Experiment No. 7

Aim: To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.

Theory:

- **Regular Web App**

  A regular web app is a website that is designed to be accessible on all mobile devices such that the content gets fit as per the device screen. It is designed using a web technology stack (HTML, CSS, JavaScript, Ruby, etc.) and operates via a browser. They offer various native-device features and functionalities. However, it entirely depends on the browser the user is using. In other words, it might be possible that you can access a native-device feature on Chrome but not on Safari or Mozilla Firefox because the browsers are incompatible with that feature.

- **Progressive Web App**

  Progressive Web App (PWA) is a regular web app, but some extras enable it to deliver an excellent user experience. It is a perfect blend of desktop and mobile application experience to give both platforms to the end-users.

- **Difference between PWAs vs. Regular Web Apps:**

  1. Native Experience: Though a PWA runs on web technologies (HTML, CSS, JavaScript) like a Regular web app, it gives user experience like a native mobile application. It can use most native device features, including push notifications, without relying on the browser or any other entity. It offers a seamless and integrated user experience that it is quite tough for one to differentiate between a PWA and a Native application by considering its look and feel.

  2. Ease of Access: Unlike other mobile apps, PWAs do not demand longer download time and make memory space available for installing the applications. The PWAs can be shared and installed by a link, which cuts down the number of steps to install and use. These applications can easily keep an app icon on the user's home screen, making the app easily accessible to the users and helps the brands remain in the users' minds, and improving the chances of interaction.

  3. Faster Services: PWAs can cache the data and serve the user with text stylesheets, images, and other web content even before the page loads completely. This lowers the waiting time for the end-users and helps the brands improve the user engagement and retention rate, which eventually adds value to their business.

  4. Engaging Approach: As already shared, the PWAs can employ push notifications and other native device features more efficiently. Their interaction does not depend on the browser user uses. This eventually improves the chances of notifying the user regarding your services, offers, and other options related to your brand and keeping them hooked to your brand.

5. **Updated Real:** Time Data Access: Another plus point of PWAs is that these apps get updated on their own. They do not demand the end-users to go to the App Store or other such platforms to download the update and wait until installed.

6. **Discoverable**: PWAs reside in web browsers. This implies higher chances of optimizing them as per the Search Engine Optimization (SEO) criteria and improving the Google rankings like that in websites and other web apps.

7. **Lower Development Cost:** Progressive web apps can be installed on the user device like a native device, but it does not demand submission on an App Store. This makes it far more cost-effective than native mobile applications while offering the same set of functionalities.

- **The main features are:**

1. Progressive — They work for every user, regardless of the browser chosen because they are built at the base with progressive improvement principles.
2. Responsive — They adapt to the various screen sizes: desktop, mobile, tablet, or dimensions that can later become available.
3. Updated — Information is always up-to-date thanks to the data update process offered by service workers.
4. Secure — Exposed over HTTPS protocol to prevent the connection from displaying information or altering the contents.
5. Searchable — They are identified as “applications” and are indexed by search engines.
6. Reactivable — Make it easy to reactivate the application thanks to capabilities such as web notifications.
7. Installable — They allow the user to “save” the apps that he considers most useful with the corresponding icon on the screen of his mobile terminal (home screen) without having to face all the steps and problems related to the use of the app store.
8. Linkable — Easily shared via URL without complex installations.
9. Offline — Once more it is about putting the user before everything, avoiding the usual error message in case of weak or no connection. The PWA are based on two particularities: first of all the ‘skeleton’ of the app, which recalls the page structure, even if its contents do not respond and its elements include the header, the page layout, as well as an illustration that signals that the page is loading.

Code:

1. **Manifest.json**

```
{
    "name": "Simple PWA",
    "short_name": "PWA",
    "start_url": "index.html",
    "display": "standalone",
    "background_color": "#ffffff",
    "theme_color": "#000000",
    "description": "A simple Progressive Web App that explains what a PWA is.",
    "icons": [
        {
            "src": "pwa-banner.png",
            "sizes": "192x192",
            "type": "image/png"
        }
    ]
}
```

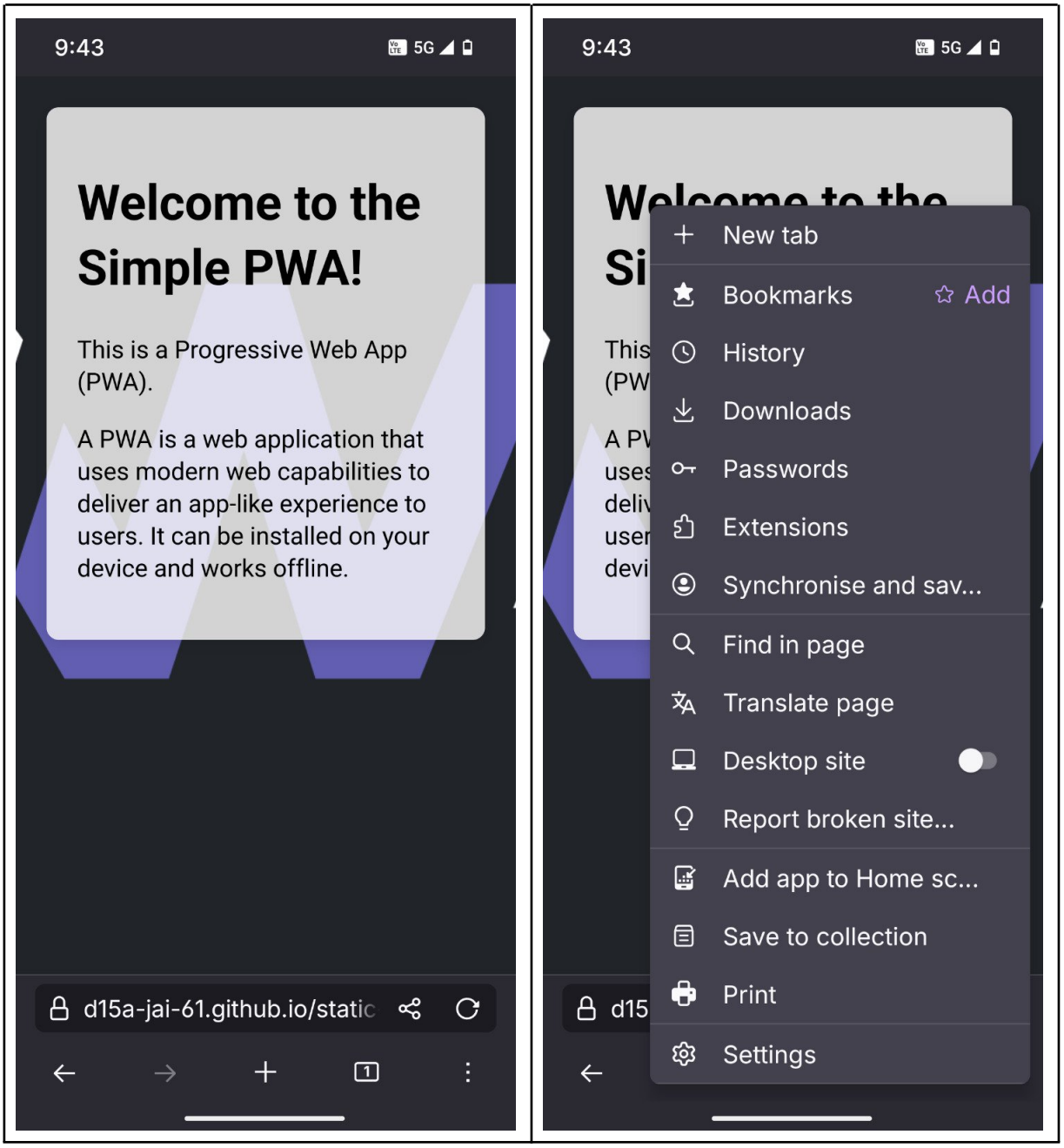
2. **Service-worker.js**

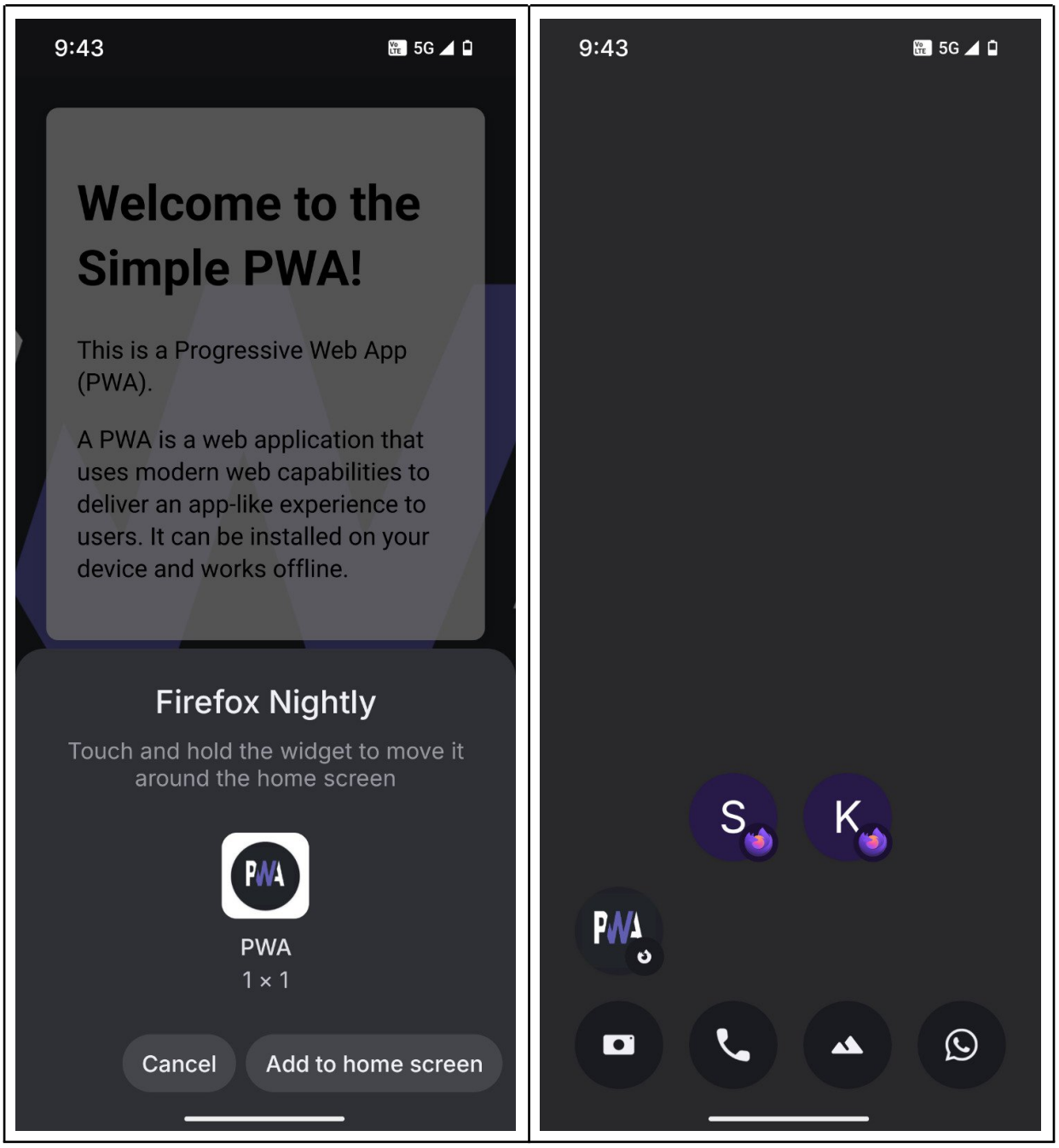
```
const CACHE_NAME = 'simple-pwa-cache-v1';
const urlsToCache = [
    'index.html',
    'offline-form.html',
    'styles.css',
    'manifest.json',
    'pwa-banner.png',
];

// Install the service worker and cache assets
self.addEventListener('install', (event) => {
    event.waitUntil(
        caches.open(CACHE_NAME).then((cache) => {
            console.log('[Service Worker] Caching essential assets');
            return cache.addAll(urlsToCache);
        })
    );
});

// Fetch event - serve cached assets or fetch from network
self.addEventListener('fetch', (event) => {
    event.respondWith(
        caches.match(event.request).then((response) => {
            if (response) {
                console.log('[Service Worker] Returning cached resource:',
```

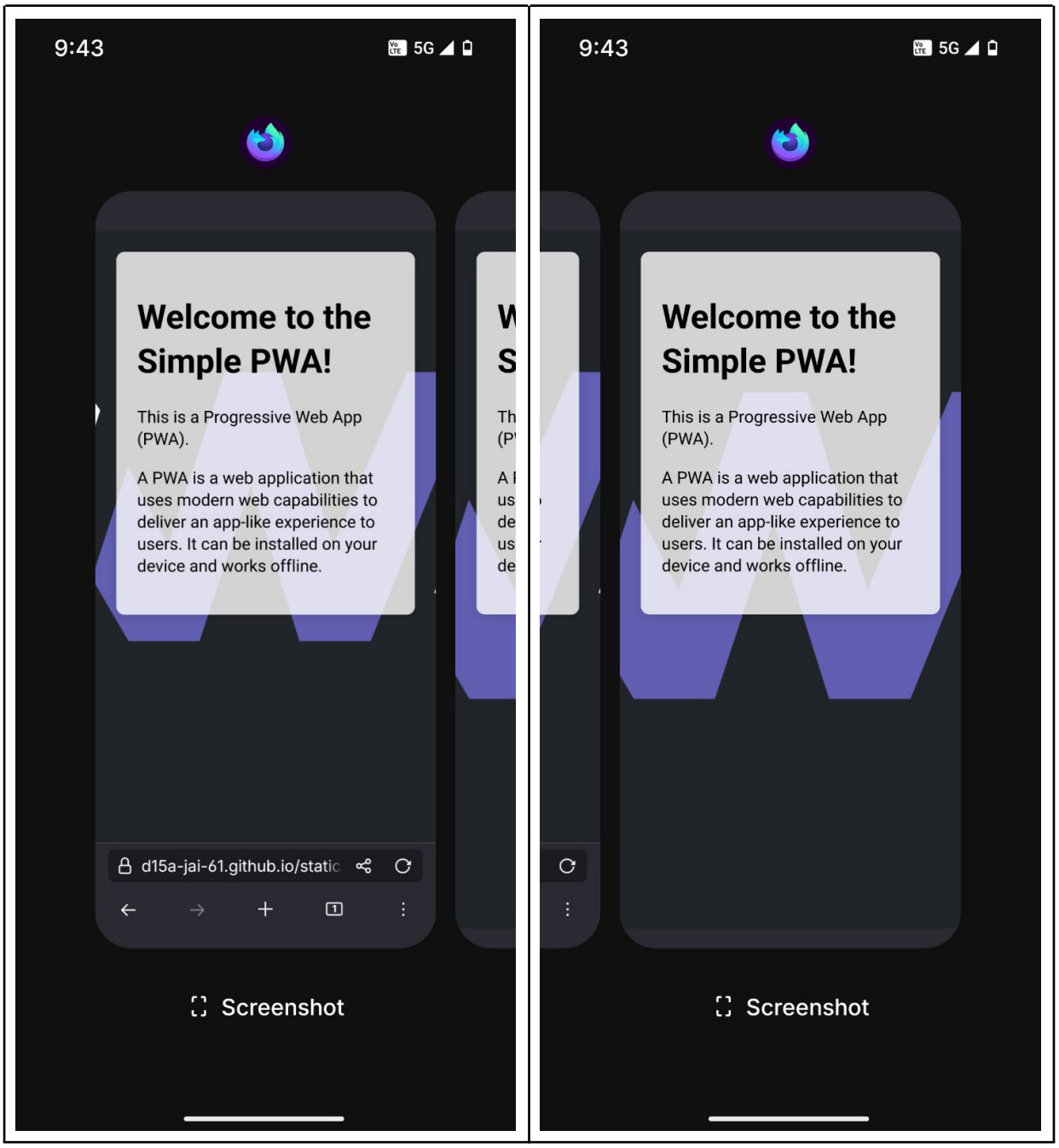
```
event.request.url);
                return response;
            }
            console.log('[Service Worker] Fetching from network:', event.request.url);
            return fetch(event.request);
        })
    );
});

// Sync event - retry offline form submissions
self.addEventListener('sync', (event) => {
    if (event.tag === 'sync-form') {
        console.log('[Service Worker] Sync event triggered: Submitting offline form
data...');
        event.waitUntil(syncFormData());
    }
});

function syncFormData() {
    return getFormDataFromIndexedDB().then((formData) => {
        if (formData) {
            console.log('[Service Worker] Syncing form data...');
            console.log('[Service Worker] Form Data:', formData);

            // Instead of sending to an API, let's store it back in IndexedDB or localStorage
for now
            saveFormDataLocally(formData);

            // Clear the form data from IndexedDB after "syncing"
            clearFormDataFromIndexedDB();
        }
    });
}

function getFormDataFromIndexedDB() {
    return new Promise((resolve, reject) => {
        const request = indexedDB.open('offlineFormData', 1);
        request.onsuccess = function() {
            const db = request.result;
            const tx = db.transaction('formData', 'readonly');
            const store = tx.objectStore('formData');
            const data = store.getAll();  // Get all stored form data
            data.onsuccess = function() {
                if (data.result.length > 0) {
                    resolve(data.result[0]);  // Return the first form data entry
                } else {
                    resolve(null);  // No data found
                }
            };
            data.onerror = reject;
        };
        request.onerror = reject;
    });
}
```

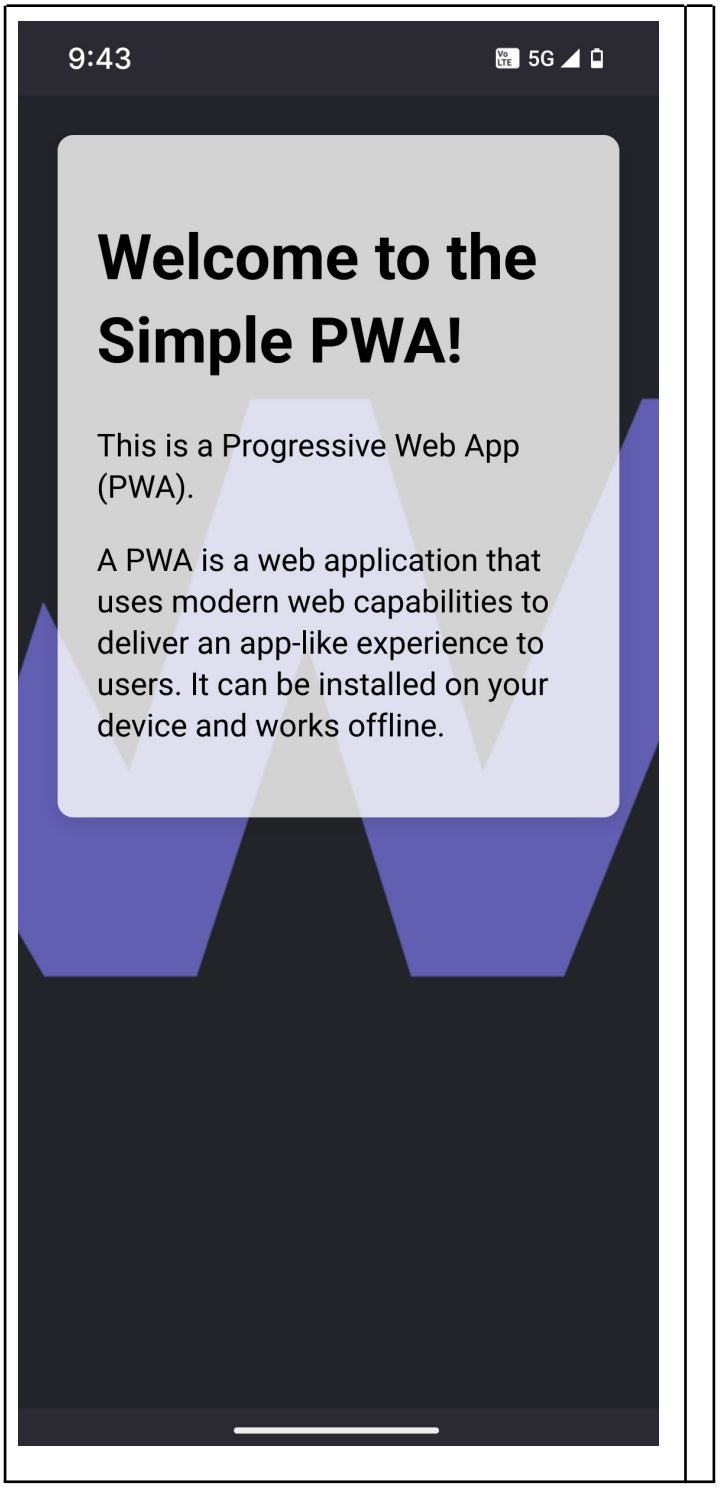
```
function saveFormDataLocally(formData) {
    // Save form data back into IndexedDB (simulating pushing data back into the app)
    const request = indexedDB.open('offlineFormData', 1);
    request.onsuccess = function() {
        const db = request.result;
        const tx = db.transaction('formData', 'readwrite');
        const store = tx.objectStore('formData');
        store.add(formData);  // Push data back into IndexedDB
        tx.oncomplete = function() {
            console.log('[Service Worker] Form data saved locally to IndexedDB:',
formData);
        };
    };
}

function clearFormDataFromIndexedDB() {
    const request = indexedDB.open('offlineFormData', 1);
    request.onsuccess = function() {
        const db = request.result;
        const tx = db.transaction('formData', 'readwrite');
        const store = tx.objectStore('formData');
        store.clear();  // Clear the form data from IndexedDB after syncing
        console.log('[Service Worker] Form data cleared from IndexedDB.');
    };
}

// Push event - handle push notifications (you can modify this part as per your needs)
self.addEventListener('push', (event) => {
    const options = {
        body: event.data.text(),
        icon: 'pwa-banner.png',  // Use pwa-banner.png for the notification icon
        badge: 'badge.png',      // Optional: You can use a badge if needed
    };

    event.waitUntil(
        self.registration.showNotification('New Content Available!', options)
    );
});

// Handle notification click
self.addEventListener('notificationclick', (event) => {
    event.notification.close();
    event.waitUntil(
        clients.openWindow('https://d15a-jai-61.github.io/static-pwa/') // Open your app's
homepage or a specific page
    );
});
```

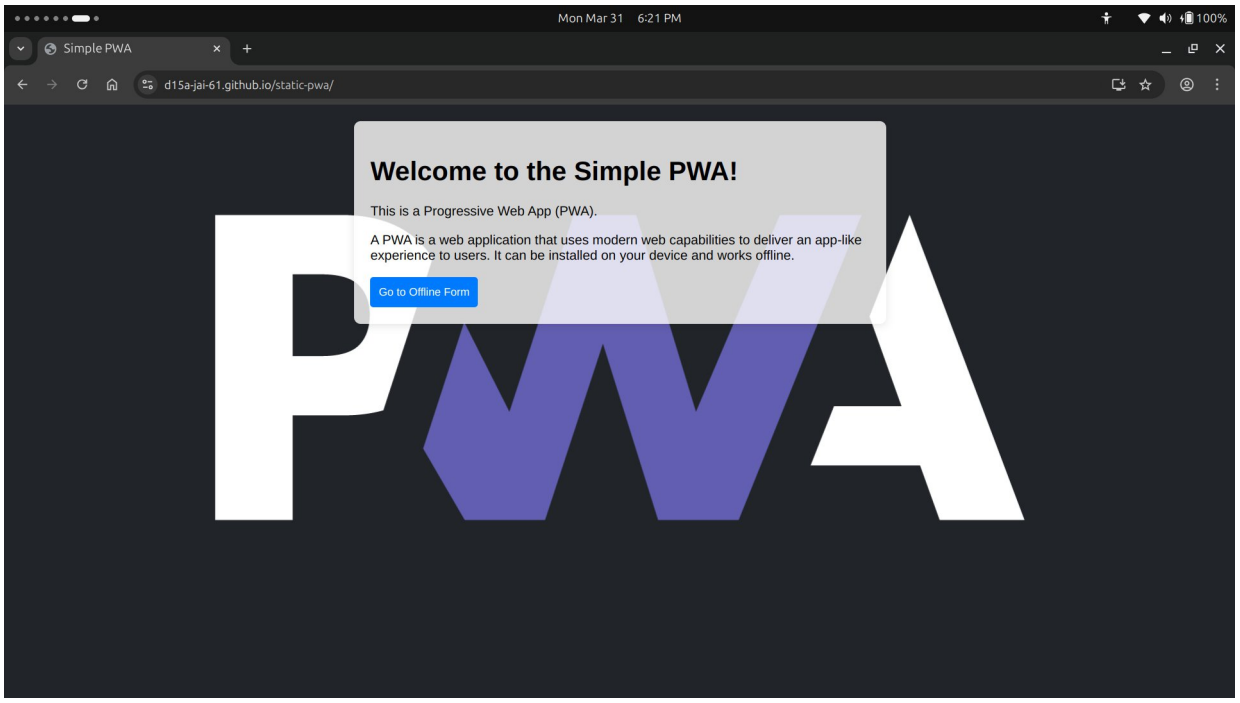
Output:

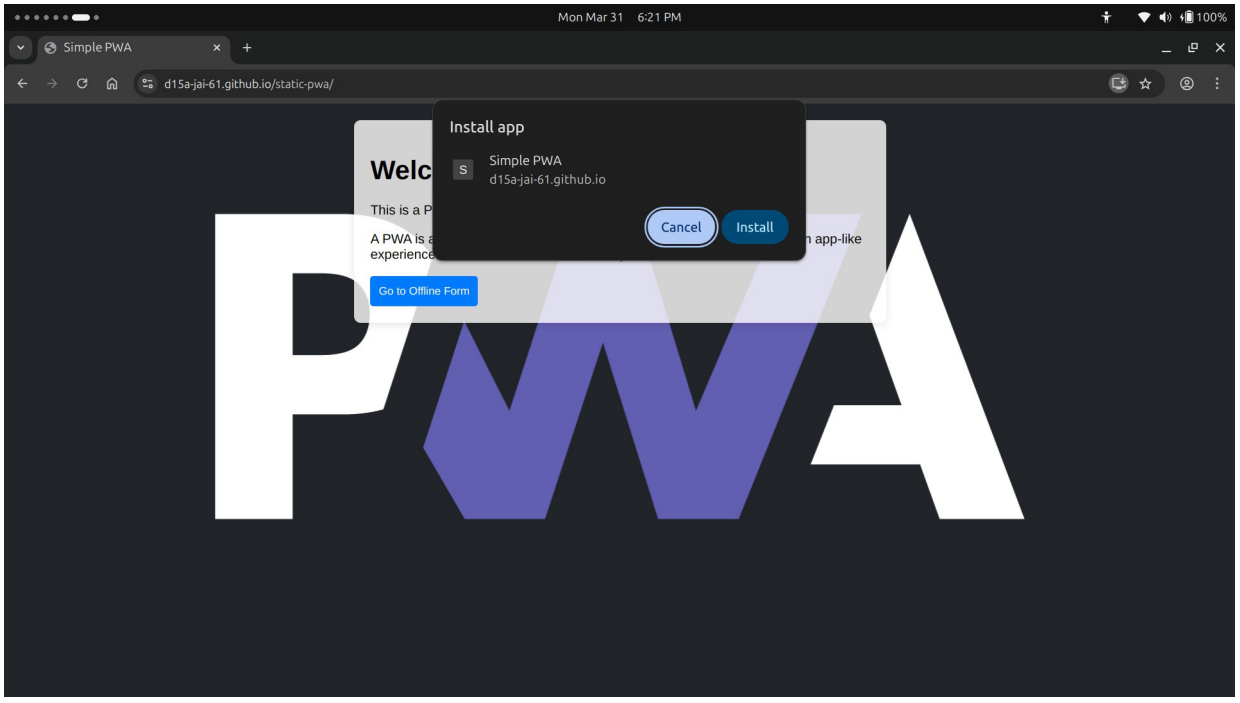

9:43
5G
Welcome to the Simple PWA!
This is a Progressive Web App (PWA).
A PWA is a web application that uses modern web capabilities to deliver an app-like experience to users. It can be installed on your device and works offline.
d15a-jai-61.github.io/static
9:43
5G
New tab
Bookmarks
Add
History
Downloads
Passwords
Extensions
Synchronise and sav...
Find in page
Translate page
Desktop site
Report broken site...
Add app to Home sc...
Save to collection
Print
Settings

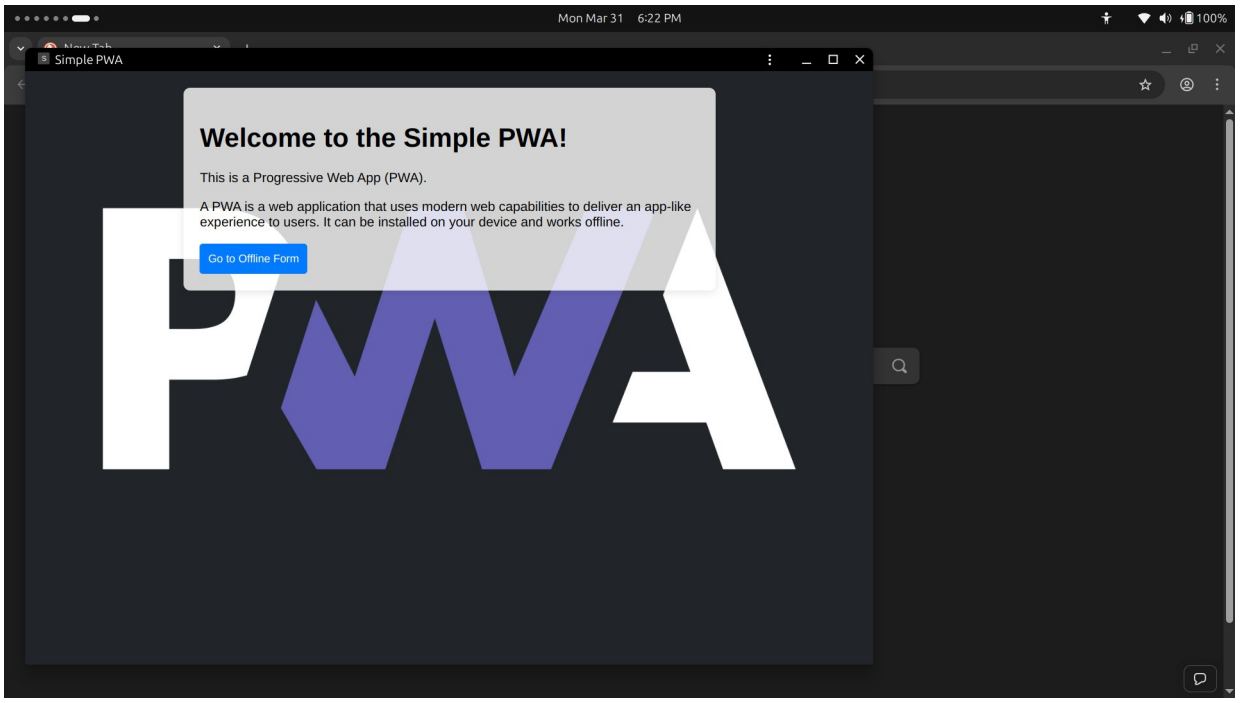

9:43
5G
Welcome to the Simple PWA!
This is a Progressive Web App (PWA).
A PWA is a web application that uses modern web capabilities to deliver an app-like experience to users. It can be installed on your device and works offline.
Firefox Nightly
Touch and hold the widget to move it around the home screen
PWA
1 × 1
Cancel
Add to home screen
9:43
5G
S
K
PWA

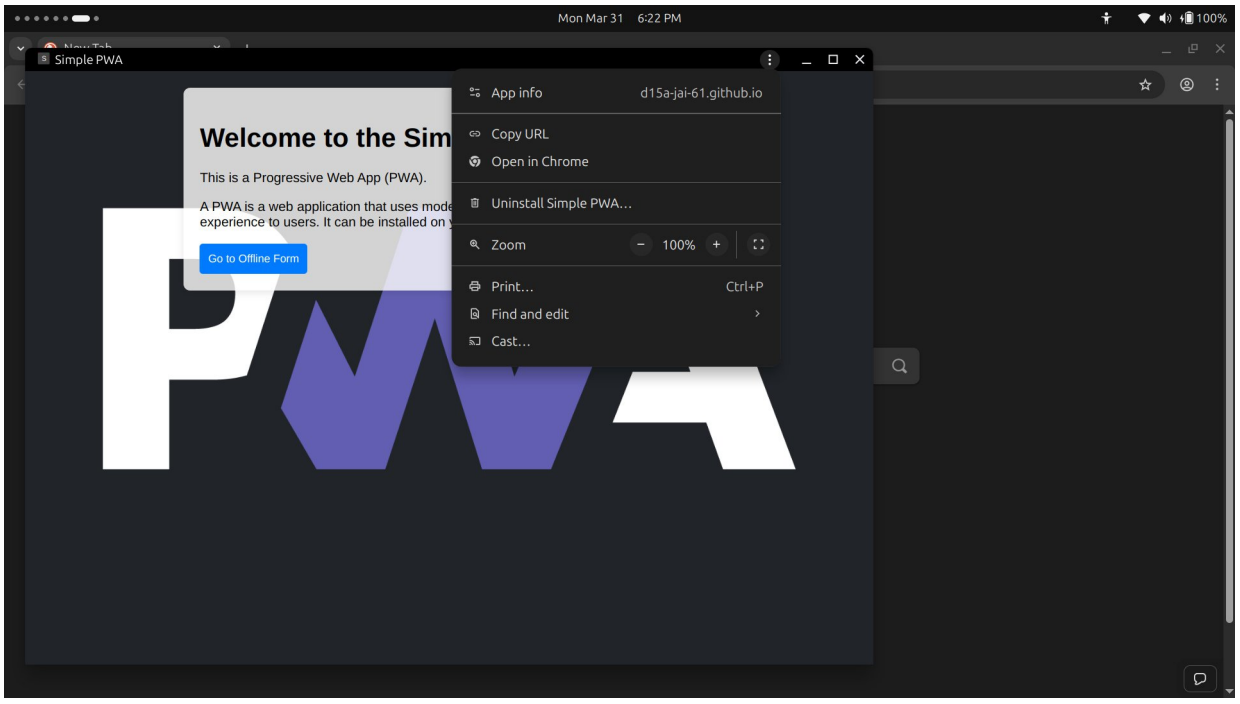

9:43
5G
Welcome to the Simple PWA!
This is a Progressive Web App (PWA).
A PWA is a web application that uses modern web capabilities to deliver an app-like experience to users. It can be installed on your device and works offline.
d15a-jai-61.github.io/static
Screenshot
9:43
5G
Welcome to the Simple PWA!
This is a Progressive Web App (PWA).
A PWA is a web application that uses modern web capabilities to deliver an app-like experience to users. It can be installed on your device and works offline.
Screenshot

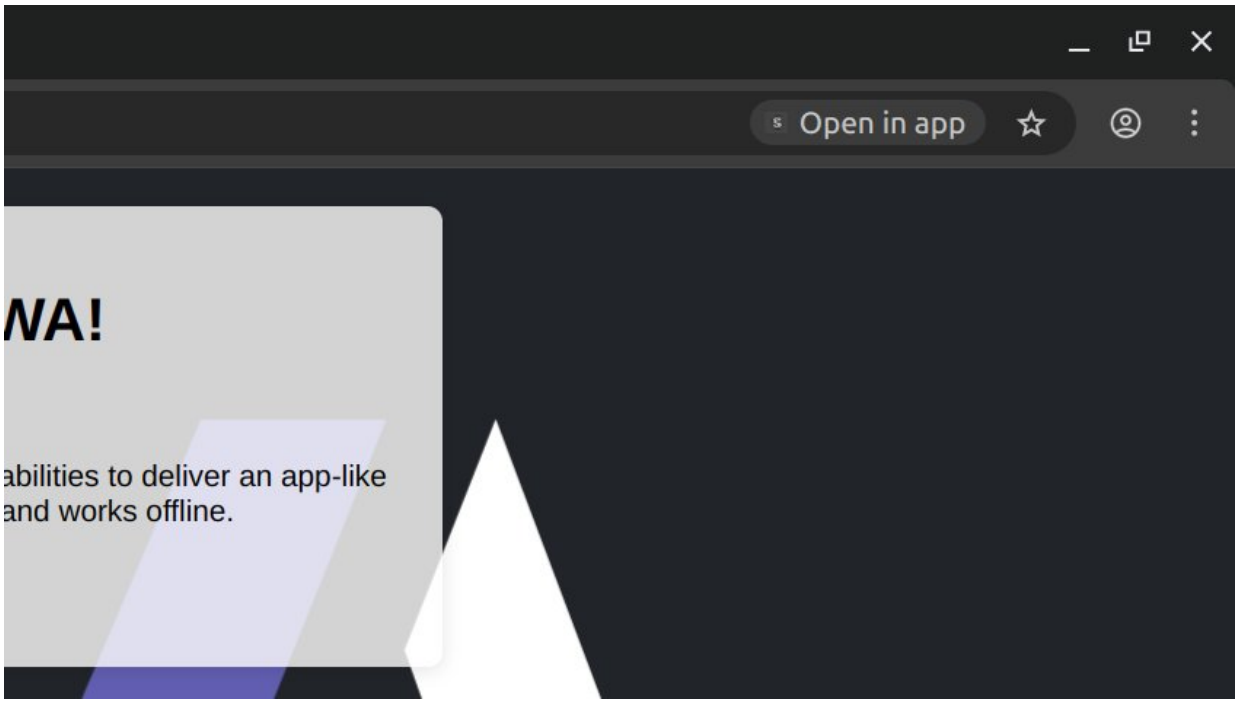

9:43
5G
Welcome to the Simple PWA!
This is a Progressive Web App (PWA).
A PWA is a web application that uses modern web capabilities to deliver an app-like experience to users. It can be installed on your device and works offline.

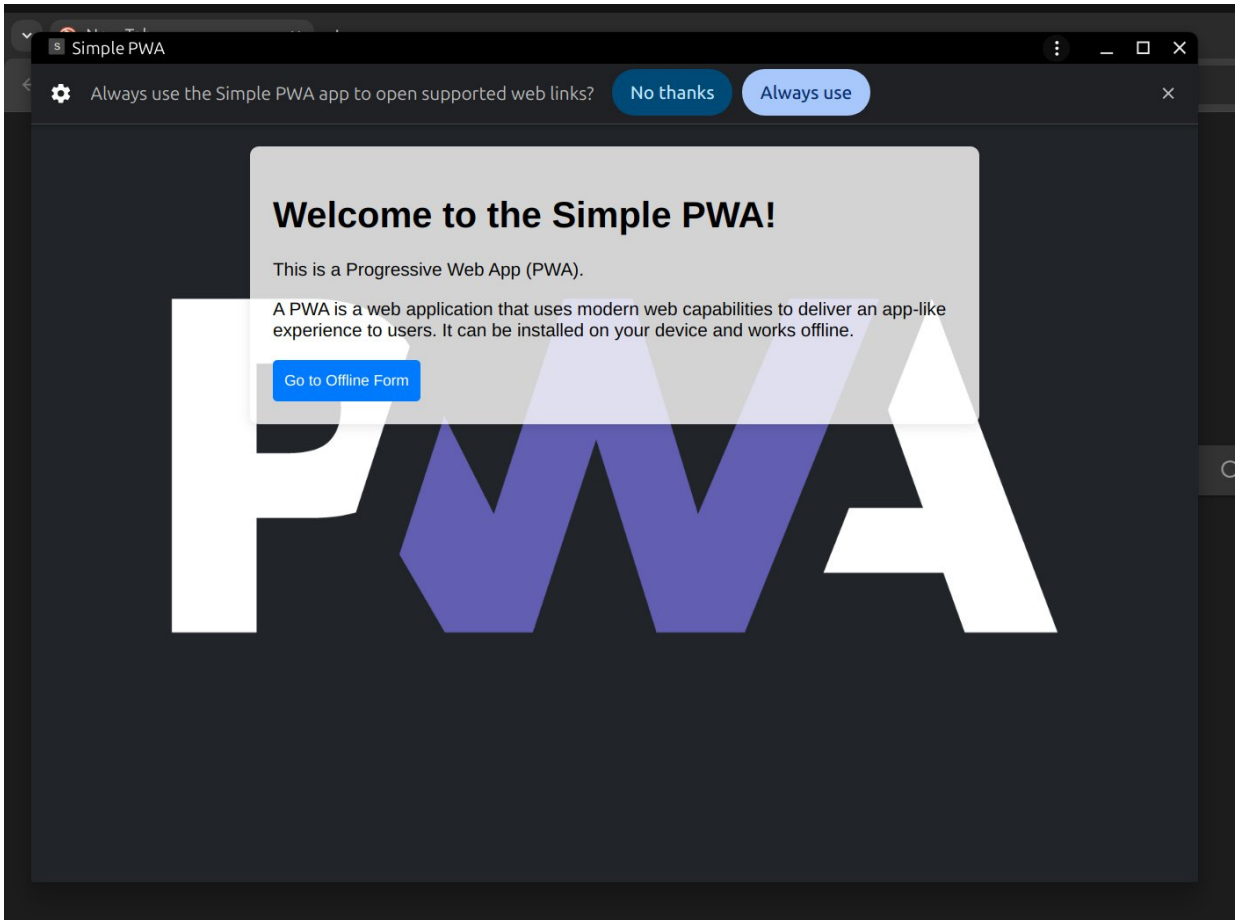

Mon Mar 31 6:21 PM
100%
Simple PWA
d15a-jai-61.github.io/static-pwa/
Welcome to the Simple PWA!
This is a Progressive Web App (PWA).
A PWA is a web application that uses modern web capabilities to deliver an app-like experience to users. It can be installed on your device and works offline.
Go to Offline Form
PWA


Mon Mar 31 6:21 PM
100%
Simple PWA
d15a-jai-61.github.io/static-pwa/
Install app
S
Simple PWA
d15a-jai-61.github.io
Cancel
Install
Welc
This is a P
A PWA is a
experience
app-like
Go to Offline Form
PWA


Mon Mar 31 6:22 PM
100%
Simple PWA
Welcome to the Simple PWA!
This is a Progressive Web App (PWA).
A PWA is a web application that uses modern web capabilities to deliver an app-like experience to users. It can be installed on your device and works offline.
Go to Offline Form


Mon Mar 31 6:22 PM
100%
Simple PWA
Welcome to the Sim
This is a Progressive Web App (PWA).
Go to Offline Form
App info
d15a-jai-61.github.io
Copy URL
Open in Chrome
Uninstall Simple PWA...
Zoom
100%
Print...
Ctrl+P
Find and edit
Cast...


Open in app
NA!
abilities to deliver an app-like
and works offline.


Simple PWA
Always use the Simple PWA app to open supported web links?
No thanks
Always use
Welcome to the Simple PWA!
This is a Progressive Web App (PWA).
A PWA is a web application that uses modern web capabilities to deliver an app-like experience to users. It can be installed on your device and works offline.
Go to Offline Form
PWA