and “network first” requests and return a response that you want.

Of course, you can use many different methods but you can find in the following example a “cache first” and “network first” approach. In this example, if the request’s and current location’s origin are the same (Static content is requested.), this is called “cacheFirst” but if you request a targeted external URL, this is called “networkFirst”.

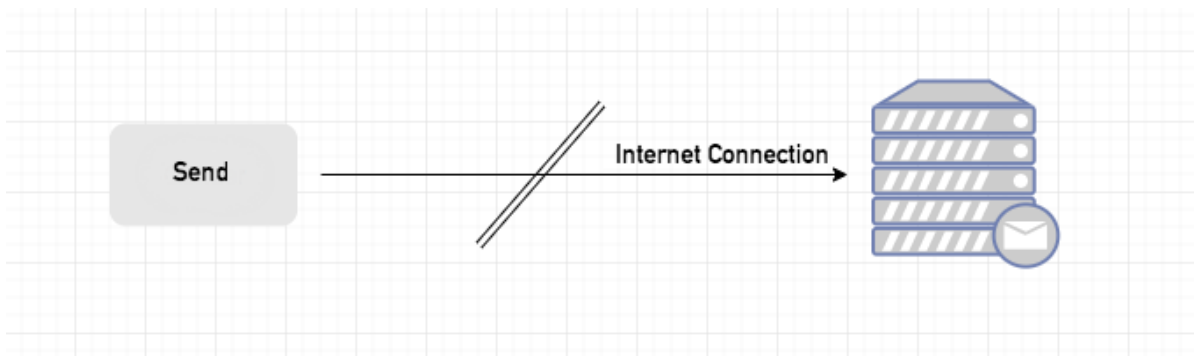
- **CacheFirst** - In this function, if the received request has cached before, the cached response is returned to the page. But if not, a new response requested from the network.
- **NetworkFirst** - In this function, firstly we can try getting an updated response from the network, if this process completed successfully, the new response will be cached and returned. But if this process fails, we check whether the request has been cached before or not. If a cache exists, it is returned to the page, but if not, this is up to you. You can return dummy content or information messages to the page.

## Sync Event

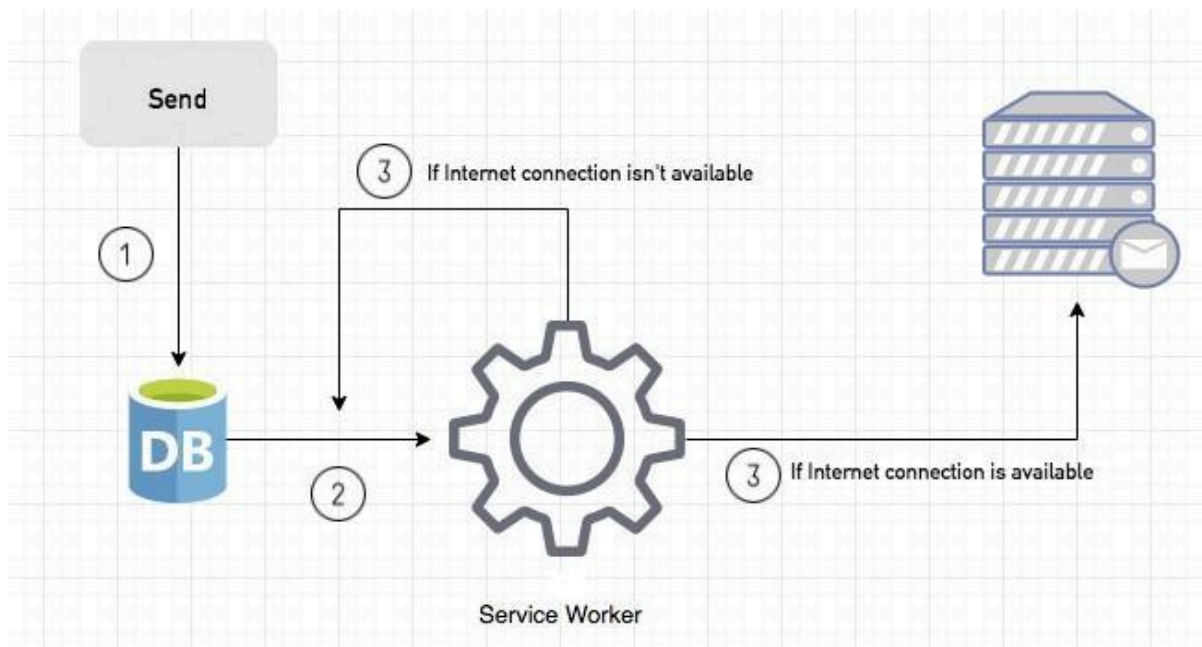
Background Sync is a Web API that is used to delay a process until the Internet connection is stable. We can adapt this definition to the real world; there is an e-mail client application that works on the browser and we want to send an email with this tool. Internet connection is broken while we are writing e-mail content and we didn't realize it. When completing the writing, we click the send button.

Here is a job for the Background Sync.

The following view shows the classical process of sending email to us. If the Internet Connection is broken, we can't send any content to Mail Server.



Here, you can create any scenario for yourself. A sample is in the following for this case.



1. When we click the "send" button, email content will be saved to IndexedDB.
2. Background Sync registration.
3. **If the Internet connection is available**, all email content will be read and sent to Mail Server.  
**If the Internet connection is unavailable**, the service worker waits until the connection is available even though the window is closed. When it is available, email content will be sent to Mail Server.

You can see the working process within the following code block.

#### Event Listener for Background Sync Registration

```
document.querySelector("button").addEventListener("click", async () => {  
  var swRegistration = await navigator.serviceWorker.register("sw.js");  
  swRegistration.sync.register("helloSync").then(function () {  
    console.log("helloSync success [main.js]");  
  });  
});
```

#### Event Listener for sw.js

```
self.addEventListener('sync', event => {  
  if (event.tag == 'helloSync') {  
    console.log("helloSync [sw.js]");  
  }  
});
```

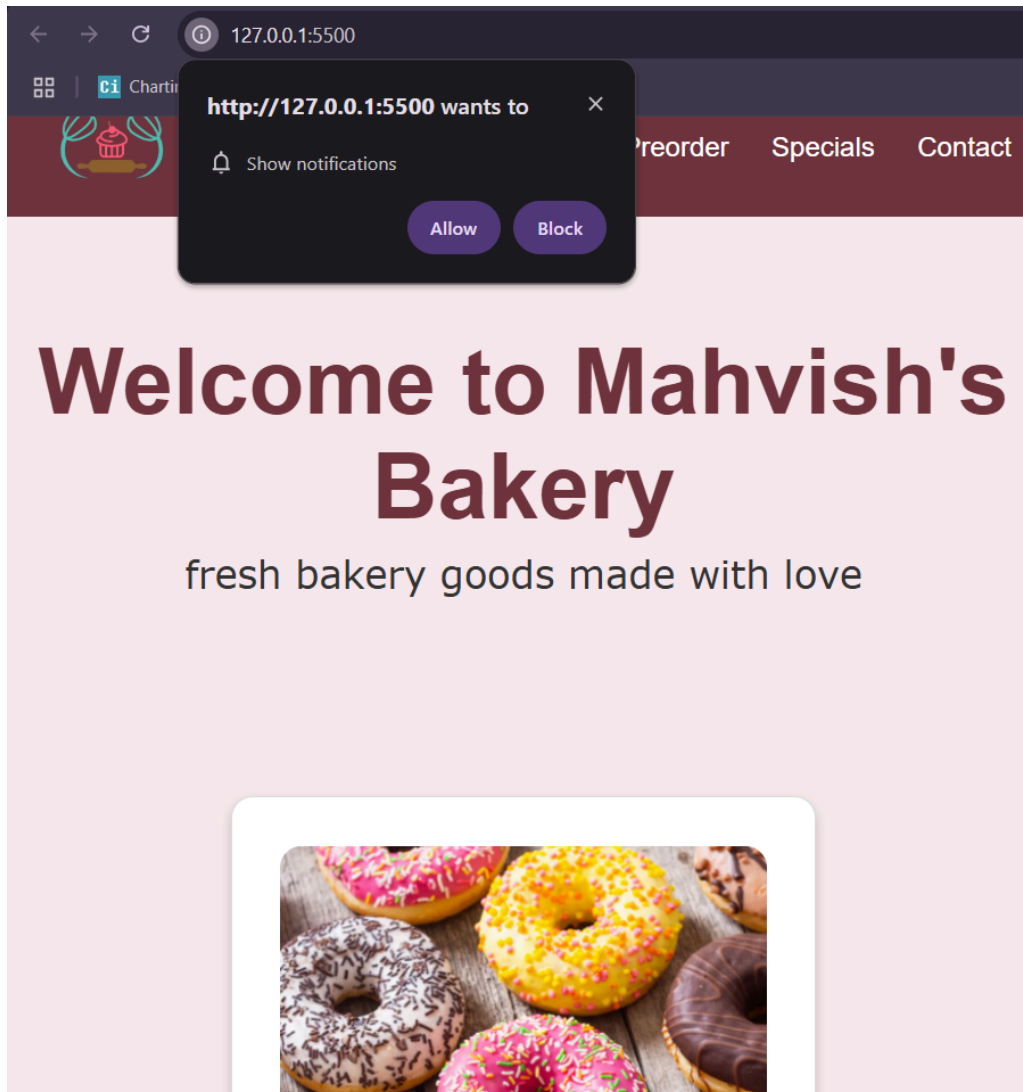
#### Push Event

This is the event that handles push notifications that are received from the server. You can apply any method with received data.

We can check in the following example.

“Notification.requestPermission();” is the necessary line to show notification to the user. If you don’t want to show any notification, you don’t need this line.

In the following code block is in sw.js file. You can handle push notifications with this event. In this example, I kept it simple. We send an object that has “method” and “message” properties. If the method value is “pushMessage”, we open the information notification with the “message” proper



127.0.0.1:5500

Your connection to this site is not secure

You should not enter any sensitive information on this site (for example, passwords or credit cards), because it could be stolen by attackers.

[Learn more](#)

Notifications

Reset permission

Cookies and site data

Site settings

Mahvish's Bakery

Home

Menu


Preorder

Specials

Contact

Welcome to Mahvish's Bakery

fresh bakery goods made with love



Application

Manifest

Service workers

Storage

Local storage

Session storage

Extension storage

IndexedDB

Cookies

Private state tokens

Interest groups

Shared storage

Cache storage

Storage buckets

Background services

Back/forward cache

Background fetch

Background sync

Bounce tracking miti...

Notifications

Payment handler

Periodic background ...

Speculative loads

Push messaging

Reporting API

Frames

Service workers

☐ Offline ☐ Update on reload ☐ Bypass for network

http://127.0.0.1:5500/

Source [service-worker.js](#)

Received 3/25/2025, 1:42:42 PM

Status ● #1071 activated and is running 

Stop

Push 

{ "method": "pushMessage", "message": "This is a test i

Push

Sync 

test-tag-from-devtools

Sync

Periodic sync 

test-tag-from-devtools

Periodic sync

Update Cycle

Version	Update Activity	Timeline
▶ #1071	Install	
▶ #1071	Wait	
▶ #1071	Activate	<div></div>

Service workers from other origins

[See all registrations](#)

Google Chrome

R

Mahvish's Bakery

This is a test message

127.0.0.1:5500

Mahvish's Bakery


Home

Menu

Preorder

# Welcome to Mahvish's Bakery

fresh bakery goods made with love



Application

- Manifest
- Service workers
- Storage

Storage

- Local storage
- Session storage
- Extension storage
- IndexedDB
- Cookies
- Private state tokens
- Interest groups
- Shared storage
- Cache storage
- Storage buckets

Background services

- Back/forward cache
- Background fetch
- Background sync
- Bounce tracking mitigations
- Notifications

Service workers

- ☐ Offline
- ☐ Update on reload
- ☒ Bypass for network

http://127.0.0.1:5500/

Source service-worker.js

Received 3/25/2025, 4:08:12 PM

Status #547 activated and is running 

Stop

Push { "method": "pushMessage", "message": "Hi" } 

Push

Sync test-tag-from-devtools 

Sync

Periodic sync test-tag-from-devtools 

Periodic sync

Update Cycle

Version	Update Activity	Timeline
#547	Install	
#547	Wait	
#547	Activate	

Console

- What's new
- AI assistance

Notification permission: granted

Permission granted for notifications

Service Worker registered successfully!

Notification displayed!

Google Chrome

R

Mahvish's Bakery

Hi

127.0.0.1:5500

Mahvish's Bakery

Home

Menu


Preorder

Specials

Con

# Welcome to Mahvish's Bakery

fresh bakery goods made with love



Application

- Manifest
- Service workers
- Storage

Storage

- Local storage
- Session storage
- Extension storage
- IndexedDB
- Cookies
- Private state tokens
- Interest groups
- Shared storage
- Cache storage
- my-pwa-cache-v1 - http://...
- Storage buckets

Service workers

- ☐ Offline
- ☒ Update on reload
- ☐ Bypass for network

http://127.0.0.1:5500/

Source service-worker.js

Received 3/26/2025, 1:14:44 AM

Status #559 activated and is running 

Stop

Push { "method": "pushMessage", "message": "Hi" } 

Push

Sync syncMessage 

Sync

Periodic sync test-tag-from-devtools 

Periodic sync

Update Cycle

Version	Update Activity	Timeline
---------	-----------------	----------

Console

- What's new
- AI assistance

Fetch successful! service-worker.js:55

Notification permission: granted index.html:190

Permission granted for notifications index.html:192

Service Worker: Fetching http://127.0.0.1:5500/manifest.json service-worker.js:49

Serving from cache: http://127.0.0.1:5500/manifest.json service-worker.js:54

Fetch successful! service-worker.js:55

Service Worker registered successfully! index.html:177

Service Worker: Fetching http://127.0.0.1:5500/icons/icon-192-1.png service-worker.js:49

Serving from cache: http://127.0.0.1:5500/icons/icon-192-1.png service-worker.js:54

Fetch successful! service-worker.js:55

