Yash Rahate, D15, B, 44

MPL Assignment 02

05/03/25

Page No
Date

Q1) Define Progressive Web App (PWA) and explain its significance in modern web development. Discuss the key characteristics that differentiate PWAs from traditional mobile apps.

→ • A Progressive web App is a type of web application that works like a mobile app but runs in a browser. It can be installed on a device, works offline, and provides a fast and smooth user experience.

• Significance of PWA in Modern Web Development:

(1) Cross Platform Compatibility - Works on both mobile and desktop with a single codebase.

(2) Offline Support - Can function without the internet using cached data.

(3) Fast Performance - Loads quickly, even on slow networks.

(4) No App Store Required - Users can install it directly from the browsers.

(5) Lower Development cost - One PWA can replace separate Android and iOS apps.

• Key Differences Between PWA and Traditional Mobile Apps:

Feature	PWA	Traditional mobile app
Installation	Direct from browser	Download from App store
Internet Required	Works offline with caching	Usually requires internet
Performance	Fast with service workers	Faster but needs installation
Updates	Automatic, no app store approval	Manual updates needed
Development cost	Lower (one codebase for all)	Higher (separate app for each platform

2) Define responsive web design and explain its importance in the context of Progressive Web Apps. Compare and contrast responsive, fluid and adaptive web design approaches.

→ Definition of Responsive Web Design:
Responsive Web Design (RWD) is a technique that makes web pages adjust automatically to different screen sizes and devices. It ensures a good user experience on mobiles,

tablets and desktops without needing separate versions of a website.

- Importance of Responsive Design in PWAs
  1. Better User Experience – PWAs work smoothly on any device.
  2. Faster Load Time – Optimized design improves speed.
  3. SEO Benefits – Google ranks responsive sites higher
  4. Cost-Effective – No need to build multiple versions for different screens.

- Comparison of Web Design Approaches:

Approach	How it work	Pros	Cons
Responsive	Uses flexible grids and CSS media queries to adjust layout.	works on all devices improves SEO	can be complex to design
Fluid	Uses percent-based widths instead of fixed pixels, so elements resize smoothly.	Works well on different screen sizes, easy to implement.	Less control over layout on large screens.

Approach	How it Works	Pros	Cons
Adaptive	Uses fixed layouts that change at specific breakpoints	Optimized for known screen sizes.	More effort required to design for each screen size.

Key Differences:
- Responsive adapts dynamically to all screens
- Fluid resizes smoothly but may not be fully optimized.
- Adaptive loads different layouts based on device type.

Responsive design is best for PWAs because it ensures a seamless experience on all devices.

8 B) Describe the lifecycle of Service workers, including registration, installation and activation phases.

→ Lifecycle of Service Workers is a script that runs in the background and helps a web app work offline, load faster and send push notifications. Its lifecycle has three main phases:

Page No.
Date

1) Registration Phase

- The browser registers the Service Worker using JavaScript.

Code Example:

```
if ('serviceWorker' in navigator) {
navigator.serviceWorker.register('/sw.js')
  .then(() => console.log('Service Worker
    Registered'))
  .catch(error => console.log('Registration
      Failed:', error));
}
```

This tells the browser to install and activate the Service Worker.

2) Installation Phase

- The Service Worker downloads necessary files (HTML, CSS, JS) and stores them in cache.
- If successful, it moves to the activation phase.

Code Example:

```
self.addEventListener('install', event => {
  event.waitUntil(
  caches.open('app-cache').then(cache => {
    return cache.addAll(['/', '/index.html',
'/styles.css']);
  }));});
```

- This ensures the app loads even without the internet.

③ Activation Phase

- The old service Worker is replaced with new one.
- Unused cache files from the previous version are deleted.

Code Example:

```
self.addEventListener('activate', event => {
  event.waitUntil(
    caches.keys().then(keys => {
      return Promise.all(keys.map(key => {
        if (key !== "app-cache") {
          return caches.delete(key);
        }
      }));
    })
  );
});
```

The service worker is now fully active and controls network requests.

④ Fetch & Sync (Final Step)

Once activated, the service worker intercepts network requests, serves cached files and syncs data when the internet is available.

This life cycle makes PWAs faster, more reliable and capable of working offline.

Page No.
Date

Q4) Explain the use of IndexedDB in the Service Worker for data storage.

→ • Use of IndexedDB in Service Worker for Data Storage

IndexedDB is a browser database that stores large amounts of structured data like JSON objects. It helps PWAs work offline by saving and retrieving data efficiently.

• Why use IndexedDB in Service Workers?

1. Offline Support - Stores data when offline and syncs it later.
2. Efficient Storage - Saves structured data like user settings, cart items, or form inputs.
3. Faster Access - Retrieves data quickly without needing a network request.
4. Persistent Data - Data remains saved even after the browser is closed.

• How Service Workers used IndexedDB?

• Opening the Database

```
let db;
let request = indexedDB.open('MyDatabase', 1);
request.onsuccess = function(event) {
    db = event.target.result; };
```

- Creating a Store and Adding Data.

```
request.onupgradeneeded = function (event) {
  let db = event.target.result;
  let store = db.createObjectStore('User',
  {keyPath:'id'});
   store.add({id:1, name:'Yash', age:213});
};
```

- Fetching Data in Services Worker

```
let transaction = db.transaction(
'Users', "readonly");
let store = transaction.objectStore('Users');
letgetUser = store.get(1);

getUser.onsuccess = function () {
    console.log(getUser.result);
};
```