**Name:Komal Singh**
**Div:D15B**
**Roll No:56**

**MAD lab 9**

## Aim:

To implement and understand different service worker lifecycle events such as *install*, *activate*, *fetch*, *sync*, and *push*, in order to enhance the functionality of Progressive Web Apps (PWA) by enabling offline support, background sync, and real-time notifications.

## Theory:

A **Service Worker** is a JavaScript script that operates in the background of a PWA. It intercepts network requests, manages caching, enables offline functionality, and handles background processes like sync and push notifications.

**Key Service Worker Events:**

1. **Install Event:**
    - Caches essential assets (HTML, CSS, JS, images) during installation.
    - Allows the app to load quickly and work offline.
2. **Activate Event:**
    - Deletes old caches from previous versions.
    - Ensures the app uses updated resources and reduces storage usage.
3. **Fetch Event:**
    - Intercepts network requests.
    - Supports **Cache-First** (load from cache) and **Network-First** (fetch and cache) strategies for improved performance and reliability.
4. **Sync Event (Background Sync):**
    - Temporarily stores data (e.g., form submissions) when offline.
    - Automatically sends the data when the user reconnects to the internet.

5. **Offline Media Upload (e.g., Screenshots):**
    - Media captured offline is saved locally.
    - Automatically uploaded once a connection is restored, ensuring no data is lost.
6. **Push Event:**
    - Enables push notifications.
    - Notifies users about updates or important information even when the app is not active.

**1 Install Event: Cache Assets**

- During the `install` event, essential assets such as HTML, CSS, JavaScript, and images are stored in the cache.
- This ensures the application can load quickly and function offline if needed.

**2 Activate Event: Cleanup Old Caches**

- When a new service worker is activated, outdated caches from previous versions are deleted.
- This prevents unnecessary storage usage and ensures users always receive the latest updates.

**3 Fetch Event: Supports Both Cache-First & Network-First Strategies**

- If the requested resource is available in the cache, it is served immediately (cache-first strategy).
- If not, a network request is made, and the response is stored in the cache for future use (network-first strategy).
- This approach improves performance and ensures availability even when offline.

**4 Sync Event: Retry Sending Data When Online**

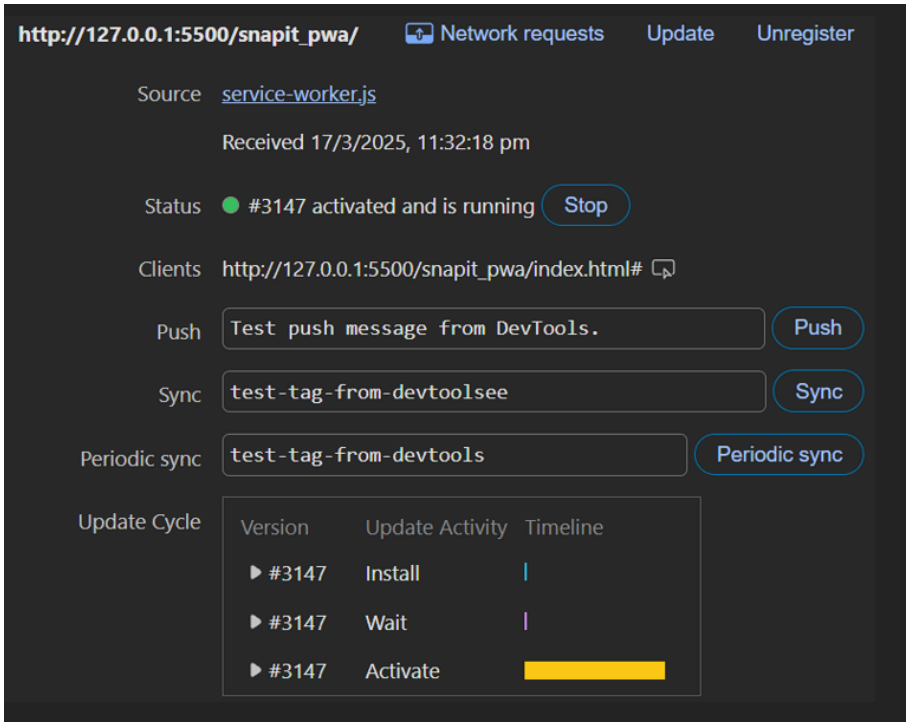
- If the user submits data while offline (e.g., form submissions or transactions), the service worker stores it locally.
- When the device reconnects to the internet, the stored data is automatically sent to the server.
- This ensures a seamless user experience and prevents data loss.

**5 Function to Send Pending Screenshots to the Server**

- Screenshots or other pending media files captured while offline are temporarily stored.
- When an internet connection is restored, these files are automatically uploaded to the server.
- This guarantees that critical data is not lost due to connectivity issues.

**6 Push Event: Display Push Notifications**

- The service worker listens for push notifications from the server.
- When a notification is received, it displays a system notification to the user.
- This helps keep users engaged by providing real-time updates.






## Conclusion:

By implementing various service worker events, we enhanced the PWA to be more reliable, responsive, and user-friendly. The app can now cache resources, clean up outdated files, fetch data smartly, sync offline interactions, upload media once online, and display push notifications. This leads to a seamless and engaging experience for users regardless of their network connectivity, making the web app function more like a native mobile app.