Aditya Gupta
D15B-17 MPL

NAME: STD.: DIV.: DATE: 05/03 PAGE:

Q1] Define progressive web app (PWA) and explain the significance in mordern web development. Discuss the key characteristics that differentiate PWA's from traditional mobile apps

A1] A progressive web app (PWA) is a type of web application that works like a mobile app but runs in browser. It can be installed on a device, works offline and provides a fast and smooth user experience.

Significance of PWA in mordern web development:

- cross platform compatibility
- offline support
- fast performance
- No app store required
- lower development cost
- Key difference

	PWA	Traditional
1]	Installation Direct from browser	Download from App store
2]	Works offline with caching	It usually require Internet
3]	Fast with service workers	Faster but need installation

Q2] Define responsiveness web design and explain its importance in context of progressive web Apps. Compare and contrast responsiveness fluids and adaptive design approaches

NAME: STD.: DIV.: DATE: PAGE:

⇒ Responsive web design (RWD) is technique that makes web page adjust automatically to different screen size and device. It ensure a good user experience on mobile tablets and desktops without needing seperate version of website

Importance

- Better user experience PWA i work smoothly on any device
- Faster load time. optimized design improves speed
- SEO benefits - Google ranks responsive sites tougher

Comparision

Approaches	How it works	Pros	Cons
Responsiveness	user flexible grids and css media queris to adjust layout	works on all device	Can be complex to design
Fluid	user percent based widths instead of fixed pixels. i.e elements resize smoothly	works well on different screen size easy to imple-ment	Less control over layout on large screens

Key difference

- Responsive adapts dynamically to all screen
- Fluid resize smoothly, but may be fully a prioritized
- Adaptive load different layout based on device types

NAME:_____ STD.:_____ DIV.:_____ DATE: PAGE:

Q3) Describe the lifecycle of service workers including to registration installation and activation phase

→ Lifecycle of Service Worker

A service workers is a script that runs in the background and helps a web apps work offline load faster and send push notification. Its lifecycle has there main phases

1) Registration Phase

The browser register the source worker using Javascript

code Ex

```
if ('service worker in navigator') {
    navigator.service worker.register ('/sw.js')
    .then (() => console.log ('service worker register'))
    .catch (error => console.log (Registration failed, error)
```

This tells the browser to install and activate the service worker

2. Installation phase

The source worker downloads necessary files (HTML, CSS, JS) and stores them in cache

If successful it moves to the activation phase

code Ex

```
self add EventListner ('install event => &
    event wait until (
    'cache.open (app.cache') then (cache => {
    returns cache add All (['[1' (index.html',
        'style.css'])}
})
{ })
```

NAME:_____ STD.:_____ DIV.:_____ DATE : PAGE :

3] Activation phase
The old service worker is replaced with new one.
Unused cache files from the previous version are deleted
One activated service worker intercepts network requests
Serves cached files and sync data when the data
Internet is available

4] Explain the use of Indexed DB in the system service worker of data

A Use of Indexed DB in service worker for data storage Indexed DB is a browser database that store large amount of structure data like JSON objects. It helps PWA to work offline

1] Offline support - stores data when offline and sync is later
2] Efficient storage - saves structured data like user setting cost item or form inputs
3] Faster Access - Retrieves data quickly without needing a network request
4] Persistent Data - Data remains saved even after the browser

Opening the database

```
let db;
let request = indexedDB.open('MyDatabase');
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 : randomly')
let store = transaction.objectStore ('users')
getUser.onsuccess = function () { console.log (getUserdet') }
```