Vaibhav Boudh
D15B
Roll no: 06 MPL Assignment - 2

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.

ns A progressive web app (PWA) is a web application that combines the best features of both web and mobile apps to deliver a seamless, reliable, and fast user experience. PWAs work offline, load quickly, and provide an app like experience on web browsers.

Significance in Modern Web development:

- Platform Independance - Runs on any device with a web browser
- Improved Performance - Faster load times due to caching and Service workers.
- Offline Functionality - Works without an internet connection
- No App Store Dependencies - Users can install PWAs directly from the browser.
- Engaging User Experience - provides push notification and background syncing.

Key characteristics of PWAs vs Traditional Mobile Apps:

Feature	PWAs	Traditional Mobile Apps
Installation	Installed from a browser	Downloaded from App store
Platform	Works across platforms with one codebase	Requires separate development for iOS and Android
Dependency	Uses Service Workers for offline access	Usually requires native implementation
Offline Support	Updated automatically via the web	Requires app store updates
Updates	Faster due to caching and lightweight assets	Can be heavier, but optimized for specific platforms
Performance		

Teacher's Sign.:

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.

Ans Responsive web Design is an approach that ensures web pages adapt to different screen sizes and orientations using flexible grids, media queries, and scalable images.

Importance in PWAS:

- Ensures a consistent user experience across different devices.
- Eliminates the need for multiple codebases for different devices.
- Enhances usability by making content readable on various screen.

Comparison of Responsive, Fluid and Adaptive web design.

Feature	Responsive	Fluid	Adaptive
Definition	Uses CSS media queries to adjust layout dynamically	Uses percentage based units for elements to scale naturally	Uses predefined layouts for different screen sizes
Flexibility	Highly Flexible	Completely flexible	Fixed at specific breakpoints
Performance	Efficient but requires more CSS adjustments	Smooth scaling	May cause layout shifts
Best use case	Websites and PWAS for all screen	Apps requiring seamless scaling	websites with predefined layouts

3. Describe the lifecycle of Service Workers, including registration, installation, and activation phases.

Ans A Service Worker is a background script that runs independently from the main browser thread, enabling features like offline caching and push notification.

Teacher's Sign.: _____

Lifecycle Phases:

1. Registration:

· The service worker is registered in JavaScript using navigator.serviceWorker.register(). e.g.

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

2. Installation:

· Occurs when the service worker is first downloaded.

· Typically used for caching assets. e.g.

```
self.addEventListener('install', event =>{
  event.waitUntil(
    caches.open('v1').then(cache =>{
return cache.addAll(['/index.html', '/style.css']);
  })
  );
});
```

3. Activation:

· Runs after installation and ensures old caches are cleared if necessary. e.g.

```
self.addEventListener('activate', event =>{
  event.waitUntil(
    caches.keys().then(keys =>{
      return Promise.all(
keys.filter(key => key !== 'v1').map(key => caches.delete(key))
      );
    })
  );
});
```

Teacher's Sign.: _____

4. Fetching and Updates:
- The service worker intercepts network requests and serves cached content. e.g.

```
self.addEventListener('fetch', event => {
  event.respondWith(
    caches.match(event.request).then(response => {
      return response || fetch(event.request);
    })
  );
});
```

4. Explain the use of Indexed DB in the Service Worker for data storage.

Ans Indexed DB is a low-level NoSQL database in the browser that allows web apps to store and retrieve large amounts of structured data efficiently.

Used of Index DB in Service Workers:

1. Offline storage - Saves user data when offline and syncs it when back online
2. Persistent Data - Unlike local Storage, Indexed DB is asynchronous and handles large amounts of data.
3. Background Sync - Service Workers can use Indexed DB to store data and sync it later.

E.g.-

```
Const dbRequest = indexedDB.open('myDatabase', 1);
dbRequest.onupgradeneeded = event => { const db = event.target;
db.createObjectStore('messages', { keyPath: 'id' }); };
dbRequest.onsuccess = event => { const db = event.target.result;
const transaction = db.transaction('messages', 'readwrite');
const store = transaction.objectStore('messages');
store.put({ id: 1, Text: 'Hello from IndexedDB' }); };
```

Teacher's Sign.: