05/05

SDS Page No. Date

GANESH SANJAY PANDHRE/DISB/41

*MPL Assignment 02 *

Q. 1 Define Progressive web App (PWA) and explain its significance in modern web development. Discuss the key characteris 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 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 browser.
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
i) Installation	Direct from browser	Download from App Store.
ii) Internet Required	Works offline with caching	Usually requires internet.
iii) Performance	Fast with service workers.	Faster but needs installation.
iv) Updates	Automatic, no app store approval	Manual updates needed.
v) Development cost	Lower	Higher

PWAs 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.

=> 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.

SDS | Page No.
Date / /

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 Works	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.
Adaptive	Uses fixed layouts that changes at specific breakpoints.	Optimized for known screen sizes.	More effort required to design for each screen size.

Q. 3 Describe the lifecycle of services workers, including 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 browers registers the service worker using Javascript.

Code Example :-

```
if ('serviceWorker' in navigator) {
    navigator.service.worker.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 and stores them in cache.
- If successful, it moves to the activation phase.

SDS | Page No.
Date / /

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.

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 requests, serves cached files, and syncs data when the internet is available.

The 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.

=> - Use of IndexedDB in service workers 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 syns it later.
2. Efficient storage - saves structured data like user settings, cart items, or forms inputs.
3. Faster Access - Retrieves data quickly without needing a network request.
4. Persistent Data - Data remains saved even after the brower is closed.

SDS Page No.
Date / /

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);
};
```