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 :
By now you must have heard this buzz word called “ Progressive Web App ”. Let me give you a simple definition about PWAs. Progressive Web Apps are Web Apps which combines the best features of Web and Native Apps. It is progressive because it is constantly progressing.

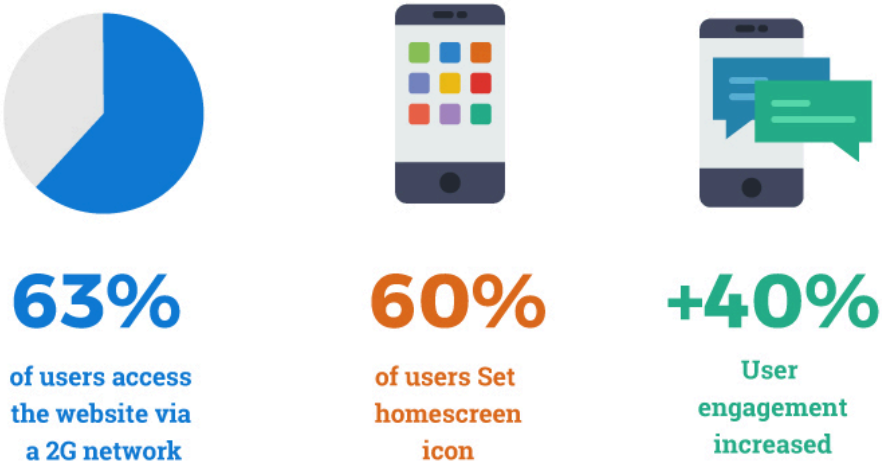
Why Progressive Web Apps?
Now let’s talk about the “WHY”.

Frances Berriman and Alex Russell came up with a concept called Progressive Web Apps? Before that let’s understand what problems does it solve.

Problems with Native Apps?
We all have used Android or iOS apps on our smartphones. We use them for all kinds of thing. But while installing any Android/iOS apps we go through these problems :

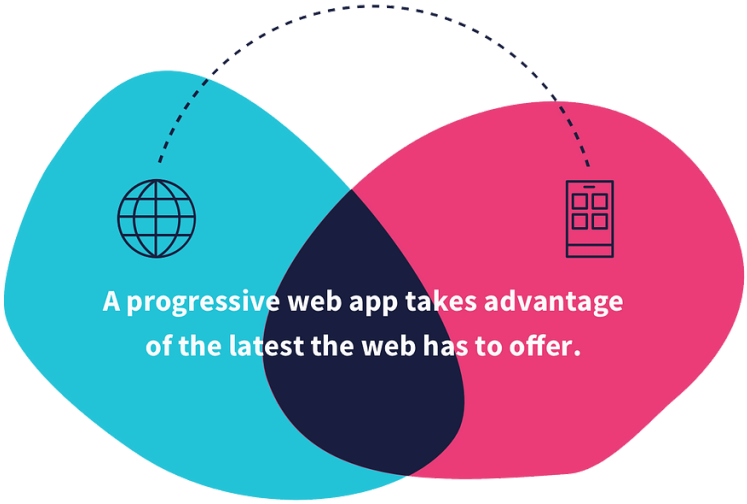
1. Is this app worth downloading?
2. Do I have enough space?
3. My available data is not sufficient.

One recent survey shows that people are turning away from Android/iOS apps, because not all app experiences are satisfying or worthwhile. Some people simply don’t want any more apps on their phone, some even hesitate to download any app.
If you take a look at the apps installed on your mobile right now there might be at least a dozen apps that you do not use regularly. Sometimes apps only works good when the phone has an active internet connection.



The irony is that most of the apps have a fully responsive website performing the same functions. So why waste your precious disk space and your internet data on your smartphone by

installing the native app? The average size of apps that we install from play store/app stores would range from 30–200MB. Moreover, these app needs to updated every week! But Progressive Web Apps are within some KBs and are automatically updated. Thanks to service worker.
What if a website can do that and much more than a Native app? This is what Progressive Web Apps (PWA) are trying to accomplish.



In short, Progressive web apps combine everything that is great about a native mobile application with everything that is great about a mobile website.
Some other ways I like to describe them:
“The best of the web, plus the best of native apps”
Or, in Alex’s words:
“Just websites that took all the right vitamins”

Features of PWAs:

- Progressive — The word progressive means it works for every user, regardless of browser choice because they're built with progressive enhancement as a core tenet.
- Responsive — Automatically adjustable to any form: desktop, mobile, tablet etc.
- Load Time — Progressive Web Apps are instantly available
- App-like — Feels like a mobile app with app-style interactions since it's built on the app shell model.
- Fresh — Always up-to-date so you do not need to update it again and again like any other Android/iOS apps.
- Safe — Served via HTTPS to ensure content is securely delivered
- Engaging — Features like push notifications, etc. makes it very engaging.
- Installable — Allows users to install the website as an app on their home screen without taking the user to an app store.
- Linkable — Easily shared via a URL and does not require complex installation.

Benefits of making a Progressive Web App rather than building a fully functional Android App?

- Cost Effective — For an app publisher, the biggest advantage is the cost saving in terms of app development and maintenance. Because it is assumed that making a website is a lot easier than making an Android App.
- Cross Platform — Unlike any other apps, Progressive Web Apps are not restricted to any specific platform. That means you do not need to develop separate versions of app for different platforms.

IGNITE ONLINE

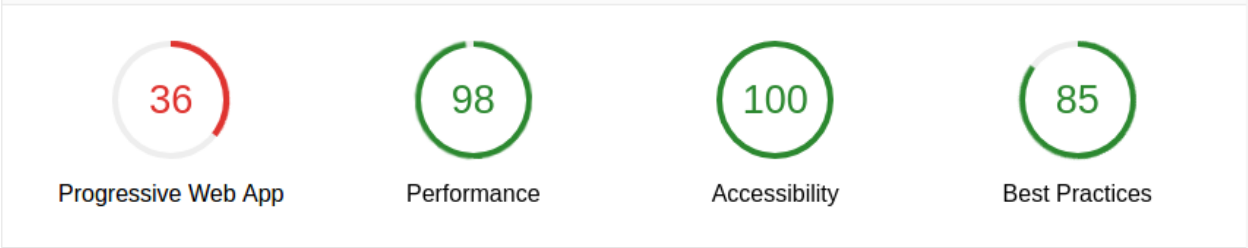
FEATURE	PROGRESSIVE WEB APP	NATIVE APP
FUNCTIONS OFFLINE	✓	✓
MOBILE-SPECIFIC NAVIGATION	✓	✓
PUSH NOTIFICATIONS	✓	✓
HOME SCREEN ACCESS	✓	✓
NO DOWNLOAD REQUIRED	✓	✗
BYPASSES THE MARKETPLACE	✓	✗
LINKABLE AND SHAREABLE	✓	✗
INDEXED BY GOOGLE	✓	✗
LOW DATA REQUIREMENTS	✓	✗
REQUIRES NO UPDATES	✓	✗

Some Popular Companies that Do Progressive Web Apps

- Ola
- Flipkart
- pinterest
- Twitter

- Alibaba
- BookMyShow
- MakeMyTrip
- OLX
- The Weather Channel
- Forbes
- JioCinema
- Trivago

Now Let's create a Progressive Web App:
Here, I'll show you how I've created a Progressive Web App for simple blog. You can get the codes here. Now that we have a basic website we can start turning it into a progressive web app. To do this we need to add a few things to it which I'll go through as we need them.
Testing your PWA
To check if your site is working as a PWA you can use Lighthouse. Lighthouse is a chrome extension that will tell you if your site is a good PWA and if it isn't how to improve it.
Once installed open up your website and click on the Lighthouse icon in the top right of your browser then Generate Report.



Lighthouse results before I started working on the PWA parts of the site
Make an app icon
Your site is going to be on a home screen, you need some sort of icon to represent it. Here I've used a downloaded logo from internet.

Register Service Worker
Add service worker <script> to index.html:

A service worker is another file we add to our project, it will allow the site to work offline. It is also a requirement of a PWA, so we need one.

service-worker.js

```
self.addEventListener("install", function (event) {
    event.waitUntil(preLoad());
});

self.addEventListener("fetch", function (event) {
    event.respondWith(checkResponse(event.request).catch(function () {
        console.log("Fetch from cache successful!")
        return returnFromCache(event.request);
    }));
    console.log("Fetch successful!")
    event.waitUntil(addToCache(event.request));
});
self.addEventListener('sync', event => {
    if (event.tag === 'Sync from cache') {
        console.log("Sync successful!")
    }
});
self.addEventListener('push', function (event) {
    if (event && event.data) {
        var data = event.data.json();
        if (data.method == "PushMessageData") {
            console.log("Push notification sent");
            event.waitUntil(self.registration.showNotification("RED
STORE", {body: data.message}))
        }
    }
})

var filesToCache = [
    '/index.html',
    '/product_details.html',
    '/products.html',
    '/cart.html',
    '/account.html'
];
```

```
var preLoad = function () {
    return caches.open("offline").then(function (cache) { // caching index
and important routes
        return cache.addAll(filesToCache);
    });
};

self.addEventListener("fetch", function (event) {
    event.respondWith(checkResponse(event.request).catch(function () {
        return returnFromCache(event.request);
    }));
    event.waitUntil(addToCache(event.request));
});

var checkResponse = function (request) {
    return new Promise(function (fulfill, reject) {
        fetch(request).then(function (response) {
            if (response.status !== 404) {
                fulfill(response);


            } else {
                reject();
            }
        }, reject);
    });
};

var addToCache = function (request) {
    return caches.open("offline").then(function (cache) {
        return fetch(request).then(function (response) {
            return cache.put(request, response);
        });
    });
};

var returnFromCache = function (request) {
    return caches.open("offline").then(function (cache) {
        return cache.match(request).then(function (matching) {
            if (! matching || matching.status == 404) {
```

```
                return cache.match("index.html");
            } else {
                return matching;
            }
        });
    });
};
```

Web App Manifest
To support add to homescreen feature, we need to create a manifest file.
“ The web app manifest provides information about an application (such as name, author, icon, and description) in a JSON text file. The purpose of the manifest is to install web applications to the homescreen of a device, providing users with quicker access and a richer experience.“
The simplest way I did this was by using this online icon generator tool. Feed it your shiny new icon and it will spit out a bunch of resized versions and some HTML code.

- Download the file it gives you and unzip it.
- Put the icons in a folder next to the rest of your site.
- Add the code it gave you to the <head> of you index.html file
- Make sure the path to the icons is right. I put them all in a sub folder so had to add “icons/” to each line.

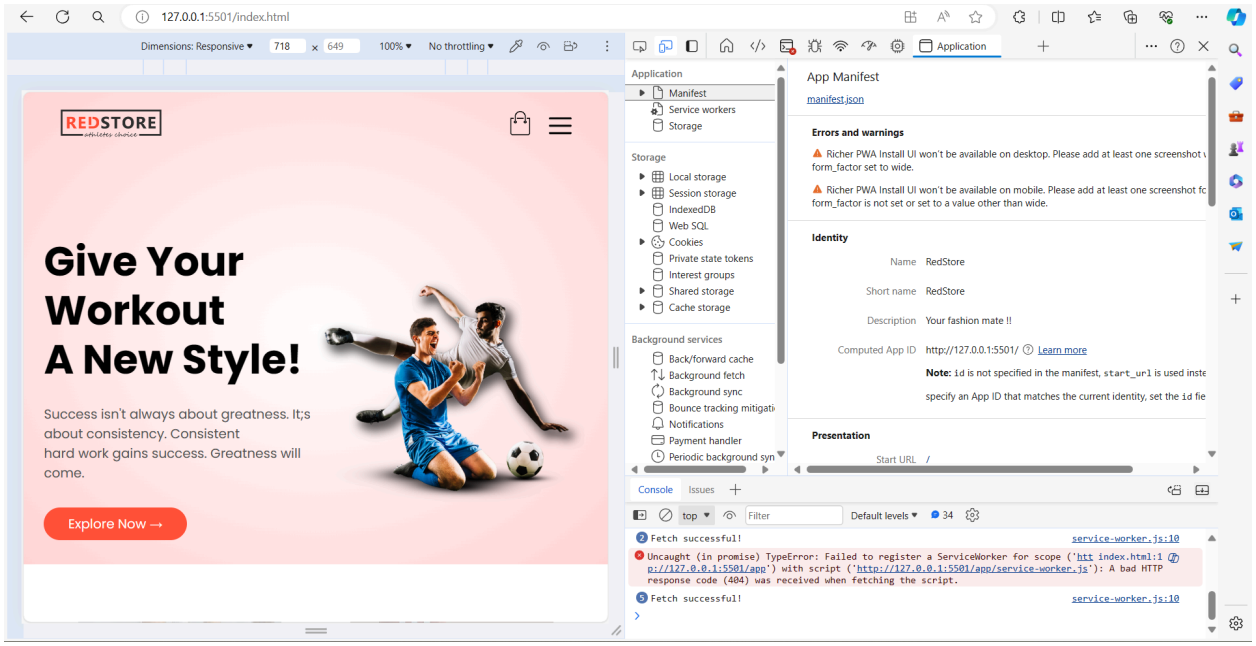
manifest.json

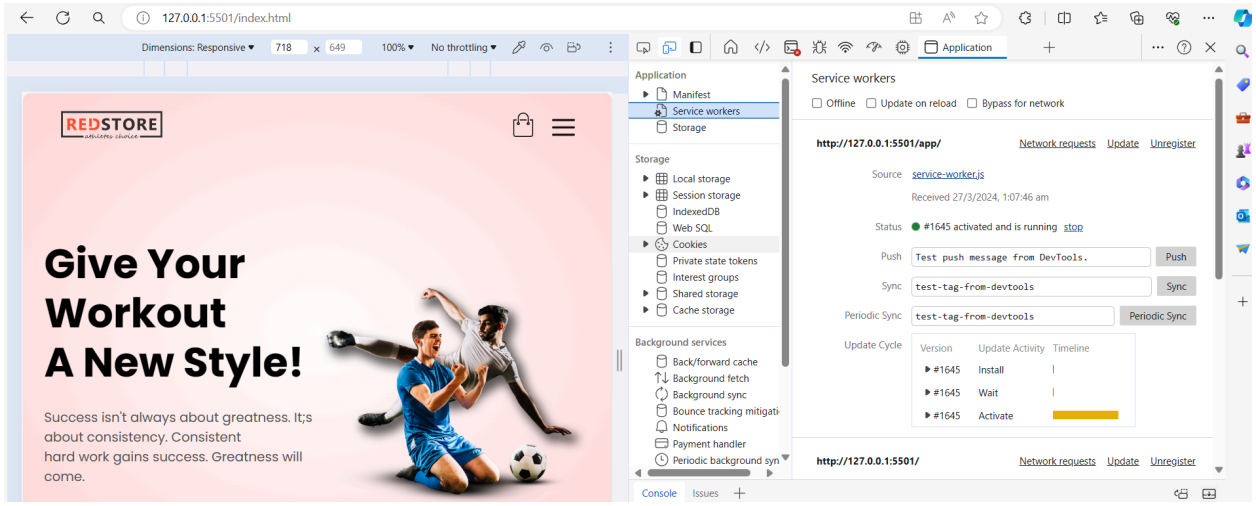
```
{
    "name": "RedStore",
    "short_name": "RedStore",
    "start_url": "/",
    "background_color": "#212529",
    "description": "Your fashion mate !!",
    "display": "fullscreen",
    "theme_color": "#212529",
    "icons": [
        {
            "src": "/icon2.png",
            "sizes": "512x512",
            "purpose": "any"
        },
        {
            "src": "/icon1.png",
            "sizes": "192x192",
```

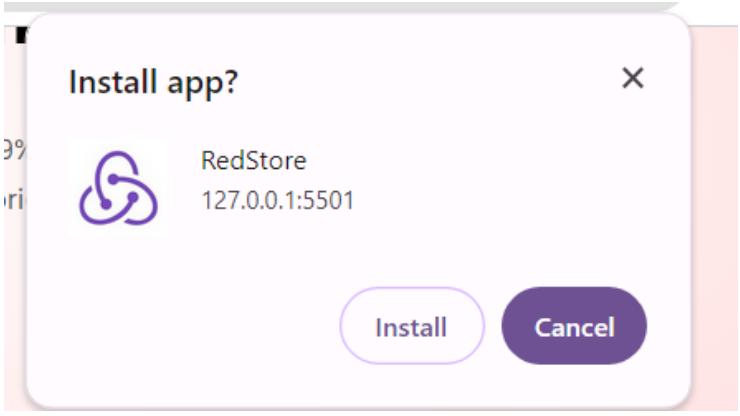
```
            "purpose": "any"
        },
        {
            "src": "/icon1.png",
            "sizes": "192x192",
            "purpose": "maskable"
        },
        {
            "src": "/icon1.png",
            "sizes": "192x192",
            "purpose": "maskable"
        }

    ],
    "prefer_related_applications": true,
    "related_applications": [
        {
            "platform": "play",
            "id": "com.google.samples.apps.iosched"
        }
    ]
}
```

Now you have a manifest that was created by the icon generator tool, but we're going to add a little bit more to it.
Head over to a web app manifest generator and start filling in the info about your site. After doing all these things my manifest ended up looking like this.

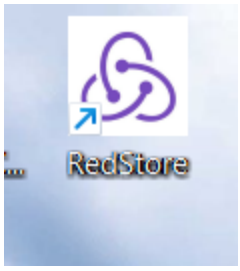
Name : Palak Najawan EXPT : 7 Rollno. 43







Name : Palak Najawan EXPT : 7 Rollno. 43



Conclusion : By following these steps and providing the necessary metadata in our Web App Manifest file, you can successfully enable the "Add to Home Screen" feature for our eCommerce PWA, improving user accessibility and engagement.