Maniya Motiramani
Maniya Motiramani
D16b / 36.

MPL assignment - 2.

Q.1 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 (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 & 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 browser.

key difference betn PWA & 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 (seperate apps for each platform)

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.

→ Defination of Responsive web Design:
Responsive web design (RWD) is a technique that makes web pages adjust automatically to different screen sizes and devices.

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.

compare.

Approach	How it works	pros	cons
Responsive	Uses flexible grids & CSS media queries to adjust layout	Works on all devices, improves SEO	Can be complex to design.
fluid	Uses percent - based width instead of fixed pixels, so element resize smoothly	Works well on different screen sizes, easy to implement	less control over layout on large screen.
Adaptive	Uses fixed layouts that changes 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.

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

→ lifecycle of service workers
A service workers is a script that runs in the background and helps a web app work offline, load faster and send push notifications.

1. Registration phase
- The browser registers the service worker using Javascript

Code Example.
```
if ('service worker' in navigator) {
    navigator.serviceWorker.register('/sw.js')
    .then(() => console.log('service worker registred'))
    .catch (error => console.log ('Registration failed
    );
```
- 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 the new one.
- Unused cache files from the previous versions 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);
        }
      }));
    })
  );
});
```

Final step : fetch & sync.

once activated. the service worker intercepts network request, serves cached files, and syncs data when the ~~internat~~ internet is available

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

Q.4 Explain the use of indexedDB in the service worker for data storage

Ans Use of indexedDB in service worker for Data storage.

IndexedDB is a browser database that stores large amount of structured data like JSON objects.

It helps PWAs work offline by saving and retrieving data efficiently.

Why use IndexedDB in service worker?

1. offline support : stores data when offline and syncs it
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

How service workers use indexedDB?

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', {key
                path: 'id'});
    store.add({id:1, name: 'John Doe', age: 25});
};
```

Fetching Data in service worker.

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

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