MPL Experiment - 7

* **Aim:** To implement the "Add to Home Screen" feature in a Progressive Web App (PWA) using a service worker, manifest file, and caching techniques, allowing users to install the web application on their device like a native app.

* **Theory:** A PWA is a modern web application that behaves like a native mobile or desktop app. It allows users to install the app on their home screen, work offline, and load quickly, providing a seamless user experience.

Key features of a PWA:

- **Responsive & Adaptive** - Works on all screen sizes.
- **Offline Support** - Uses a service worker for caching and offline access.
- **App-like Experience** - Runs in standalone mode, without a browser UI.
- **Push Notifications** - Supports background updates.
- **Secure & Fast** - Uses HTTPS and caching techniques.

How "Add to Home Screen" Works in a PWA?

1. **Manifest File (manifest.json)**
   - Defines app metadata (name, icons, theme, start URL, etc.).
   - Enables installation on the home screen.

2. **Service Worker (serviceworker.js)**

- Handles caching and offline access.
- Ensures app loads quickly even with poor network conditions.

3. Registration in index.html
- Registers the service worker to enable background operations.

When a PWA is correctly configured, the browser detects it and prompts the user with an "Install this site as an app" option, allowing them to add to their home screen.

* Conclusion: In this experiment, we successfully implemented the "Add to Home Screen" feature in a Progressive Web App (PWA). This was achieved by:
1. Manifest file (manifest.json) - Defined app metadata, making the app installable.
2. Service Worker (serviceworker.js) - Handled caching and enabled offline access.
3. Service Worker Registration (index.html) - Allowed the browser to detect and prompt app installation.

After implementation, we verified the installation process and confirmed that the PWA icon appeared on the home screen. This experiment demonstrated how PWAs provide a native-like experience, improve user engagement, and enhance performance by allowing fast, offline-ready access.

Code Implementation of Experiment 7:

Index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>SneakCart - Sneaker Store</title>
  <link rel="manifest" href="manifest.json" />
  <link rel="stylesheet" href="style.css" />
  <meta name="theme-color" content="#000000" />
</head>
<body>
  <h1>Welcome to SneakCart</h1>
  <p>Shop the coolest sneakers in town!</p>
</body>
</html>
```

Manifest.json:

```
{
    "name": "SneakCart - Sneaker Shop",
    "short_name": "SneakCart",
    "start_url": "index.html",
    "display": "standalone",
    "background_color": "#ffffff",
    "theme_color": "#000000",
    "description": "A simple PWA sneaker e-commerce app",
    "icons": [
      {
        "src": "icons/icon-192.png",
        "sizes": "192x192",
        "type": "image/png"
      },
      {
        "src": "icons/icon-512.png",
        "sizes": "512x512",
        "type": "image/png"
      }
    ]
  }
```

Style.css:

```
body {
```

```
  font-family: sans-serif;
  text-align: center;
  padding: 2rem;
  background-color: #f4f4f4;
  color: #333;
}
```

Output: