Experiment No: - 8

Aim:- To code and register a service worker, and complete the install and activation process for a new service worker for the E-commerce PWA.

Theory:-

A service worker is a script that your browser runs in the background, separate from a web page, and can be used to handle tasks such as push notifications, background synchronization, and caching resources to improve performance and offline functionality.

Here are some key uses and benefits of service workers:

1. Offline Functionality: Service workers enable your web application to work offline by caching essential resources such as HTML, CSS, JavaScript, and images. When the user is offline, the service worker can intercept network requests and serve the cached resources, allowing the application to continue functioning.

2. Performance Optimization: By caching resources, service workers can significantly improve the performance of your web application by reducing the need for repeated network requests. Cached resources can be served quickly from the local cache, reducing latency and improving load times.

3. Push Notifications: Service workers enable web applications to receive push notifications even when the application is not open in the browser. This feature allows you to engage users with timely updates, messages, or alerts, similar to native mobile applications.

4. Background Synchronization: Service workers can perform background synchronization tasks, allowing your web application to sync data with a server even when the application is not actively being used. This feature is useful for applications that require real-time data updates or offline data synchronization.

5. Network Request Interception and Routing: Service workers can intercept and handle network requests made by your web application, giving you control over how requests are handled. This capability allows you to implement custom caching strategies, route requests to different servers, or serve responses from the cache.

6. Improved Reliability: Service workers can help improve the reliability and robustness of your web application by providing a layer of redundancy for critical functionality. Even if the user loses network connectivity or the server is temporarily unavailable, the service worker can continue to serve cached content or perform background tasks.

Overall, service workers are a powerful feature of modern web browsers that enable developers to create faster, more reliable, and more engaging web applications with advanced functionality previously only available in native applications. However, it's essential to use service workers responsibly and consider the potential impact on user privacy, security, and resource usage.

ServiceWorker.js:

```
self.addEventListener('install', function(event) {

console.log('Service Worker: Installed');

});

self.addEventListener('activate', function(event) {

console.log('Service Worker: Activated');

});

self.addEventListener('fetch', function(event) {

console.log('Service Worker: Fetching');

event.respondWith(
```

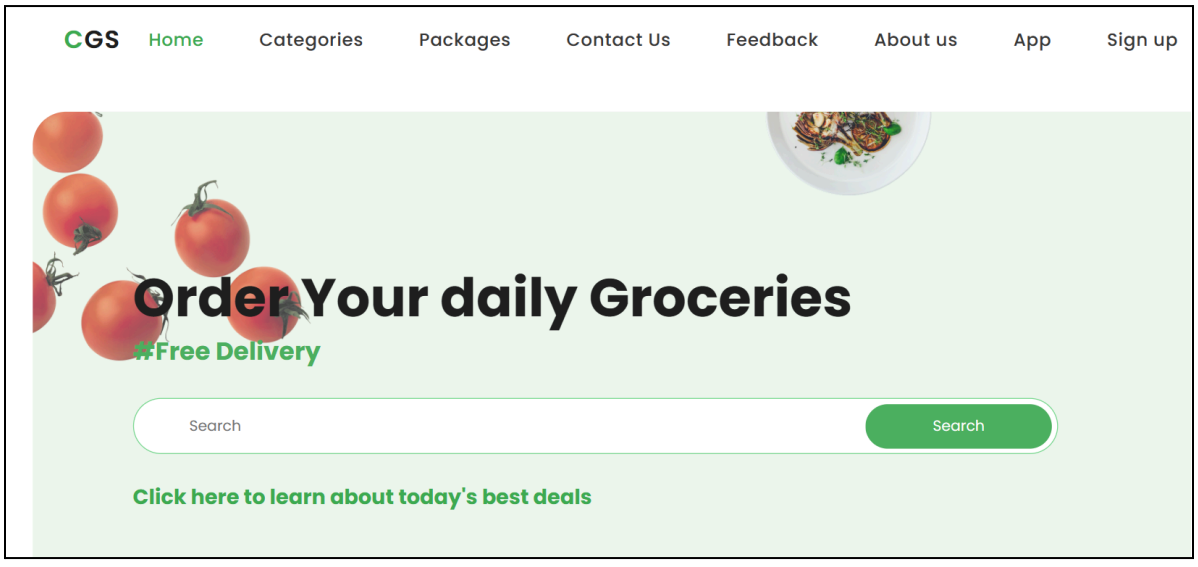
```
fetch(event.request)
.then(function(response) {
if (!response || response.status !== 200 || response.type !== 'basic') {
return response;
}
var responseToCache = response.clone();
caches.open('v1').then(function(cache) {
cache.put(event.request, responseToCache);
});
return response;
})
.catch(function(err) {
console.error('Service Worker: Error fetching and caching new data:', err);
})
);
});
```

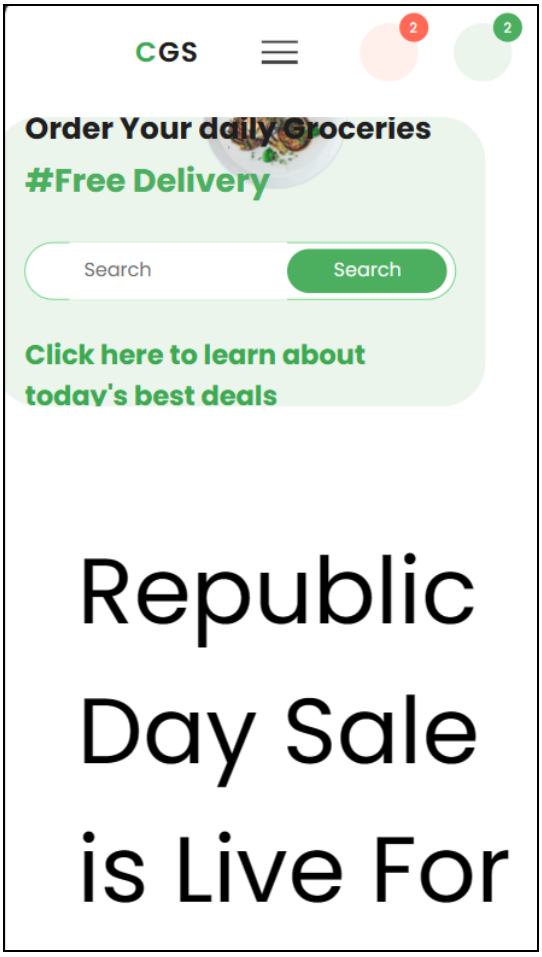
index.html:

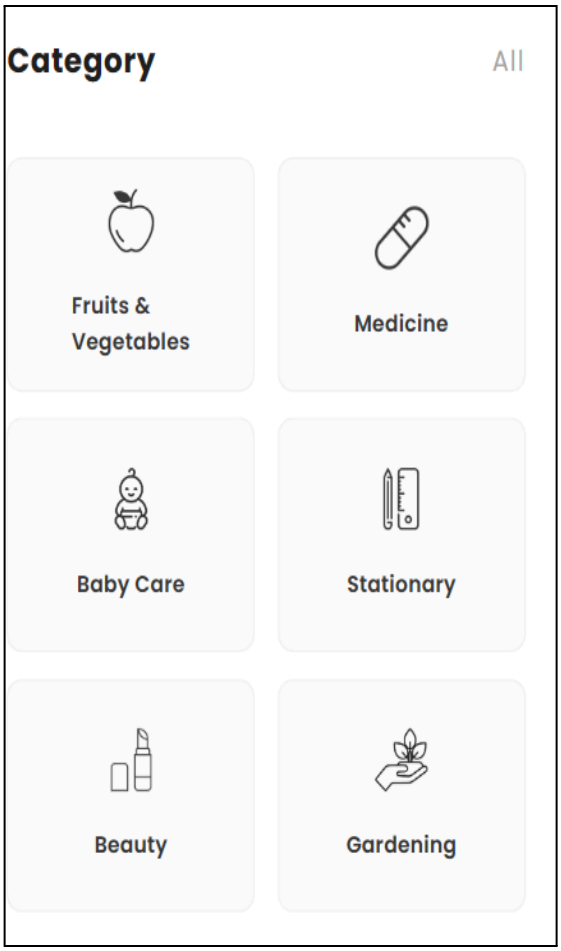
```
<script>
if ('serviceWorker' in navigator) {
window.addEventListener('load', () => {
navigator.serviceWorker.register('ServiceWorker.js')
```

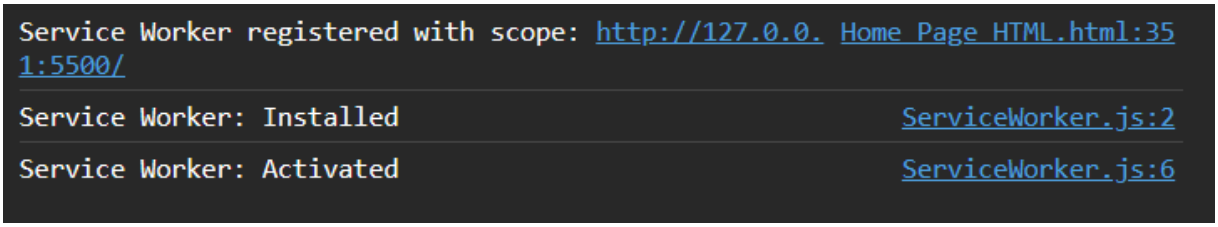
```
.then(registration => {
console.log('Service Worker registered with scope:', registration.scope);
})
.catch(error => {
console.error('Service Worker registration failed:', error);
});
});
}
</script>
```

Output:









Conclusion:

Hence, successfully implemented a service worker for the E-commerce Progressive Web App (PWA) involves writing the service worker code to handle caching and network requests, registering the service worker in the main JavaScript file of the app, and completing the install and activation process.