Name: Anya Manoj Madhavi
Class: D15B
Roll no: 29

05/05

# MPL Assignment 2

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

→ A Progressive Web App (PWA) 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
2. Offline Support
3. Fast Performance
4. No App Store Required
5. Lower Development Cost.

Key differences Between PWA & Traditional Mobile Apps:

| Features | 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 apps for each platform) |

PWA's combine the best of web and mobile apps, making them efficient and user friendly.

Q.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 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 website.

Importance of Responsive Design in PWA's:

① Better User Experience - PWA's 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 works | Pros | ~~Cons~~ Cons |
|---|---|---|---|
| Responsive | Uses flexible grids and CSS media queries to adjust layout. | Works on all devices | 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. |
| 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 types.

Responsive design is best for PWA's because it ensures a seamless experience on all devices.

Q.3. Describe the lifecycle of service workers, including its registration, installation and activation phases.

→ Lifecycle of service workers:
A service worker 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:

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 brower 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 the new one.
- Unused cache files from the previous version are deleted.

Final Step: Fetch & Sync

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

This lifecycle makes PWA's faster, more reliable and capable of working offline.

Q.4. Explain the use of Indexed DB in the System Service worker for data storage.

→ Use of Indexed DB in Service workers for Data Storage:
Indexed DB is a browser database that stores large amounts of structured data like JSON objects. It helps PWA's work offline by saving and retrieving data efficiently.

Why use Indexed DB in Service Workers?

① Offline Support - Stores data when offline and syncs it later.
② Efficient Storage - Saves structured data like user setting, 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 use Indexed DB?

Opening the Database

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

## Creating a Store & Adding Data

```
request.onupgradeneeded = function (event) {
let db = event.target.result;
let store = db.createObjectStore('users', { keyPath: 'id' });
store.add({ id: 1, name: 'John Doe', age: 25 });
};
```

## Fetching Data in Service Worker

```
let transaction = db.transaction('users', 'readonly');
let store = transaction.objectStore('users');
let getUser = store.get(1);

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