ASSIGNMENT NO. 2
MAD and PWA Lab

Aim: To study the requirement for progressive web application for E-commerce using the concept of service worker, Web app Manifest and framework tools.

Progressive Web App

At its core, a Progressive Web App (PWA) is a web application that uses the latest web capabilities to deliver a native app-like experience to users. It is the technology that will bring consistency between web and native apps and replace both with a single instance.

Progressive Web Apps are:

1. Reliable. They load instantly, regardless of the network state.They eliminate dependence on a network connection, ensuring an instant and reliable user experience, even with bad or nonexistent internet connection.
2. Incredibly fast. If your site is slow, customers leave, making you miss out on potential sales. PWAs load extremely fast and respond quickly to user interactions. Improved website performance positively affects conversion, user experience, and retention rates.
3. Highly engaging. They provide a true native app experience, including web capabilities such as the option to send Push Notifications and appear on the homescreen.

Importance of PWA:

1. Better user adoption by installing on home screen: Progressive Web Apps provide a highly engaging user experience, with the same capabilities as native apps. Progresssive Web Apps can be installed on the home screen, making them directly accessible to users.

In a typical month, the average amount of apps downloaded by a smartphone user is close to zero. People don’t download apps.

With a Progressive Web App, ‘downloading’ is as simple as visiting a web link and accepting a prompt. From a user perspective, the website will transform into a ‘real’ app – so not a browser shortcut – by being placed on the homescreen. This achieves the same result as downloading from an app store, but without the hassle of going through the app store process. Additionally, after the PWA is installed on the homescreen, no further updates are required. PWA’s are automatically updated in the background, assuring your customers always get to see the most recent content.

2. Boosting user engagement with push notifications: Similar to a native app, PWA offers the capability to send Push Notifications to mobile devices. With a very intuitive way, businesses can reach their audience where they are the most: directly on their mobile device, creating a highly engaging user experience.

These small messages have a big impact: they grab attention, always. Compared to traditional marketing, such as emails, Push Notifications have the big advantage that they appear directly on a persons' phone; they stand out from the crowd and most importantly, a business does not need to make a lot of effort to collect email addresses or other information before they can engage with their audience.

3. Reduced development costs, faster innovation & continuous delivery: Businesses with an omnichannel, multiplatform strategy are currently building, maintaining and promoting up to 4 systems: their (responsive) website, their Android app, their iOS apps and in some cases also Windows 10 native apps.
A Progressive Web App completely eliminates the need for development, maintenance and marketing of any platform other than the PWA website. This provides a unique opportunity to serve all channels from one platform, which is built, maintained and serviced by one team. It reduces costs and time-to-market drastically, while still serving the same user experience and capabilities to all customers, and keeping the multi-channel strategy.

4. Excellent instant performance.
Internet users are always demanding more speed, especially with the massive rise of mobile-first and even mobile-online usage. Research shows that 53% of visitors leave a website after just 3 seconds page load and conversion rates decrease with 21,8% on each 1 second delay in page speed.

Progressive Web Apps are by design built for extremely smooth and fast user experiences. Capitalising on the speed benefits of the web and 'native-style' client-side caching, it outperforms both on page loads. PWAs can load within just 1 second, creating a true 'instant' speed experience, engaging the user right from the start.

Additionally, Progressive Web Apps are using significantly less disk space – on both the business' server and the user's device – providing benefits such as faster loading times, less data usage and less required storage space.

5. PWAs can improve your SEO.
Because a PWA is web-based, everything is discoverable by search engines. Any content within a PWA can be linked to, shared and ranked by Google and Bing. This contrasts native apps, which don't allow to be crawled by search engine robots. By breaking down the barriers between web and native, users can be directly linked to the most sophisticated digital products, on any device, at any time.

Super fast loading times, reduced bouncing rates, minimum data usage and highly engaging experiences are by default boosters for SEO rankings. With PWA offering advantages like these out-of-the-box, it's logical to reason that using a PWA will automatically result in higher SEO rankings. If done right, this is absolutely true. Google will rank well-designed PWA's higher than any other regular website.

However, technical SEO is very important. It's key to make sure the technology of Progressive Web Apps is served in the right way. Progressive Web Apps make use of the flexibility of JavaScript, which means Google sees the single published page as a JavaScript site. Google crawls PWA sites just like it would crawl an AJAX or JavaScript site, but the ability of search engines to correctly discover, crawl, and accurately index content — which is heavily reliant on JavaScript — has historically been poor.

Luckily, there is no need to worry. Developers building PWAs need to be aware of how to optimize the site to ensure the page gets indexed appropriately, which can easily be done with techniques such as server side rendering — providing PWAs the right architecture to get the top SEO rankings they deserve.

6. Increased conversion rates.
It stands to reason that a better user experience results in higher conversion rates. If friction and frustrations are removed from an online store, it becomes easier for customers to achieve their goals – and therefore the business' goals.

PWAs present huge opportunities to deliver better experiences, in every aspect. Everything about PWAs is faster, leaner, slicker and more efficient, and it's felt by the user. From instant background loading onto their home screen to full screen browsing, instant page loads, and minimal usage of device space. The user has a much better experience, completes their tasks, and will happily return again in the future — whether or not triggered by a highly engaging push notification.

Technical components of PWA
1. Web App Manifest:
The web app manifest is the first component of the PWA. It is a simple JSON file that controls a user's application. Usually, it is named "manifest.json". It is the most important component for the presence of PWA. When you first connect PWA to a network, a mobile browser reads the "manifest.json" file and stores it locally in cache memory.
When there is no network access in PWA, the mobile browser uses the application's cache memory to run the PWA program while offline.

The "manifest.json" file helps the PWA to give a look of a native app. With the help of the manifest.json file, the PWA developer can control how the application is presented to the user mobile screen. The PWA developer can also set a theme for the mobile's splash-screen and the application's address bar.

The "manifest.json" file allows the PWA developer to search for a centralized location for the metadata of the web application. The JSON file defines the links to icons and icon sizes, the full and abbreviated name of an app, types, background color, theme, locations, and other visual details that are required for an app-like experience.

2. Service Worker
A service worker is a web worker. It is a JavaScript file that runs aside from the mobile browser. In other words, it is another technical component that promotes the functionality of PWA. The service worker retrieves the resources from the cache memory and delivers the messages.

It is independent of the application to which they are connected, and has many consequences:

- The service worker does not block the synchronized XHR, so it cannot be used in local storage (It is designed to be completely asynchronous).
- It can receive information from the server even when the application is not running. It shows notifications in the PWA application without opening in the mobile browser.
- It cannot directly access the DOM. Therefore, the PostMessage and Message Event Listener method is used to communicate with the webpage. The PostMessage method is used to send data, and the message event listener is used to receive data.

Service worker lifecycle
The service worker lifecycle is the most complex part of the PWA. There are three stages in the lifetime of a service worker:

- Registration
- Installation
- Activation

Framework tools:

Webpack :- webpack-pwa-framework

Webpack is a static module bundler for modern JavaScript applications. Without such bundlers, running JS files in a browser may cause issues with loading too many scripts in large .js files. This dev tool is recommended to solve the scaling problem for complex PWA storefronts. Webpack Module Bundler aimed at building and managing dependency graphs between CSS elements, stylesheets, fonts, images, and other

JavaScript assets. The graphs give developers better control over assets processing, ease code-splitting, eliminate dead assets, and reduce the server calls.

Among the drawbacks of Webpack are unfriendly documentation, painful source code, and steep learning curve.

Lighthouse : - lighthouse pwa-pwa-framework

Once your PWA is created, you want to measure its performance, SEO, accessibility, and other important benchmarks. For this purpose, Google provides an awesome analytic tool – Lighthouse. that offers a set of metrics to test the app to test the application and guide you in creating PWAs with an immersive app-like experience for your visitors.

The tool significantly reduces the need to carry out manual testing to audit your web application for PWA features. Lighthouse enables developers to automate a series of tests against a given URL and fleetingly generate an exhaustive report which gives them a clue about how well the PWA is performing, as well as what bugs it has. The recorded data can be used to improve the overall performance of the newly created web application.

Lighthouse can either be run from the Chrome DevTools (from its Audits tab) or automated through the command line, as a Node module. Alternatively, it can be installed and run as a Chrome extension.

Conclusion: Therefore, these are the requirements for progressive web application for E-commerce using the concept of service worker, Web app Manifest and framework tools.