MPL Practical 09

Aim: To implement Service worker events like fetch, sync and push for E-commerce PWA.

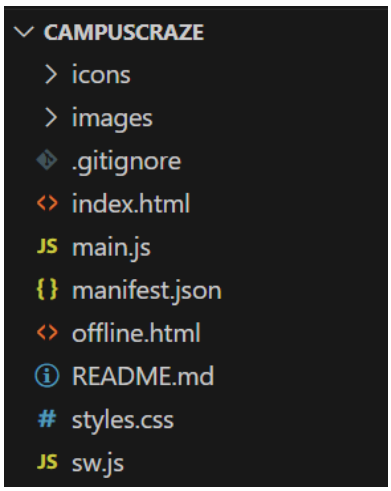
Theory

A **service worker** can handle more than just caching — it can also manage **fetch**, **sync**, and **push** events to improve a web app's performance and user experience. These events allow the app to work offline, sync data when internet is back, and send notifications to users.
In this experiment, we enhanced our CampusCraze PWA by implementing:

- A **fetch event** to serve cached files when offline, and show a custom offline.html page if the network fails.
- A **sync event** to register background sync using SyncManager, which lets the app send data to the server when the internet connection is restored.
- A **push event** that listens for push messages and shows custom notifications even when the app is not open.

These features make CampusCraze more reliable, responsive, and engaging for users, especially in low or no network conditions.

Folder Structure



---

index.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>EcoGadgetHub – Sustainable Tech Gadgets</title>
  <link rel="manifest" href="manifest.json">
```

```
   </style>
  </head>
  <body>
   <script>
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="description" content="CampusCraze - Budget-friendly stationery, hostel decor
and fashion for college students">
  <meta name="theme-color" content="#ff6f61">

  <title>CampusCraze – Cool Stuff for College Life</title>

  <link rel="stylesheet" href="styles.css">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link
href="https://fonts.googleapis.com/css2?family=Montserrat:wght@400;500;700&display=swap"
rel="stylesheet">

  <!-- PWA requirements -->
  <link rel="manifest" href="manifest.json">
  <link rel="apple-touch-icon" href="icons/icon-192x192.png">

  <!-- For better SEO and sharing -->
  <meta property="og:title" content="CampusCraze – Cool Stuff for College Life">
  <meta property="og:description" content="From hostel décor to budget fashion — shop
everything you need to live your best college life!">
  <meta property="og:image" content="images/og-image.jpg">
</head>
<body>

  <script>
     if ("serviceWorker" in navigator) {
       window.addEventListener("load", () => {
         navigator.serviceWorker
           .register("/sw.js")
           .then((registration) => {
```

```
          console.log(
            "Service Worker registered! Scope:",
            registration.scope
          );


          // Request Push Notification Permission
          if ("PushManager" in window) {
            Notification.requestPermission().then((permission) => {
              if (permission === "granted") {
                console.log("Push notifications granted.");
              } else {
                console.log("Push notifications denied.");
              }
            });
          }


          // Register Background Sync
          if ("SyncManager" in window) {
            navigator.serviceWorker.ready.then((swReg) => {
              swReg.sync
                .register("sync-data")
                .then(() => {
                  console.log("Sync registered");
                })
                .catch((err) => {
                  console.log("Sync registration failed:", err);
                });
            });
          }
        })
        .catch((error) => {
          console.log("Service Worker registration failed:", error);
        });
    });
  }
 </script>
</body>
</html>
```

offline.html

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8" />
 <meta name="viewport" content="width=device-width, initial-scale=1.0" />
 <title>CampusCraze - Offline</title>
 <style>
 </style>
</head>
<body>
   <div class="message">
    <p>Looks like your internet connection is pulling an all-nighter! 💤</p>
    <p>Check your Wi-Fi or data connection and try again.</p>
   </div>
</body>
</html>
```

serviceworker.js

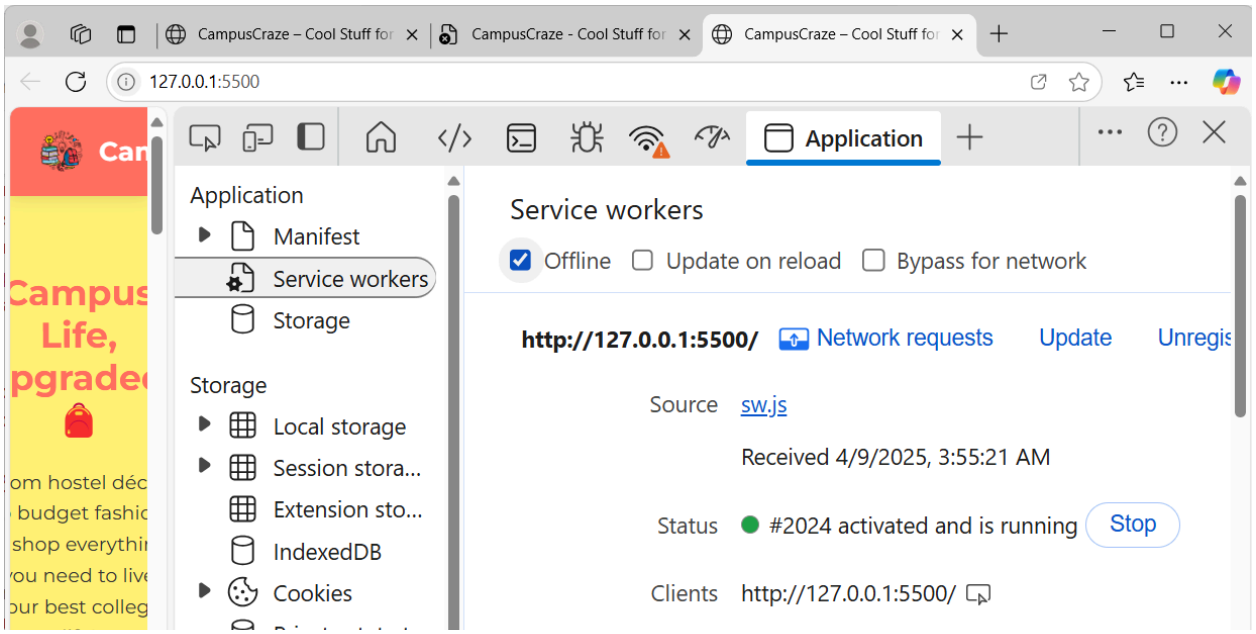
```
// CampusCraze Service Worker for PWA functionality
const CACHE_NAME = 'campuscraze-v1';
const ASSETS = [
 '/',
 '/index.html',
 '/styles.css',
 '/main.js',
 '/manifest.json',
 '/images/logo.svg',
 '/images/college-vibes.webp',
 '/images/stationery.webp',
 '/images/decor.webp',
 '/images/fashion.webp',
 '/images/tech.webp',
 '/images/study-bundle.webp',
 '/images/decor-bundle.webp',
 '/images/app-mockup.webp',
 '/images/google-play.svg',
 '/images/app-store.svg',
 '/images/avatar-simran.webp',
```

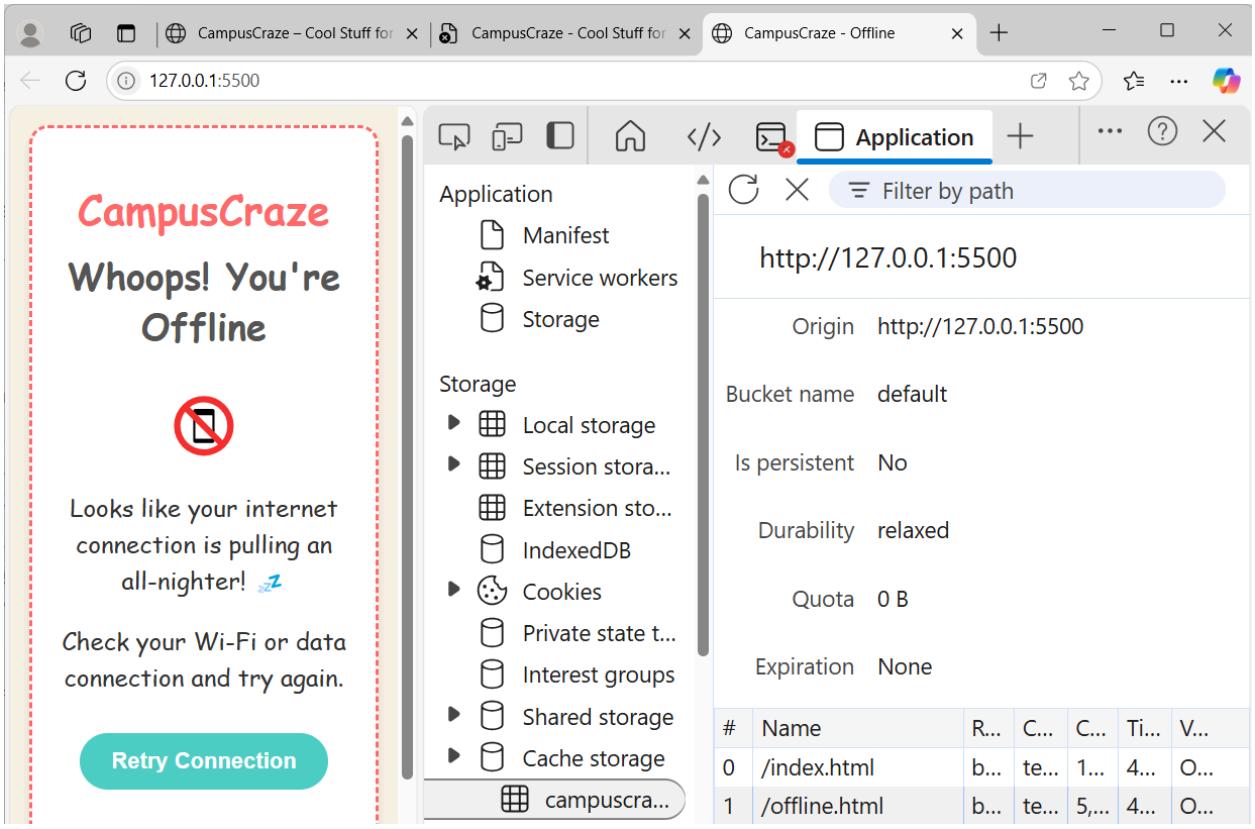
```
  '/images/avatar-rishi.webp',
  '/images/avatar-priya.webp',
  '/images/instagram.svg',
  '/images/youtube.svg',
  '/images/twitter.svg',
  '/icons/icon-192x192.png',
  '/icons/icon-512x512.png',
  'offline.html',
  'https://fonts.googleapis.com/css2?family=Montserrat:wght@400;500;700&display=swap'
];

// Install Event
self.addEventListener("install", (event) => {
  console.log("[ServiceWorker] Install");
  event.waitUntil(
    caches.open(CACHE_NAME).then((cache) => {
      console.log("[ServiceWorker] Caching files");
      return cache.addAll(ASSETS);
    })
  );
});

// Activate Event
self.addEventListener("activate", (event) => {
  console.log("[ServiceWorker] Activate");
  event.waitUntil(
    caches.keys().then((keyList) =>
      Promise.all(
        keyList.map((key) => {
          if (key !== CACHE_NAME) {
            console.log("[ServiceWorker] Removing old cache", key);
            return caches.delete(key);
          }
        })
      )
    )
  );
  return self.clients.claim();
});
```

```
// Enhanced Fetch Event
self.addEventListener("fetch", (event) => {
  console.log("[ServiceWorker] Fetch", event.request.url);
  const requestURL = new URL(event.request.url);

  // If request is same-origin, use Cache First
  if (requestURL.origin === location.origin) {
    event.respondWith(
      caches.match(event.request).then((cachedResponse) => {
        return ( cachedResponse ||
          fetch(event.request).catch(() => caches.match("offline.html"))
        );
      })
    );
  } else {
    // Else, use Network First
    event.respondWith(
      fetch(event.request)
        .then((response) => {
          return response;
        })
        .catch(() =>
          caches.match(event.request).then((res) => {
            return res || caches.match("offline.html");
          })
        )
    );
  }
});

// Sync Event (simulation)
self.addEventListener("sync", (event) => {
  if (event.tag === "sync-data") {
    event.waitUntil(
      (async () => {
        console.log("Sync event triggered: 'sync-data'");
        // Here you can sync data with server when online
      })()
    );
  }
```

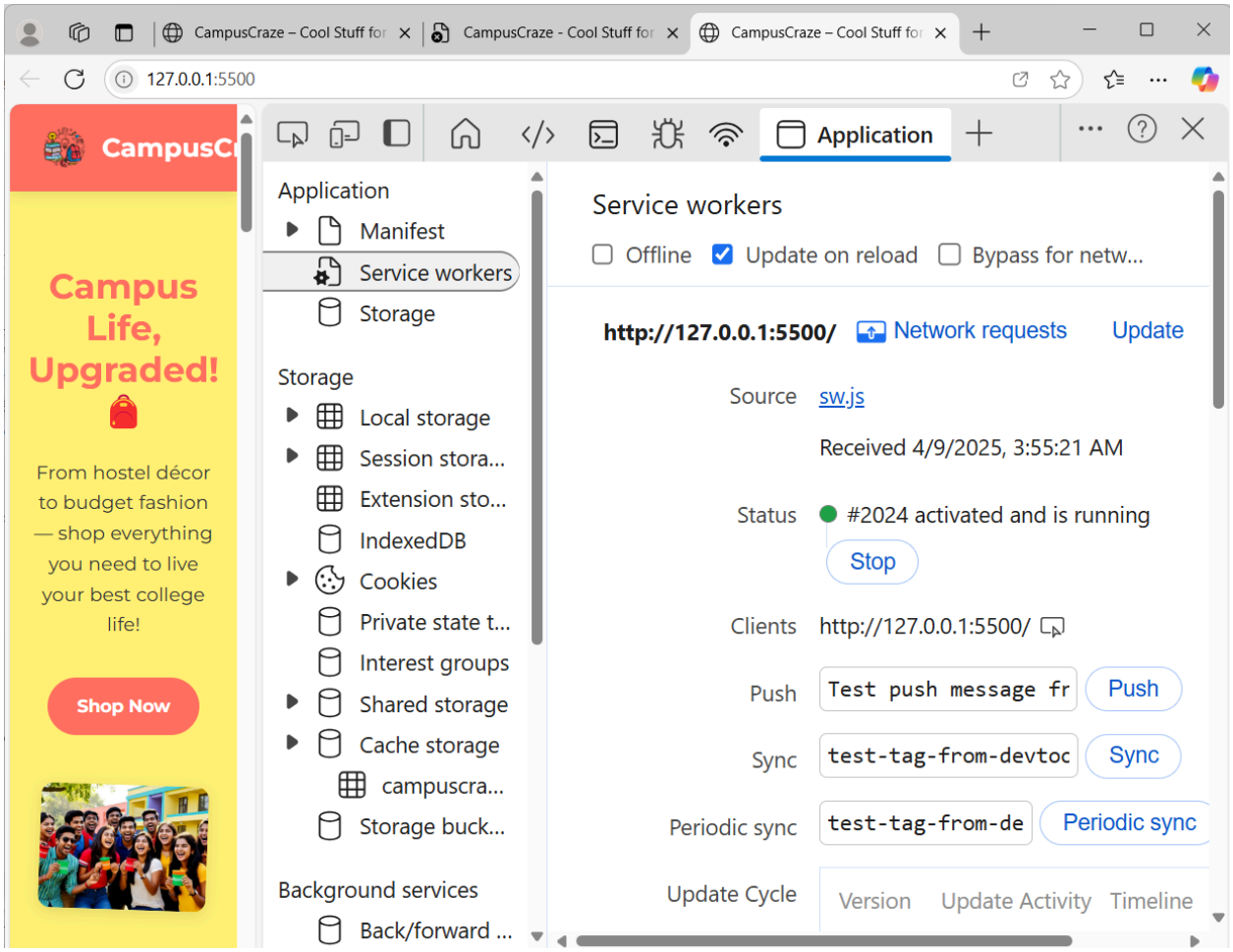
```
});

// Push Event
self.addEventListener("push", function (event) {
  if (event && event.data) {
    let data = {};
    try {
      data = event.data.json();
    } catch (e) {
      data = {
        method: "pushMessage",
        message: event.data.text(),
      };
    }

    if (data.method === "pushMessage") {
      console.log("Push notification sent");
      event.waitUntil(
        self.registration.showNotification("CampusCraze - Cool Stuff for College Life", {
          body: data.message,
        })
      );
    }
  }
});
```

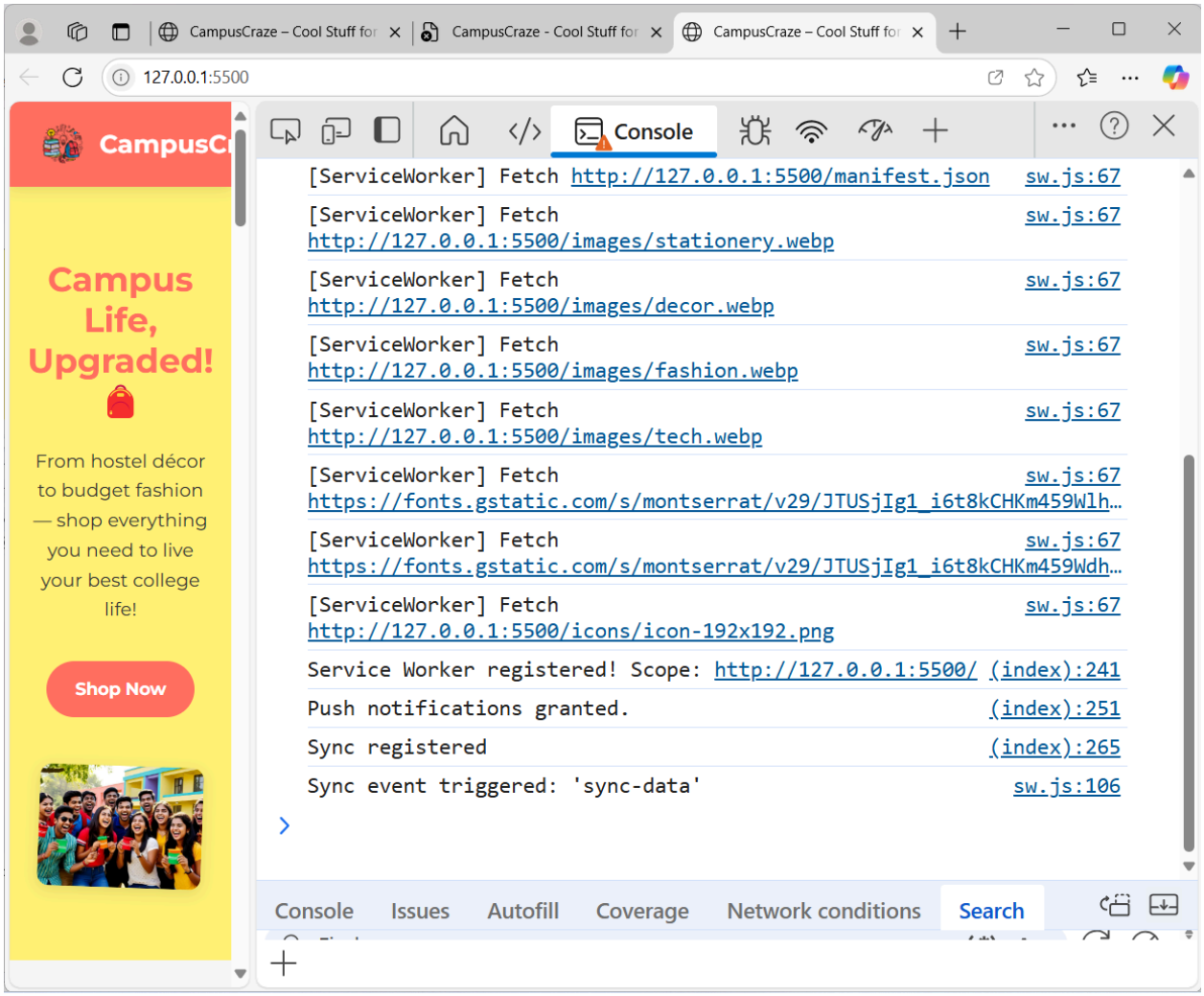
1. Test: Fetch Event (Offline Support)



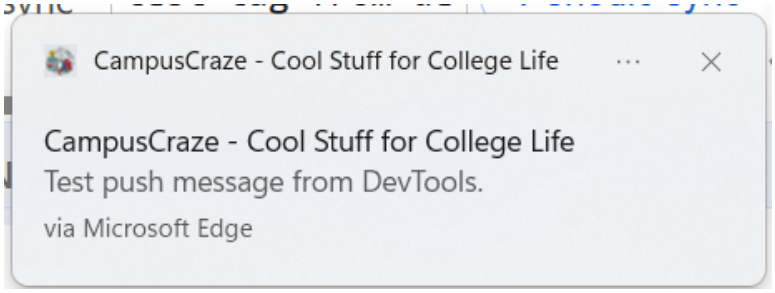


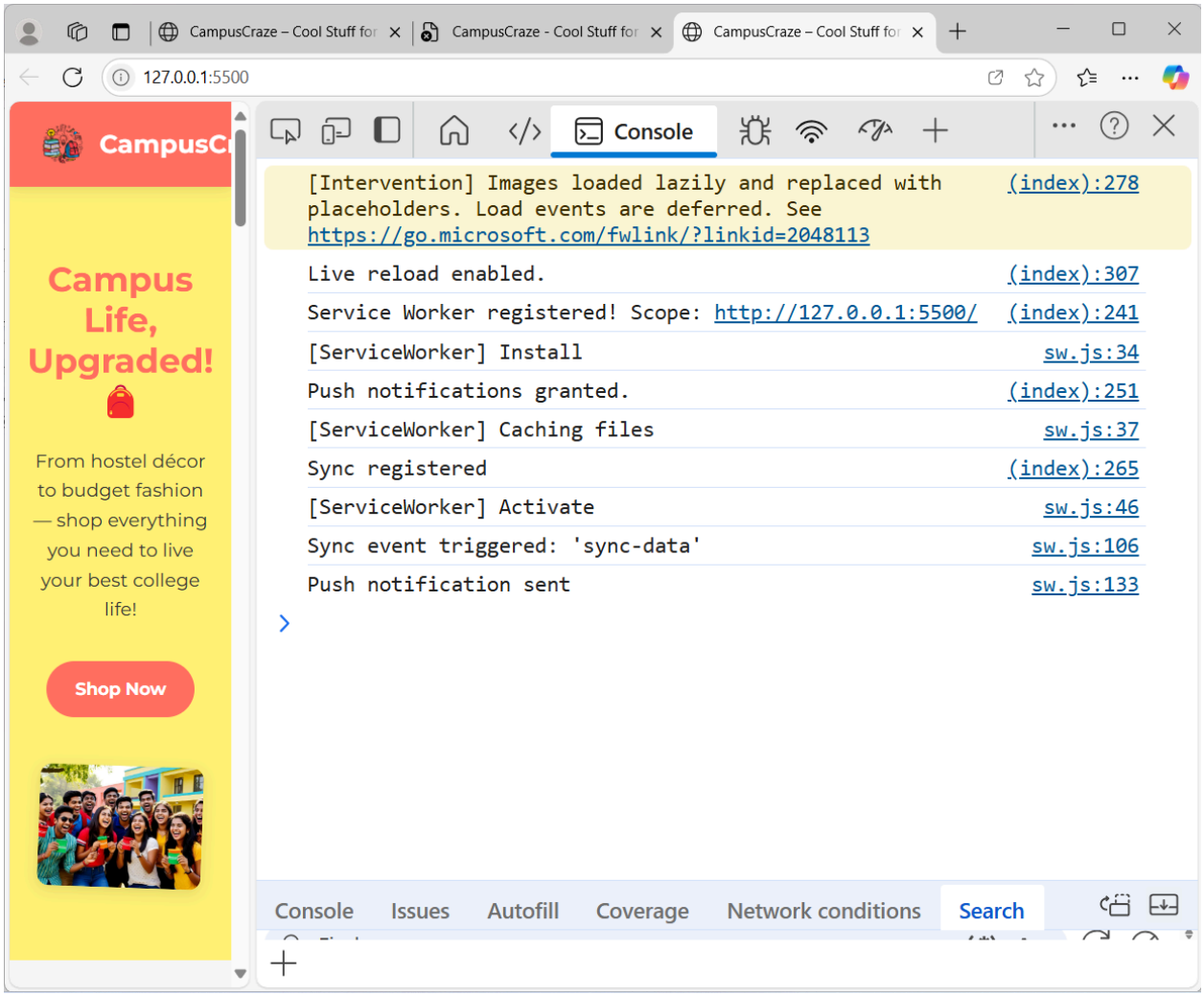
2. Test: Background Sync Event





3. Test: Push Notification Event





Conclusion

In this experiment, we implemented fetch, sync, and push events in the CampusCraze service worker to enable offline access, background syncing, and push notifications. Initially, the sync event wasn't triggering because we forgot to check for navigator.serviceWorker.ready, which we fixed by properly chaining the sync registration inside it.