DEEPA BEHRANI
D15B - 05

05/05

Date
Page

MPL ASSIGNMENT 2.

① Define progressive webapp (PWA) & explain its significance in modern development. Discuss the key characteristics PWA's from traditional mobile apps.

Soln:

A progressive webapp (PWA) is a type of web application that works like a mobile app & ~~runs~~ runs in a browser

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.

	PWA	Traditional app
Installation:	Direct from Browser	Download from app Store.
Internet required	work offline with caching	Usually requires internet.
performance	Fast with service workers	Fast but need installation
updates	Automatic	manual
Development cost	Lower	Higher

② Define responsive web design & explain its importance in the content of progressive web apps. Compare & construct responsive fluid & adaptive web design approaches.

Soln: Definition of responsive web design:

Responsive web design (RWD) is a technique that makes web pages adjust automatically to different screen size

Teacher's Sign.: ____________

and devices. It ensures a good users experience on mob
tablets & desktops without needing separate version o
website.
Importance:
1) Better user experience
2) faster Load time
3) SEO Benefits
4) cost effective.

Approach	How it works	Pros ~~cons~~	cons
Responsive	Use flexible grids & css media queries	works on all devices	Can be compl
fluid	Use % based widths instead of fixed pixels	works well on different screen size	less contro over

(Q3) Describes the lifecycle of service workers, including registration, installation & activation phases
Soln: Lifecycle of service workers:
A service workers is a script that runs in the background & helps a web app work offline, lo faster & send push notifications. Its lifecycle has 3 phases:
1] Registration phase: Browser register the servic worker using Javascript. Eg:
```
if ('serviceworker' in navigator) {
    navigator.service worker register ("/sw.js
    .then () => console.log ("service register
    .catch (error) => console.log ("failed! error))
}
```

Date
Page

2) Installation phase : ~~Iterative~~ The service workers downloads necessary files (HTML, CSS, JS) & stores them in cache. if successful, it moves to the activation phase

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

3] Activation phase.
The old services worker is replaced with new one unreal cache files from the previous version are deleted
final step: fetch and sync
once activated, the service worker intercept network request serves cached files & syncs data.

Explain the uses of ~~internal~~ indexedDB in the service worker for data storage
→ Use of IndexDB in service worker for Data storage : Index-edDB is a browser database that stores large amount of structured data like JSON. objects. It helps PWAs work offline by serving & retrieving data efficiency.
Why use IndexedDB in service workers ?
1] Offline support : store data when offline and sync its later.
2] Efficient storage : saves structure dates like user setting cart items are form inputs
3] faster Access : Retrieves data quickly without needing a network request.
4] Persistent Data : Data remains saved after the browser is closed.

Teacher's Sign.:

How service workers use IndexedDB?
# opening the database :

```
let db:
let request = indexedDB.open ("Mydatabase", 1
request.onsuccess = function (event) {
        db = event.target.open () ;
    };
```

Creating a store and adding data.

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

fetch Data in service worker

```
let transaction = db. transaction ('users' 'readon
let store = transaction. object store ('user');
let getUser = store.get (1) ;
get User. onsuccess = function () {
        console.log (get user result) ;
    };
```