Experiment No. 7

Aim: To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.

Theory:

- **Regular Web App**

  A regular web app is a website that is designed to be accessible on all mobile devices such that the content gets fit as per the device screen. It is designed using a web technology stack (HTML, CSS, JavaScript, Ruby, etc.) and operates via a browser. They offer various native-device features and functionalities. However, it entirely depends on the browser the user is using. In other words, it might be possible that you can access a native-device feature on Chrome but not on Safari or Mozilla Firefox because the browsers are incompatible with that feature.

- **Progressive Web App**

  Progressive Web App (PWA) is a regular web app, but some extras enable it to deliver an excellent user experience. It is a perfect blend of desktop and mobile application experience to give both platforms to the end-users.

- **Difference between PWAs vs. Regular Web Apps:**

  1. Native Experience: Though a PWA runs on web technologies (HTML, CSS, JavaScript) like a Regular web app, it gives user experience like a native mobile application. It can use most native device features, including push notifications, without relying on the browser or any other entity. It offers a seamless and integrated user experience that it is quite tough for one to differentiate between a PWA and a Native application by considering its look and feel.

  2. Ease of Access: Unlike other mobile apps, PWAs do not demand longer download time and make memory space available for installing the applications. The PWAs can be shared and installed by a link, which cuts down the number of steps to install and use. These applications can easily keep an app icon on the user's home screen, making the app easily accessible to the users and helps the brands remain in the users' minds, and improving the chances of interaction.

  3. Faster Services: PWAs can cache the data and serve the user with text stylesheets, images, and other web content even before the page loads completely. This lowers the waiting time for the end-users and helps the brands improve the user engagement and retention rate, which eventually adds value to their business.

  4. Engaging Approach: As already shared, the PWAs can employ push notifications and other native device features more efficiently. Their interaction does not depend on the browser user uses. This eventually improves the chances of notifying the user regarding your services, offers, and other options related to your brand and keeping them hooked to your brand. In simpler words, PWAs let you maintain the user engagement and retention rate.

5. **Updated Real:** Time Data Access: Another plus point of PWAs is that these apps get updated on their own. They do not demand the end-users to go to the App Store or other such platforms to download the update and wait until installed.

6. **Discoverable**: PWAs reside in web browsers. This implies higher chances of optimizing them as per the Search Engine Optimization (SEO) criteria and improving the Google rankings like that in websites and other web apps.

7. **Lower Development Cost:** Progressive web apps can be installed on the user device like a native device, but it does not demand submission on an App Store. This makes it far more cost-effective than native mobile applications while offering the same set of functionalities.

- **The main features are:**

1. Progressive — They work for every user, regardless of the browser chosen because they are built at the base with progressive improvement principles.
2. Responsive — They adapt to the various screen sizes: desktop, mobile, tablet, or dimensions that can later become available.
3. Updated — Information is always up-to-date thanks to the data update process offered by service workers.
4. Secure — Exposed over HTTPS protocol to prevent the connection from displaying information or altering the contents.
5. Searchable — They are identified as “applications” and are indexed by search engines.
6. Reactivable — Make it easy to reactivate the application thanks to capabilities such as web notifications.
7. Installable — They allow the user to “save” the apps that he considers most useful with the corresponding icon on the screen of his mobile terminal (home screen) without having to face all the steps and problems related to the use of the app store.
8. Linkable — Easily shared via URL without complex installations.
9. Offline — Once more it is about putting the user before everything, avoiding the usual error message in case of weak or no connection. The PWA are based on two particularities: first of all the ‘skeleton’ of the app, which recalls the page structure, even if its contents do not respond and its elements include the header, the page layout, as well as an illustration that signals that the page is loading.

Code:

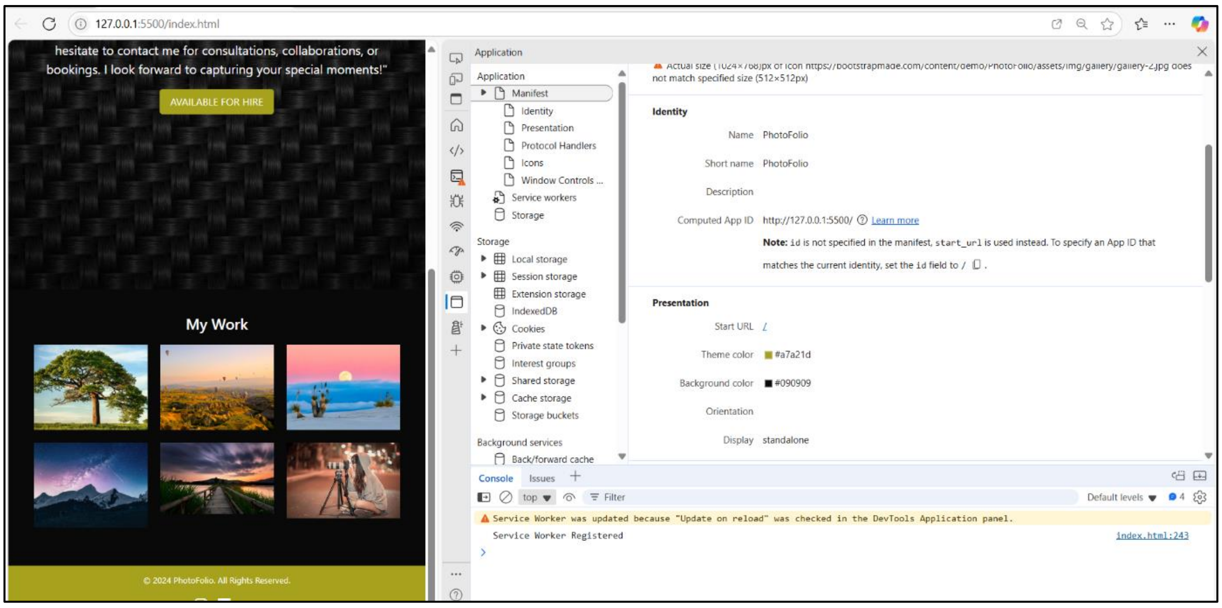
1. **Manifest.json**

```
{
    "name": "PhotoFolio",
    "short_name": "PhotoFolio",
    "start_url": "/",
    "display": "standalone",
    "background_color": "#090909",
    "theme_color": "#a7a21d",
    "icons": [
      {
        "src":
"https://bootstrapmade.com/content/demo/PhotoFolio/assets/img/gallery/gallery-
1.jpg",
        "sizes": "192x192",
        "type": "image/png"
      },
      {
        "src":
"https://bootstrapmade.com/content/demo/PhotoFolio/assets/img/gallery/gallery-
2.jpg",
        "sizes": "512x512",
        "type": "image/png"
      }
    ]
  }
```

2. Service-worker.js

```
self.addEventListener("install", (event) => {
    event.waitUntil(
        caches.open("static").then((cache) => {
            return cache.addAll(["/", "/index.html", "/styles.css"]);
        })
    );
});

self.addEventListener("fetch", (event) => {
    event.respondWith(
        caches.match(event.request).then((response) => {
            return response || fetch(event.request);
        })
    );
})
```

Output:



Actual size (1024×768)px of icon https://bootstrapmade.com/content/demo/PhotoFolio/assets/img/gallery/gallery-2.jpg does not match specified size (512×512px)

Identity

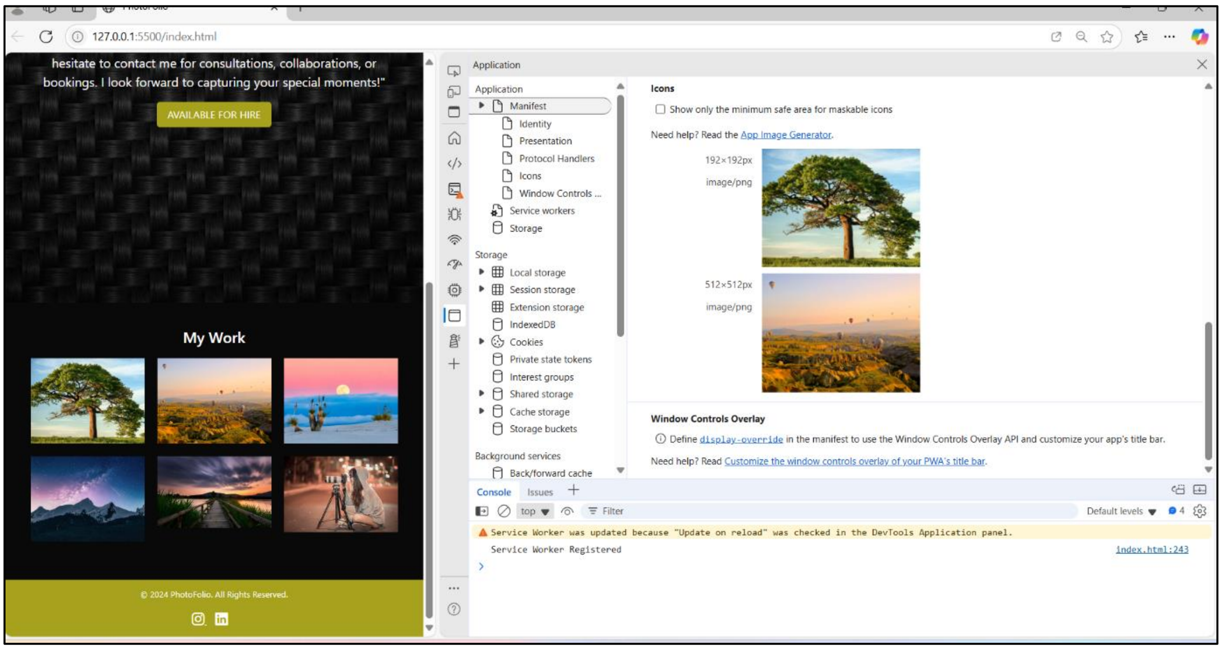
Name	PhotoFolio
Short name	PhotoFolio
Description	
Computed App ID	http://127.0.0.1:5500/ Learn more

Note: `id` is not specified in the manifest, `start_url` is used instead. To specify an App ID that matches the current identity, set the `id` field to `/`.

Presentation

Start URL	/
Theme color	#a7a21d
Background color	#090909
Orientation	
Display	standalone

```
Service Worker was updated because "Update on reload" was checked in the DevTools Application panel.
Service Worker Registered                                                    index.html:243
```



hesitate to contact me for consultations, collaborations, or bookings. I look forward to capturing your special moments!"

© 2024 PhotoFolio. All Rights Reserved.

Icons

Show only the minimum safe area for maskable icons

Need help? Read the App Image Generator.

192×192px image/png

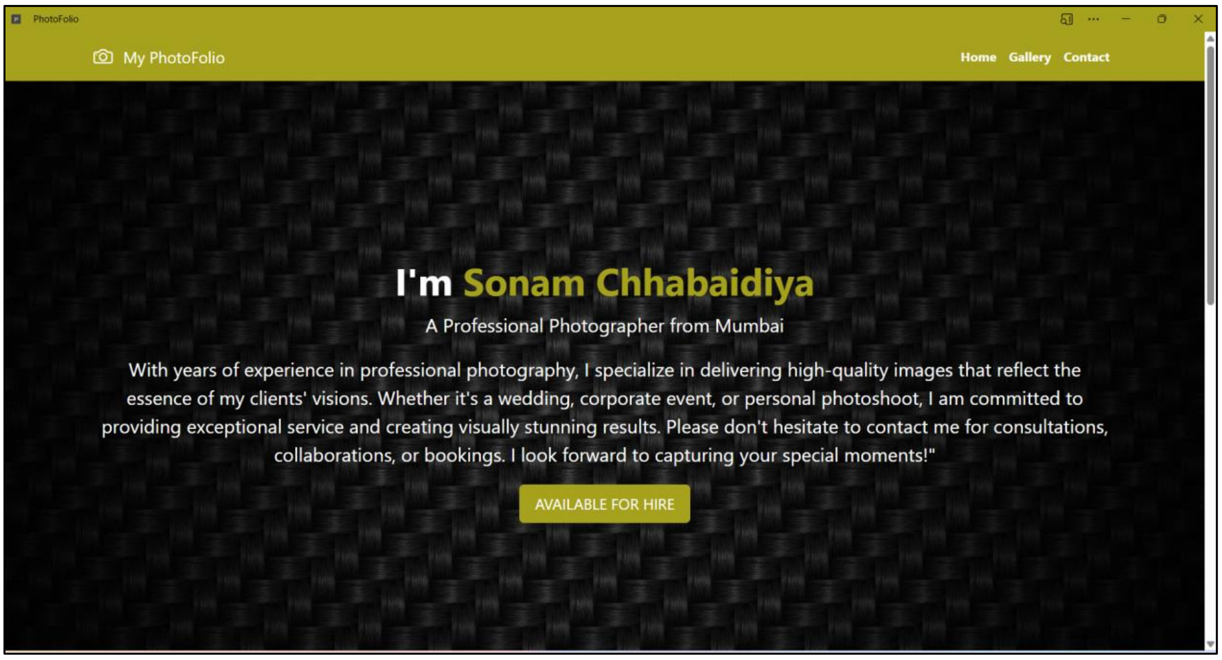
512×512px image/png

Window Controls Overlay

Define display-override in the manifest to use the Window Controls Overlay API and customize your app's title bar.

Need help? Read Customize the window controls overlay of your PWA's title bar.

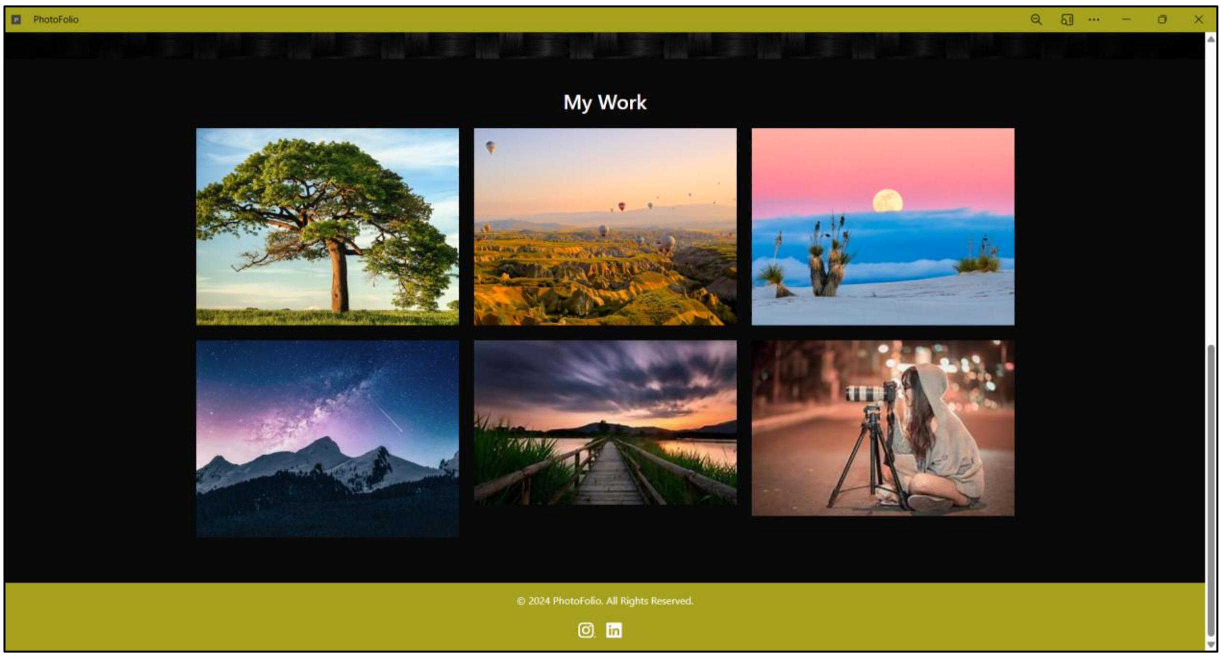
```
⚠ Service Worker was updated because "Update on reload" was checked in the DevTools Application panel.
Service Worker Registered                                                    index.html:243
```

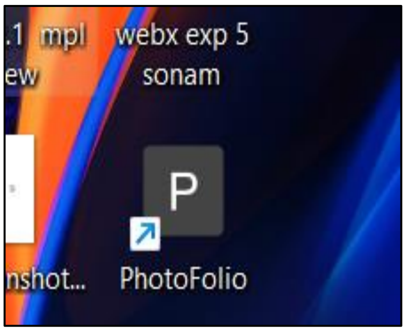


I'm Sonam Chhabaidiya

A Professional Photographer from Mumbai

With years of experience in professional photography, I specialize in delivering high-quality images that reflect the essence of my clients' visions. Whether it's a wedding, corporate event, or personal photoshoot, I am committed to providing exceptional service and creating visually stunning results. Please don't hesitate to contact me for consultations, collaborations, or bookings. I look forward to capturing your special moments!"


PhotoFolio
My Work
© 2024 PhotoFolio. All Rights Reserved.


.1 mpl
ew
webx exp 5
sonam
P
nshot...
PhotoFolio

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:

Service Worker

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

- A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
- Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
- The service worker becomes idle when not in use and restarts when it's next needed. You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.

What can we do with Service Workers?

- You can dominate **Network Traffic**

You can manage all network traffic of the page and do any manipulations. For example, when the page requests a CSS file, you can send plain text as a response or when the page requests an HTML file, you can send a png file as a response. You can also send a true response too.

- You can **Cache**

You can cache any request/response pair with Service Worker and Cache API and you can access these offline content anytime.

- You can manage **Push Notifications**

You can manage push notifications with Service Worker and show any information message to the user.

- You can **Continue**

Although Internet connection is broken, you can start any process with Background Sync of Service Worker.

What can’t we do with Service Workers?

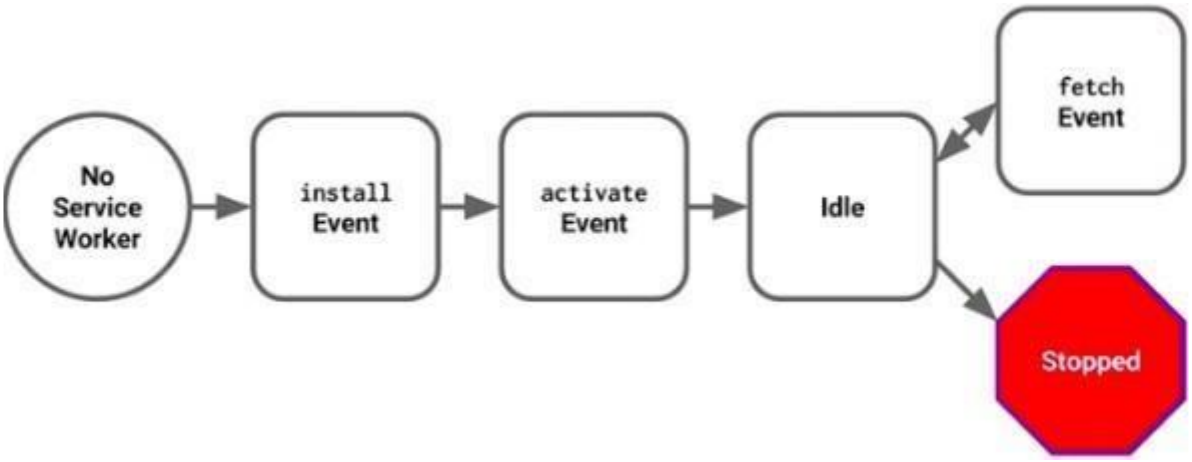
- You can’t access the **Window**

You can’t access the window, therefore, You can’t manipulate DOM elements. But, you can
communicate to the window through post Message and manage processes that you want.

- You can’t work it on **80 Port**

Service Worker just can work on HTTPS protocol. But you can work on localhost during development.

Service Worker Cycle



A service worker goes through three steps in its life cycle:

- Registration
- Installation
- Activation

Registration

To install a service worker, you need to register it in your main JavaScript code. Registration tells the browser where your service worker is located, and to start installing it in the background. Let's look at an example:

main.js

```
if ('serviceWorker' in navigator) { navigator.serviceWorker.register('/service-worker.js')
.then(function(registration) {
console.log('Registration successful, scope is:', registration.scope);
})
.catch(function(error) {
console.log('Service worker registration failed, error:', error);
});
}
```

This code starts by checking for browser support by examining **navigator.serviceWorker**. The service worker is then registered with navigator.serviceWorker.register, which returns a promise that resolves when the service worker has been successfully registered. The scope of the service worker is then logged with registration.scope. If the service worker is already installed, navigator.serviceWorker.register returns the registration object of the currently active service worker.

The scope of the service worker determines which files the service worker controls, in other words, from which path the service worker will intercept requests. The default scope is the location of the service worker file, and extends to all directories below. So if service-worker.js is located in the root directory, the service worker will control requests from all files at this domain.

You can also set an arbitrary scope by passing in an additional parameter when registering. For example: main.js

```
navigator.serviceWorker.register('/service-worker.js', { scope: '/app/'

});
```

In this case we are setting the scope of the service worker to /app/, which means the service worker will control requests from pages like /app/, /app/lower/ and /app/lower/lower, but not from pages like /app or /, which are higher.

If you want the service worker to control higher pages e.g. /app (without the trailing slash) you can indeed change the scope option, but you'll also need to set the Service-Worker-Allowed HTTP Header in your server config for the request serving the service worker script.

main.js

```
navigator.serviceWorker.register('/app/service-worker.js', { scope: '/app'

});
```

Installation

Once the browser registers a service worker, installation can be attempted. This occurs if the service worker is considered to be new by the browser, either because the site currently doesn't have a registered service worker, or because there is a byte difference between the new service worker and the previously installed one.

A service worker installation triggers an install event in the installing service worker. We can include an install event listener in the service worker to perform some task when the service worker installs. For instance, during the install, service workers can precache parts of a web app so that it loads instantly the next time a user opens it (see caching the application shell). So, after that first load, you're going to benefit from instant repeat loads and your time to interactivity is going to be even better in those cases. An example of an installation event listener looks like this:

service-worker.js

```
// Listen for install event, set callback

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

```
// Perform some task
});
```

Activation

Once a service worker has successfully installed, it transitions into the activation stage. If there are any open pages controlled by the previous service worker, the new service worker enters a waiting state. The new service worker only activates when there are no longer any pages loaded that are still using the old service worker. This ensures that only one version of the service worker is running at any given time.

When the new service worker activates, an activate event is triggered in the activating service worker. This event listener is a good place to clean up outdated caches (see the Offline Cookbook for an example).

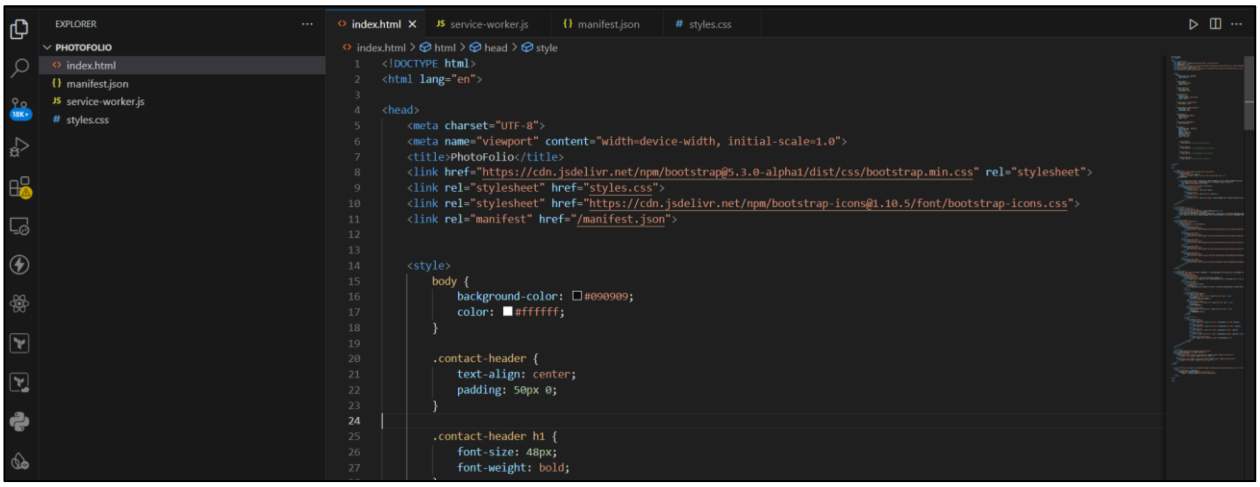
service-worker.js

```
self.addEventListener('activate', function(event) {// Perform some task
});
```

Once activated, the service worker controls all pages that load within its scope, and starts listening for events from those pages. However, pages in your app that were loaded before the service worker activation will not be under service worker control. The new service worker will only take over when you close and reopen your app, or if the service worker calls **clients.claim()**. Until then, requests from this page will not be intercepted by the new service worker. This is intentional as a way to ensure consistency in your site.

CODE:

Index.html:



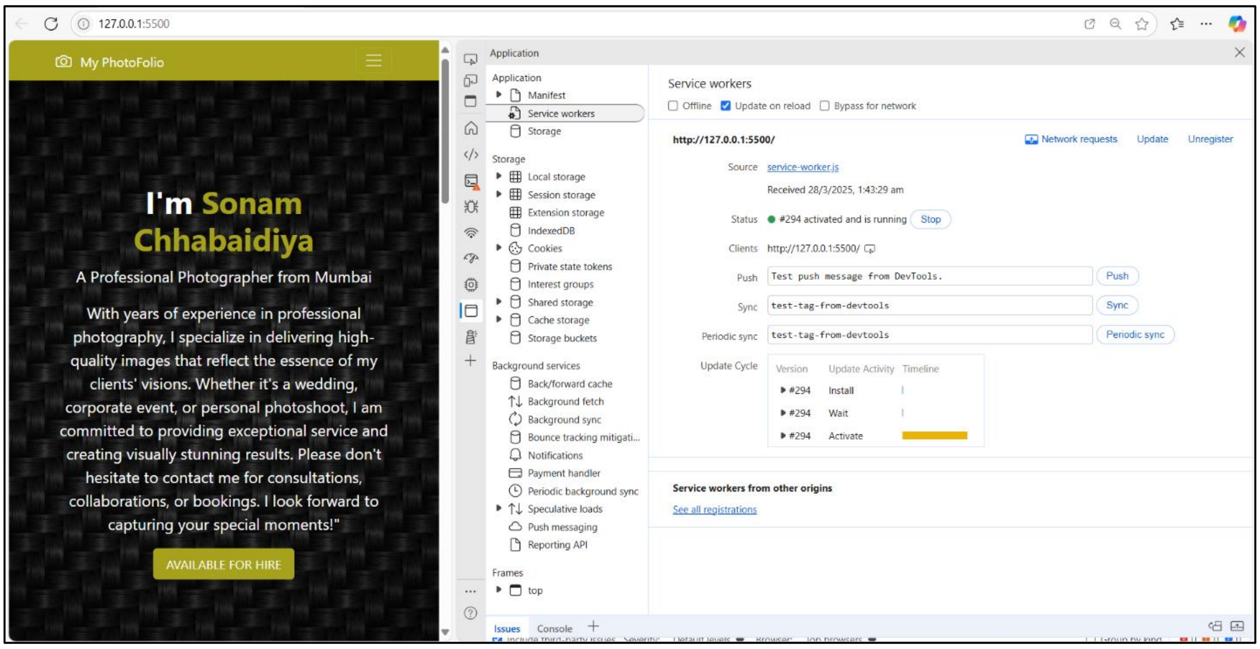
```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>PhotoFolio</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="styles.css">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
    <link rel="manifest" href="/manifest.json">


    <style>
        body {
            background-color: #090909;
            color: #ffffff;
        }

        .contact-header {
            text-align: center;
            padding: 50px 0;
        }

        .contact-header h1 {
            font-size: 48px;
            font-weight: bold;
```

Service_worker.js

```
const CACHE_NAME = "photoFolio-cache-v4";
const urlsToCache = [
  "/",
  "/index.html",
  "/styles.css",
  "/manifest.json",
  "/service-worker.js",
  "https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css",
  "https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css"
];

// Install event - Cache static assets
self.addEventListener("install", (event) => {
  event.waitUntil(
    caches.open(CACHE_NAME).then((cache) => {
      console.log("Opened cache and adding assets...");

 return cache.addAll(urlsToCache);
    })
  );
});
```

```
// Fetch event - Cache external images dynamically
self.addEventListener("fetch", (event) => {
  event.respondWith(
    caches.match(event.request).then((response) => {
      return response || fetch(event.request).then((fetchResponse) => {
        let requestUrl = event.request.url;

        // Cache only images (JPG, PNG, etc.)
        if (requestUrl.match(/\.(jpg|jpeg|png|gif|webp)$/)) {
          return caches.open(CACHE_NAME).then((cache) => {
            cache.put(event.request, fetchResponse.clone());
            return fetchResponse;
          });
        }

        return fetchResponse;
      });
    })
  );
});

// Activate event - Cleanup old caches
self.addEventListener("activate", (event) => {
  event.waitUntil(
    caches.keys().then((cacheNames) => {
      return Promise.all(
        cacheNames.map((cache) => {
          if (cache !== CACHE_NAME) {
            console.log("Deleting old cache:", cache);
            return caches.delete(cache);
          }
        })
      );
    })
  );

});
```

OUTPUT:



I'm Sonam Chhabaidiya

A Professional Photographer from Mumbai

With years of experience in professional photography, I specialize in delivering high-quality images that reflect the essence of my clients' visions. Whether it's a wedding, corporate event, or personal photoshoot, I am committed to providing exceptional service and creating visually stunning results. Please don't hesitate to contact me for consultations, collaborations, or bookings. I look forward to capturing your special moments!"

AVAILABLE FOR HIRE

Service workers

☐ Offline ☑ Update on reload ☐ Bypass for network

http://127.0.0.1:5500/ Network requests Update Unregister

Source service-worker.js
Received 28/3/2025, 1:43:29 am

Status ● #294 activated and is running Stop

Clients http://127.0.0.1:5500/

Push `Test push message from DevTools.` Push

Sync `test-tag-from-devtools` Sync

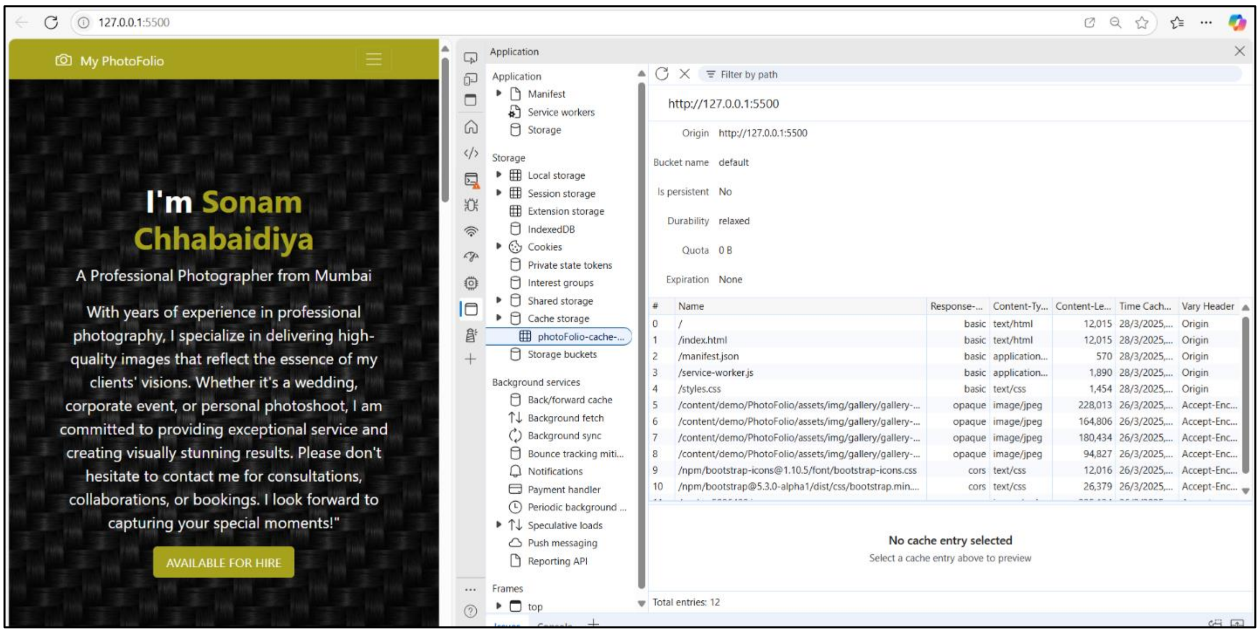
Periodic sync `test-tag-from-devtools` Periodic sync

Update Cycle

Version	Update Activity	Timeline
#294	Install	
#294	Wait	
#294	Activate	

Service workers from other origins

See all registrations



I'm Sonam Chhabaidiya

A Professional Photographer from Mumbai

With years of experience in professional photography, I specialize in delivering high-quality images that reflect the essence of my clients' visions. Whether it's a wedding, corporate event, or personal photoshoot, I am committed to providing exceptional service and creating visually stunning results. Please don't hesitate to contact me for consultations, collaborations, or bookings. I look forward to capturing your special moments!"

AVAILABLE FOR HIRE

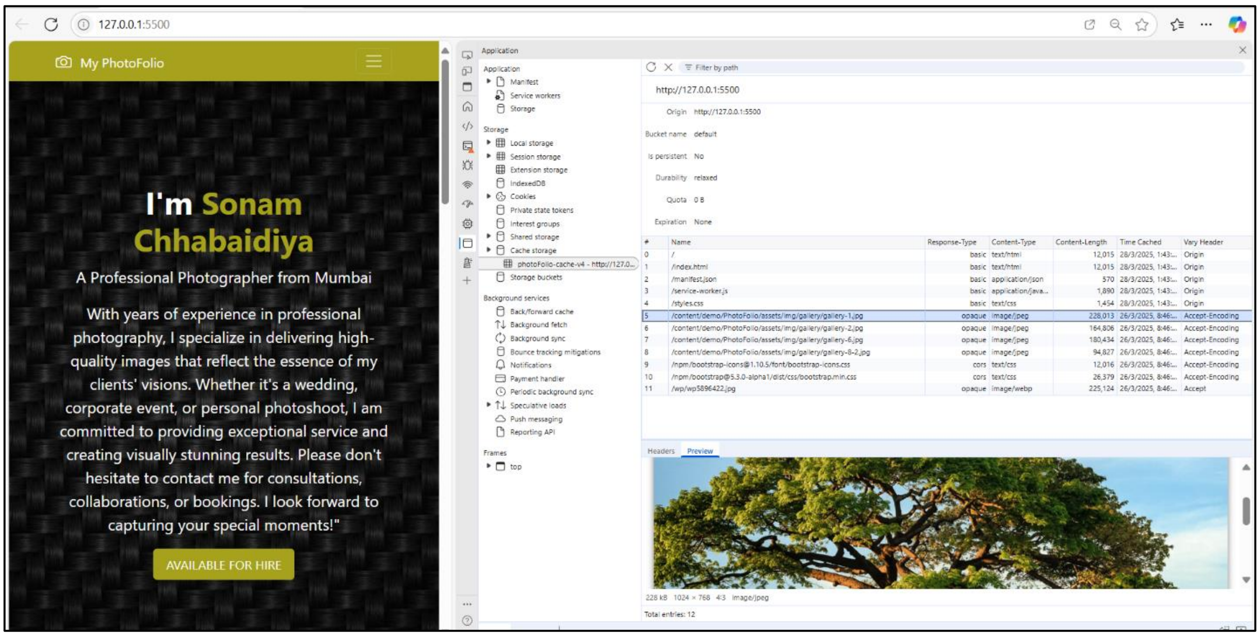
http://127.0.0.1:5500

Origin http://127.0.0.1:5500
Bucket name default
Is persistent No
Durability relaxed
Quota 0 B
Expiration None

#	Name	Response-...	Content-Ty...	Content-Le...	Time Cach...	Vary Header
0	/	basic	text/html	12,015	28/3/2025,...	Origin
1	/index.html	basic	text/html	12,015	28/3/2025,...	Origin
2	/manifest.json	basic	application...	570	28/3/2025,...	Origin
3	/service-worker.js	basic	application...	1,890	28/3/2025,...	Origin
4	/styles.css	basic	text/css	1,454	28/3/2025,...	Origin
5	/content/demo/PhotoFolio/assets/img/gallery/gallery-...	opaque	image/jpeg	228,013	26/3/2025,...	Accept-Enc...
6	/content/demo/PhotoFolio/assets/img/gallery/gallery-...	opaque	image/jpeg	164,806	26/3/2025,...	Accept-Enc...
7	/content/demo/PhotoFolio/assets/img/gallery/gallery-...	opaque	image/jpeg	180,434	26/3/2025,...	Accept-Enc...
8	/content/demo/PhotoFolio/assets/img/gallery/gallery-...	opaque	image/jpeg	94,827	26/3/2025,...	Accept-Enc...
9	/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css	cors	text/css	12,016	26/3/2025,...	Accept-Enc...
10	/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min....	cors	text/css	26,379	26/3/2025,...	Accept-Enc...

No cache entry selected
Select a cache entry above to preview

Total entries: 12



I'm Sonam Chhabaidiya

A Professional Photographer from Mumbai

With years of experience in professional photography, I specialize in delivering high-quality images that reflect the essence of my clients' visions. Whether it's a wedding, corporate event, or personal photoshoot, I am committed to providing exceptional service and creating visually stunning results. Please don't hesitate to contact me for consultations, collaborations, or bookings. I look forward to capturing your special moments!"

http://127.0.0.1:5500

Origin	http://127.0.0.1:5500
Bucket name	default
Is persistent	No
Durability	relaxed
Quota	0 B
Expiration	None

#	Name	Response-Type	Content-Type	Content-Length	Time Cached	Vary Header
0	/	basic	text/html	12,015	28/3/2025, 1:43:...	Origin
1	/index.html	basic	text/html	12,015	28/3/2025, 1:43:...	Origin
2	/manifest.json	basic	application/json	570	28/3/2025, 1:43:...	Origin
3	/service-worker.js	basic	application/java...	1,890	28/3/2025, 1:43:...	Origin
4	/styles.css	basic	text/css	1,454	28/3/2025, 1:43:...	Origin
5	/content/demo/PhotoFolio/assets/img/gallery/gallery-1.jpg	opaque	image/jpeg	228,013	26/3/2025, 8:46:...	Accept-Encoding
6	/content/demo/PhotoFolio/assets/img/gallery/gallery-2.jpg	opaque	image/jpeg	164,806	26/3/2025, 8:46:...	Accept-Encoding
7	/content/demo/PhotoFolio/assets/img/gallery/gallery-6.jpg	opaque	image/jpeg	180,434	26/3/2025, 8:46:...	Accept-Encoding
8	/content/demo/PhotoFolio/assets/img/gallery/gallery-8-2.jpg	opaque	image/jpeg	94,827	26/3/2025, 8:46:...	Accept-Encoding
9	/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css	cors	text/css	12,016	26/3/2025, 8:46:...	Accept-Encoding
10	/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css	cors	text/css	26,379	26/3/2025, 8:46:...	Accept-Encoding
11	/wp/wp5896422.jpg	opaque	image/webp	225,124	26/3/2025, 8:46:...	Accept

228 kB 1024 × 768 4:3 image/jpeg

Total entries: 12

EXPERIMENT NO. 9

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

Theory:

Service Worker

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

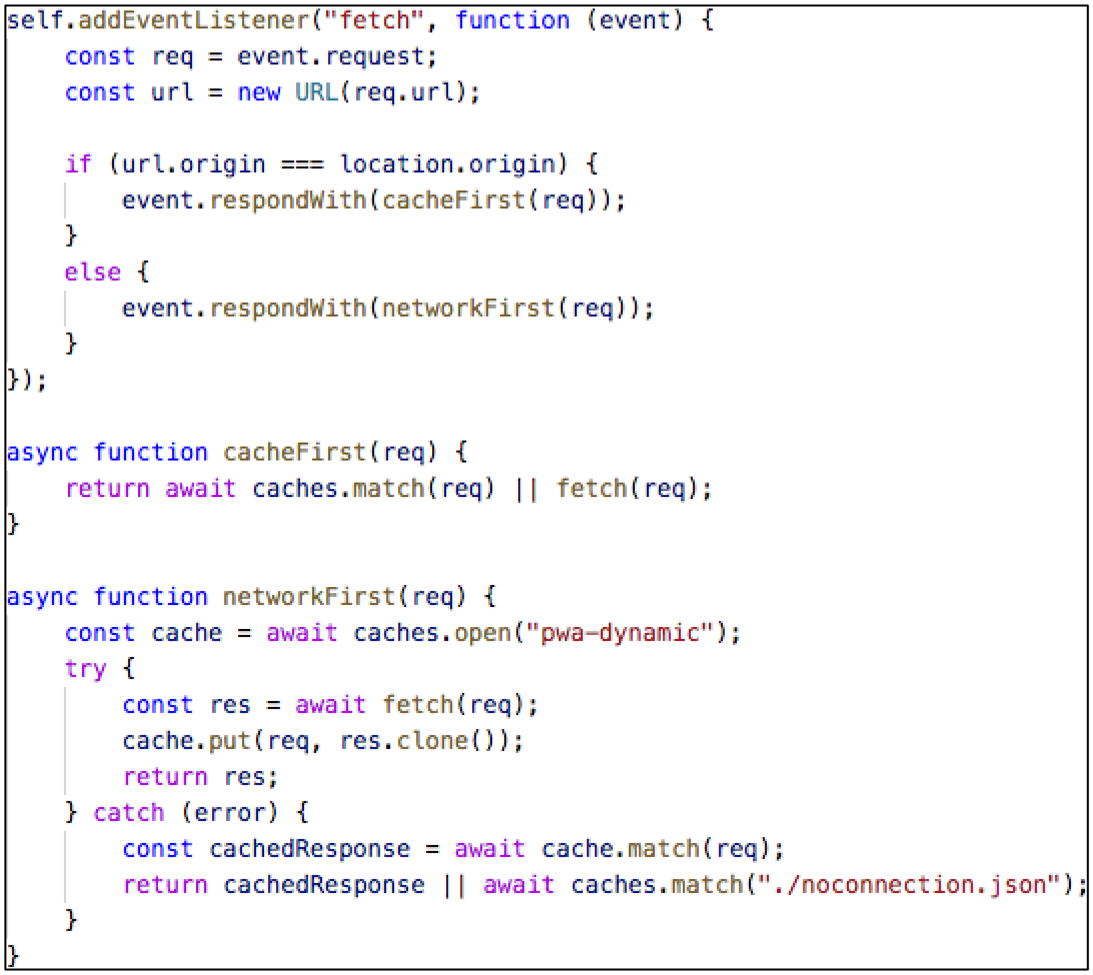
- A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
- Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
- The service worker becomes idle when not in use and restarts when it's next needed. You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.
- Service workers make extensive use of promises, so if you're new to promises, then you should stop reading this and check out Promises, an introduction.

Fetch Event

You can track and manage page network traffic with this event. You can check existing cache, manage “cache first” and “network first” requests and return a response that you want.

Of course, you can use many different methods but you can find in the following example a “cache first” and “network first” approach. In this example, if the request’s and current location’s origin are the same (Static content is requested.), this is called “cacheFirst” but if you request a targeted external URL, this is called “networkFirst”.

- **CacheFirst** - In this function, if the received request has cached before, the cached response is returned to the page. But if not, a new response requested from the network.
- **NetworkFirst** - In this function, firstly we can try getting an updated response from the network, if this process completed successfully, the new response will be cached and returned. But if this process fails, we check whether the request has been cached before or not. If a cache exists, it is returned to the page, but if not, this is up to you. You can return dummy content or information messages to the page.

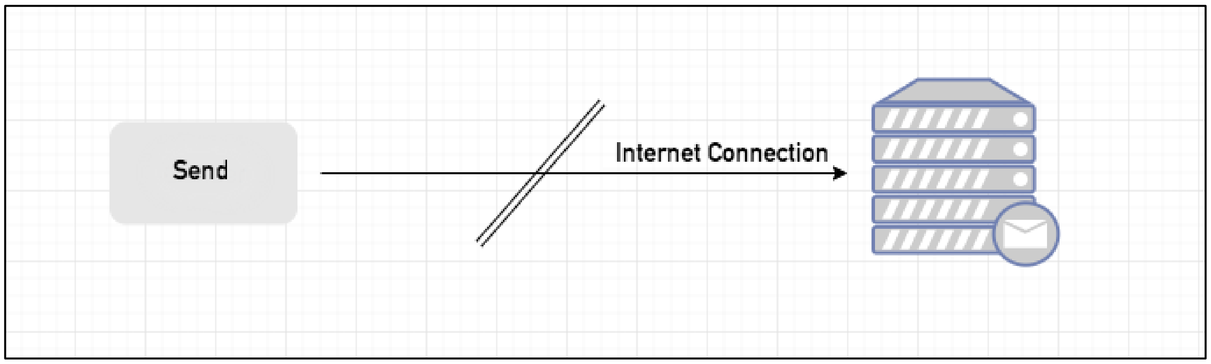


```
self.addEventListener("fetch", function (event) {
    const req = event.request;
    const url = new URL(req.url);

    if (url.origin === location.origin) {
        event.respondWith(cacheFirst(req));
    }
    else {
        event.respondWith(networkFirst(req));
    }
});

async function cacheFirst(req) {
    return await caches.match(req) || fetch(req);
}

async function networkFirst(req) {
    const cache = await caches.open("pwa-dynamic");
    try {
        const res = await fetch(req);
        cache.put(req, res.clone());
        return res;
    } catch (error) {
        const cachedResponse = await cache.match(req);
        return cachedResponse || await caches.match("./noconnection.json");
    }
}
```

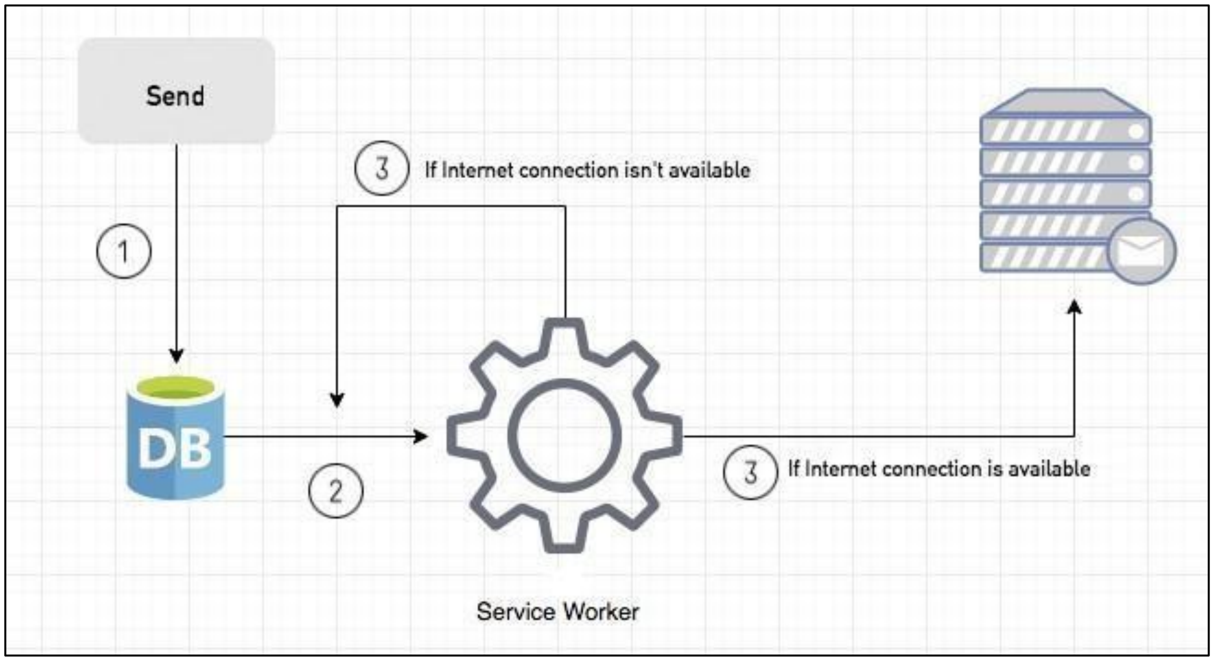
Sync Event

Background Sync is a Web API that is used to delay a process until the Internet connection is stable. We can adapt this definition to the real world; there is an e-mail client application that works on the browser and we want to send an email with this tool. Internet connection is broken while we are writing e-mail content and we didn't realize it. When completing the writing, we click the send button.

Here is a job for the Background Sync.
The following view shows the classical process of sending email to us. If the Internet Connection is broken, we can't send any content to Mail Server.



Here, you can create any scenario for yourself. A sample is in the following for this case.

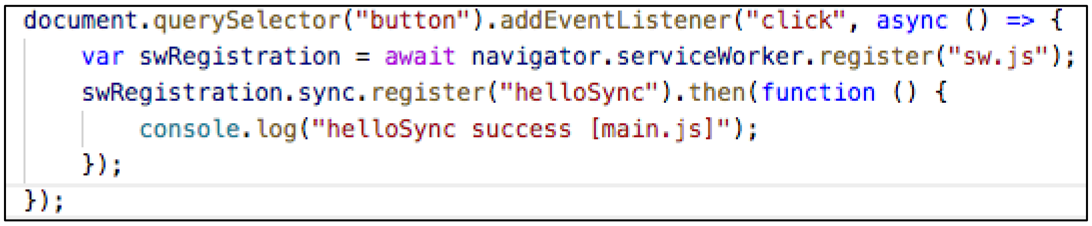

Send
1
DB
2
3 If Internet connection isn't available
3 If Internet connection is available
Service Worker

1. When we click the “send” button, email content will be saved to IndexedDB.
2. Background Sync registration.
3. **If the Internet connection is available**, all email content will be read and sent to Mail Server.

If the Internet connection is unavailable, the service worker waits until the connection is available even though the window is closed. When it is available, email content will be sent to Mail Server.

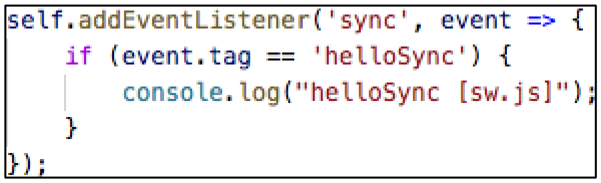
You can see the working process within the following code block.

Event Listener for Background Sync Registration



```
document.querySelector("button").addEventListener("click", async () => {
    var swRegistration = await navigator.serviceWorker.register("sw.js");
    swRegistration.sync.register("helloSync").then(function () {
        console.log("helloSync success [main.js]");
    });
});
```

Event Listener for sw.js



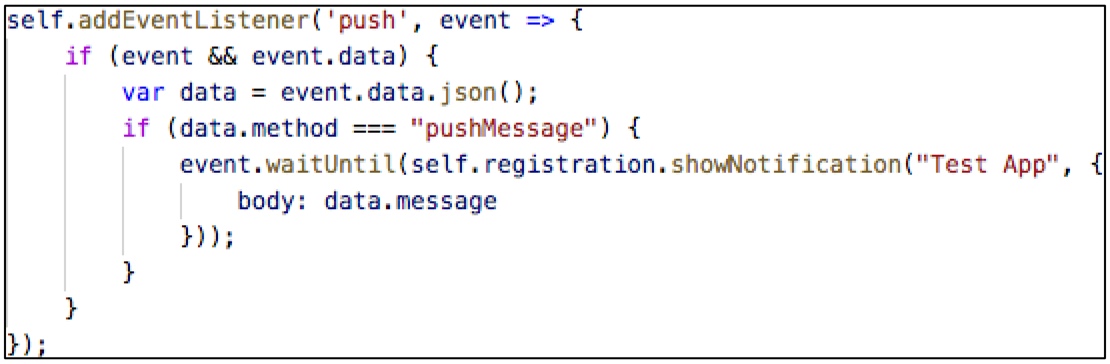
```
self.addEventListener('sync', event => {
    if (event.tag == 'helloSync') {
        console.log("helloSync [sw.js]");
    }
});
```

Push Event

This is the event that handles push notifications that are received from the server. You can apply any method with received data.

We can check in the following example.
“Notification.requestPermission();” is the necessary line to show notification to the user. If you don’t want to show any notification, you don’t need this line.

In the following code block is in sw.js file. You can handle push notifications with this event. In this example, I kept it simple. We send an object that has “method” and “message” properties. If the method value is “pushMessage”, we open the information notification with the “message” property.



```
self.addEventListener('push', event => {
    if (event && event.data) {
        var data = event.data.json();
        if (data.method === "pushMessage") {
            event.waitUntil(self.registration.showNotification("Test App", {
                body: data.message
            }));
        }
    }
});
```

You can use Application Tab from Chrome Developer Tools for testing push notification.

Code:

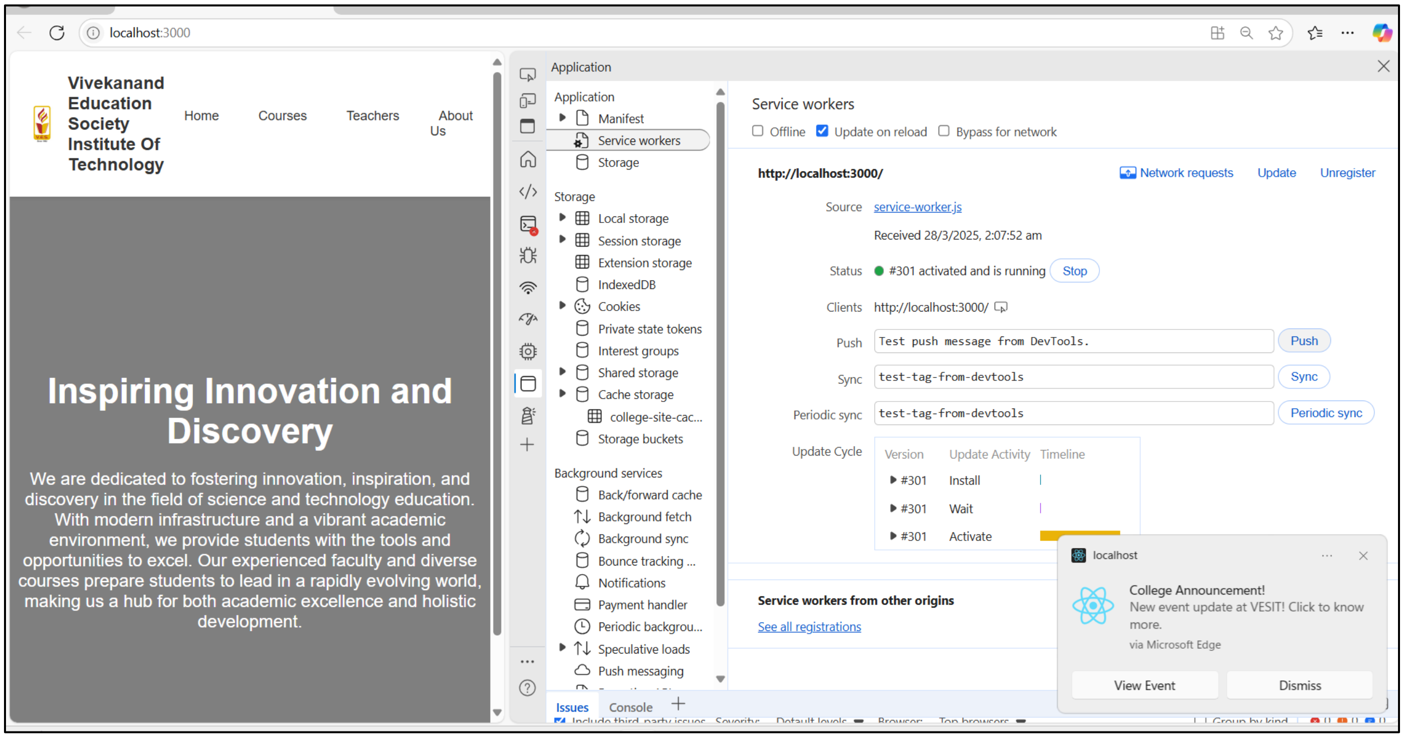
1. **Push Notification code:**

```
/ Push Notification event - College Announcements
self.addEventListener("push", async (event) => {
    if (!self.registration || Notification.permission !== "granted") {
        console.warn("Push notification permission not granted.");
        return;
    }

    const options = {
        body: "New event update at VESIT! Click to know more.",
        icon: "/logo192.png",
        badge: "/logo192.png",
        actions: [
            { action: "view", title: "View Event" },
            { action: "close", title: "Dismiss" }
        ]
    };

    event.waitUntil(
        self.registration.showNotification("College Announcement!", options)
            .catch(error => console.error("Error showing notification:", error))
    );
});

// Handle Notification Click - Redirect to Event Page
self.addEventListener("notificationclick", (event) => {
    event.notification.close();

    if (event.action === "view") {
        event.waitUntil(
            clients.matchAll({ type: "window", includeUncontrolled: true }).then((clientList) => {
                if (clientList.length > 0) {
                    return clientList[0].focus();
                }
                return clients.openWindow("/events").catch((error) => {
                    console.error("Failed to open events page:", error);
                });
            })
        );
    }
});
```

2. **Fetch and Sync Code:**

```
// Fetch event - Serve files from cache & dynamically cache new requests
self.addEventListener("fetch", (event) => {
  if (event.request.method !== "GET") return; // Handle only GET requests

  event.respondWith(
    caches.match(event.request).then((cachedResponse) => {
      if (cachedResponse) {
        return cachedResponse; // Return cached response if available
      }

      return fetch(event.request)
        .then((fetchResponse) => {
          return caches.open(CACHE_NAME).then((cache) => {
            cache.put(event.request, fetchResponse.clone());
            return fetchResponse;
          });
        })
        .catch(() => {
          return new Response("You are offline. Please check your connection.", {
            status: 503,
            headers: { "Content-Type": "text/plain" }
          });
        });
    })
  );
});

// Background Sync - Preload Important Pages
self.addEventListener("sync", (event) => {
  if (event.tag === "preload-pages") {
    event.waitUntil(preloadImportantPages());
  }
});

async function preloadImportantPages() {
  console.log("Preloading important pages...");
  const pages = ["/courses", "/teachers", "/about"];
  try {
    const cache = await caches.open(CACHE_NAME);
    await Promise.all(pages.map(async (page) => {
      const response = await fetch(page);
      if (response.ok) {
        cache.put(page, response.clone());
      }
    }));
    console.log("Pages preloaded successfully.");
  } catch (error) {
    console.error("Preloading failed:", error);
  }
}
```

OUTPUT:-


localhost:3000

Vivekanand Education Society Institute Of Technology

Home Courses Teachers About Us

Inspiring Innovation and Discovery

We are dedicated to fostering innovation, inspiration, and discovery in the field of science and technology education. With modern infrastructure and a vibrant academic environment, we provide students with the tools and opportunities to excel. Our experienced faculty and diverse courses prepare students to lead in a rapidly evolving world, making us a hub for both academic excellence and holistic development.

Application

Application: Manifest, Service workers, Storage

Storage: Local storage, Session storage, Extension storage, IndexedDB, Cookies, Private state tokens, Interest groups, Shared storage, Cache storage, college-site-cac..., Storage buckets

Background services: Back/forward cache, Background fetch, Background sync, Bounce tracking ..., Notifications, Payment handler, Periodic backgrou..., Speculative loads, Push messaging

Service workers

Offline ✓ Update on reload Bypass for network

http://localhost:3000/ Network requests Update Unregister

Source	service-worker.js
	Received 28/3/2025, 2:07:52 am
Status	#301 activated and is running Stop
Clients	http://localhost:3000/
Push	Test push message from DevTools. Push
Sync	test-tag-from-devtools Sync
Periodic sync	test-tag-from-devtools Periodic sync

Update Cycle

Version	Update Activity	Timeline
#301	Install	
#301	Wait	
#301	Activate	

Service workers from other origins

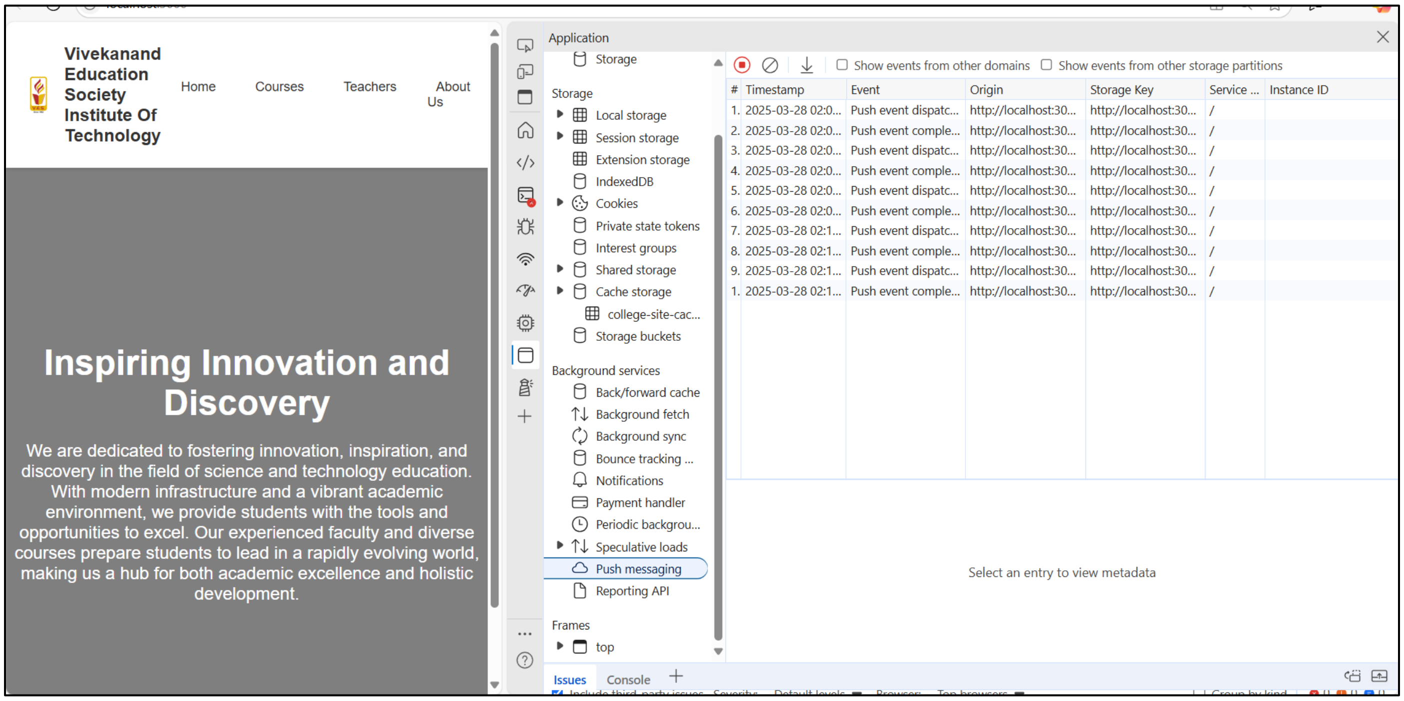
See all registrations

localhost

College Announcement!
New event update at VESIT! Click to know more.
via Microsoft Edge

View Event Dismiss

Issues Console


Vivekanand Education Society Institute Of Technology

Home Courses Teachers About Us

Inspiring Innovation and Discovery

We are dedicated to fostering innovation, inspiration, and discovery in the field of science and technology education. With modern infrastructure and a vibrant academic environment, we provide students with the tools and opportunities to excel. Our experienced faculty and diverse courses prepare students to lead in a rapidly evolving world, making us a hub for both academic excellence and holistic development.

Application

Storage: Local storage, Session storage, Extension storage, IndexedDB, Cookies, Private state tokens, Interest groups, Shared storage, Cache storage, college-site-cac..., Storage buckets

Background services: Back/forward cache, Background fetch, Background sync, Bounce tracking ..., Notifications, Payment handler, Periodic backgrou..., Speculative loads, Push messaging, Reporting API

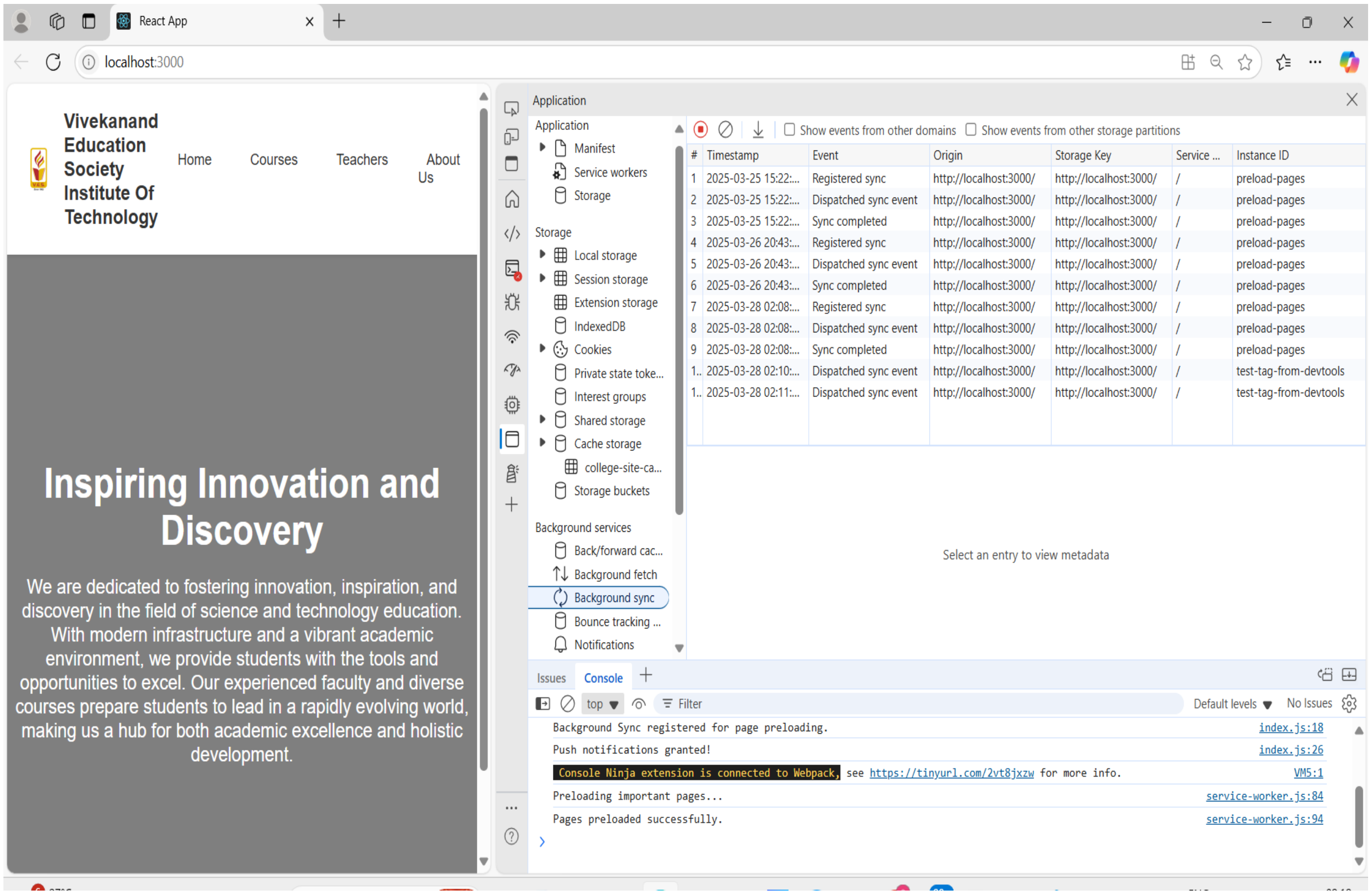
Frames: top

Show events from other domains Show events from other storage partitions

#	Timestamp	Event	Origin	Storage Key	Service ...	Instance ID
1.	2025-03-28 02:0...	Push event dispatc...	http://localhost:30...	http://localhost:30...	/	
2.	2025-03-28 02:0...	Push event comple...	http://localhost:30...	http://localhost:30...	/	
3.	2025-03-28 02:0...	Push event dispatc...	http://localhost:30...	http://localhost:30...	/	
4.	2025-03-28 02:0...	Push event comple...	http://localhost:30...	http://localhost:30...	/	
5.	2025-03-28 02:0...	Push event dispatc...	http://localhost:30...	http://localhost:30...	/	
6.	2025-03-28 02:0...	Push event comple...	http://localhost:30...	http://localhost:30...	/	
7.	2025-03-28 02:1...	Push event dispatc...	http://localhost:30...	http://localhost:30...	/	
8.	2025-03-28 02:1...	Push event comple...	http://localhost:30...	http://localhost:30...	/	
9.	2025-03-28 02:1...	Push event dispatc...	http://localhost:30...	http://localhost:30...	/	
1.	2025-03-28 02:1...	Push event comple...	http://localhost:30...	http://localhost:30...	/	

Select an entry to view metadata

Issues Console



Inspiring Innovation and Discovery

We are dedicated to fostering innovation, inspiration, and discovery in the field of science and technology education. With modern infrastructure and a vibrant academic environment, we provide students with the tools and opportunities to excel. Our experienced faculty and diverse courses prepare students to lead in a rapidly evolving world, making us a hub for both academic excellence and holistic development.

#	Timestamp	Event	Origin	Storage Key	Service ...	Instance ID
1	2025-03-25 15:22:...	Registered sync	http://localhost:3000/	http://localhost:3000/	/	preload-pages
2	2025-03-25 15:22:...	Dispatched sync event	http://localhost:3000/	http://localhost:3000/	/	preload-pages
3	2025-03-25 15:22:...	Sync completed	http://localhost:3000/	http://localhost:3000/	/	preload-pages
4	2025-03-26 20:43:...	Registered sync	http://localhost:3000/	http://localhost:3000/	/	preload-pages
5	2025-03-26 20:43:...	Dispatched sync event	http://localhost:3000/	http://localhost:3000/	/	preload-pages
6	2025-03-26 20:43:...	Sync completed	http://localhost:3000/	http://localhost:3000/	/	preload-pages
7	2025-03-28 02:08:...	Registered sync	http://localhost:3000/	http://localhost:3000/	/	preload-pages
8	2025-03-28 02:08:...	Dispatched sync event	http://localhost:3000/	http://localhost:3000/	/	preload-pages
9	2025-03-28 02:08:...	Sync completed	http://localhost:3000/	http://localhost:3000/	/	preload-pages
1..	2025-03-28 02:10:...	Dispatched sync event	http://localhost:3000/	http://localhost:3000/	/	test-tag-from-devtools
1..	2025-03-28 02:11:...	Dispatched sync event	http://localhost:3000/	http://localhost:3000/	/	test-tag-from-devtools

```
Background Sync registered for page preloading.                                   index.js:18
Push notifications granted!                                                       index.js:26
Console Ninja extension is connected to Webpack, see https://tinyurl.com/2vt8jxzw for more info.   VM5:1
Preloading important pages...                                             service-worker.js:84
Pages preloaded successfully.                                             service-worker.js:94
```

Sonam chhabaidiya/ palak Chanchlani/ Mahi jodhani D15A 09/05/21

EXPERIMENT NO: - 10

AIM: - To study and implement deployment of Ecommerce PWA to GitHub Pages.

Theory: -

GitHub Pages: Static Website Hosting Made Simple

GitHub Pages is a free hosting service that allows users to publish **public webpages directly from a GitHub repository**. It is particularly useful for **static websites, project documentation, and blogs**.

Key Features

- **Jekyll Integration**: Built-in support for Jekyll enables easy blogging.
- **Custom Domains**: Allows users to configure their own URLs.
- **Automatic Page Deployment**: Simply push your changes to the repository, and the updates go live.

Why Choose GitHub Pages?

- **Completely Free**: No hosting charges.
- **Seamless GitHub Integration**: Works directly with your repositories.
- **Quick Setup**: Just create a repository, push your files, and your site is live.

Who Uses GitHub Pages?

Companies like **Lyft, CircleCI, and HubSpot** use GitHub Pages for their documentation and static sites. It is widely adopted, appearing in **775 company stacks and 4,401 developer stacks**.

Pros & Cons

Pros

- Familiar interface for GitHub users.

- Simple deployment via the gh-pages branch.
- Supports custom domains with easy DNS configuration.

Cons

- Repositories need to be public unless you have a paid plan.
- Limited HTTPS support for custom domains (expected to improve).
- Jekyll plugins have limited support.

Firebase: A Full-Featured Real-Time Backend

Firebase is a cloud-based **real-time application platform** developed by Google. It enables developers to build **dynamic, collaborative applications** with ease.

Key Features

- **Real-Time Database**: Automatically syncs data across all connected clients.
- **Cloud-Based Storage**: JSON-based storage accessible via REST APIs.
- **Scalable Infrastructure**: Works well with existing services and scales automatically.
- **Authentication & Cloud Messaging**: Secure login and push notifications.

Why Choose Firebase?

- **Instant Backend Setup**: No need to build a separate backend.
- **Fast & Responsive**: Real-time data synchronization.
- **Built-in HTTPS**: Free SSL certificates for custom domains.

Who Uses Firebase?

Companies like **Instacart, 9GAG, and Twitch** rely on Firebase for their backend needs. Firebase is widely adopted, appearing in **1,215 company stacks and 4,651 developer stacks**.

Pros & Cons

Pros

- **Hosted by Google**, ensuring reliability and security.
- **Comes with authentication, messaging, and real-time database services.**
- **Free HTTPS support** for all custom domains.

Cons

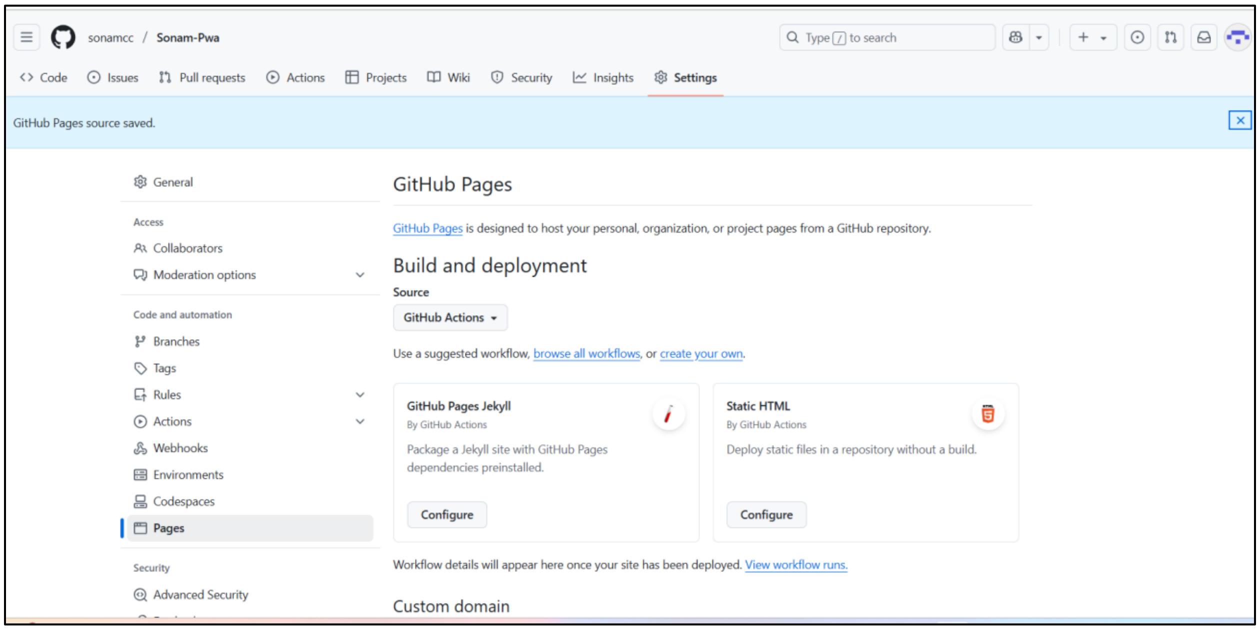
- **Limited Free Plan**: 10 GB of data transfer per month (can be mitigated with a CDN).
- **Command-Line Deployment**: No GUI for hosting.
- **No Built-in Static Site Generator Support**: Unlike GitHub Pages, Firebase doesn't natively support static site generators like Jekyll.

Link to our GitHub repository:

Sonam-Pwa

Hosted link:
https://sonamcc.github.io/Sonam-Pwa/

Github Screenshot:



GitHub Pages source saved.

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Build and deployment

Source

GitHub Actions

Use a suggested workflow, browse all workflows, or create your own.

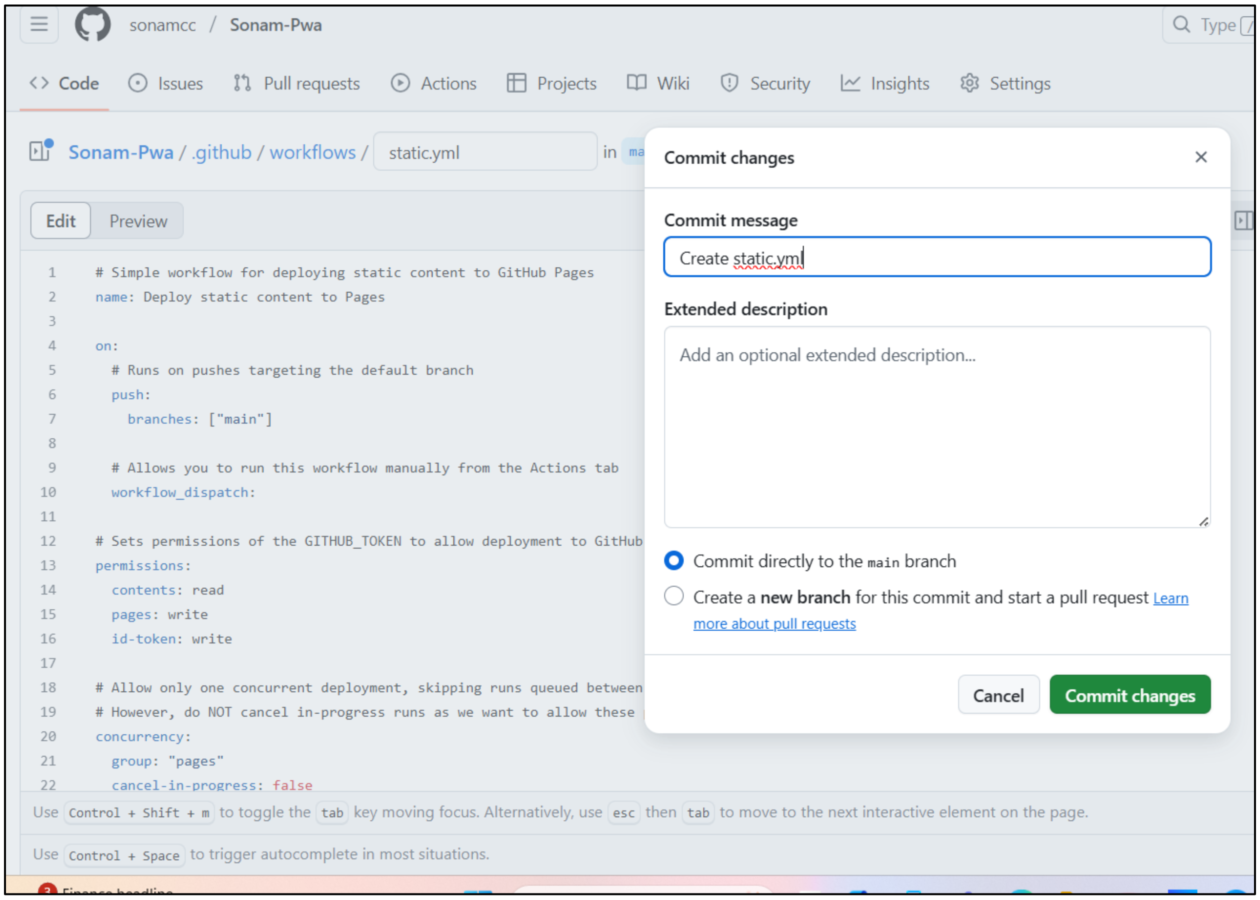
GitHub Pages Jekyll
By GitHub Actions
Package a Jekyll site with GitHub Pages dependencies preinstalled.
Configure

Static HTML
By GitHub Actions
Deploy static files in a repository without a build.
Configure

Workflow details will appear here once your site has been deployed. View workflow runs.

Custom domain

Sonam chhabaidiya/ palak Chanchlani/ Mahi jodhani D15A 09/05/21


sonamcc / Sonam-Pwa
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
Sonam-Pwa / .github / workflows / static.yml in main
Edit Preview

```
# Simple workflow for deploying static content to GitHub Pages
name: Deploy static content to Pages

on:
  # Runs on pushes targeting the default branch
  push:
    branches: ["main"]

  # Allows you to run this workflow manually from the Actions tab
  workflow_dispatch:

# Sets permissions of the GITHUB_TOKEN to allow deployment to GitHub
permissions:
  contents: read
  pages: write
  id-token: write

# Allow only one concurrent deployment, skipping runs queued between
# However, do NOT cancel in-progress runs as we want to allow these
concurrency:
  group: "pages"
  cancel-in-progress: false
```

Commit changes

Commit message: Create static.yml

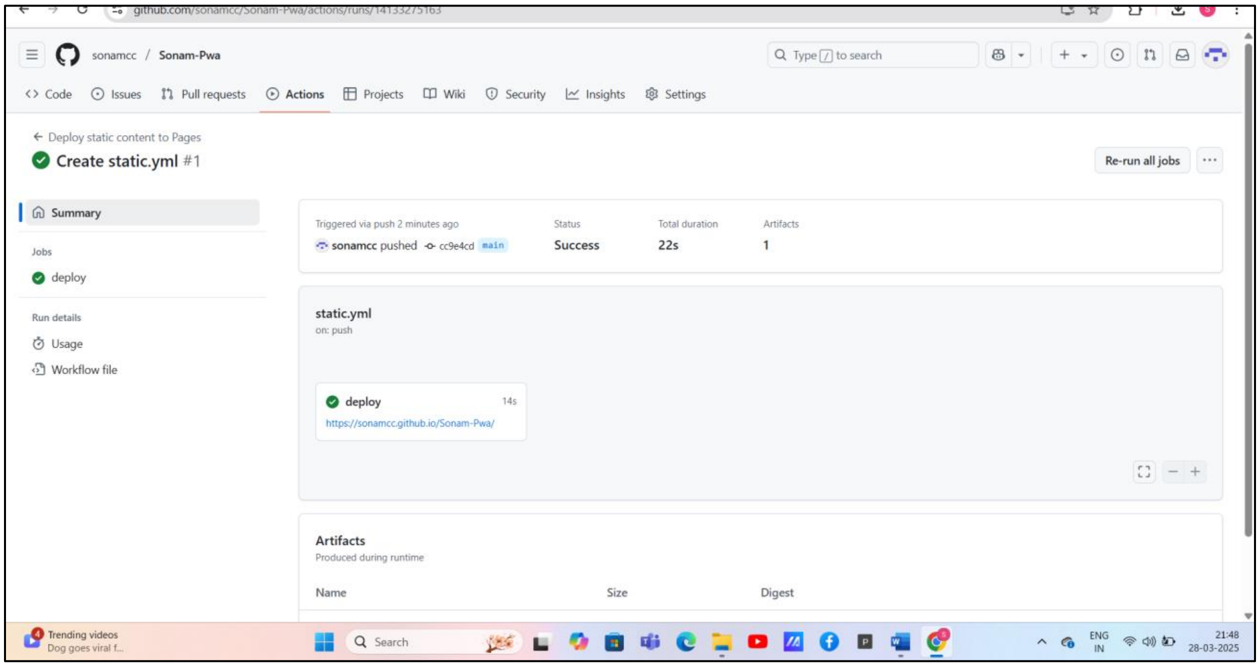
Extended description: Add an optional extended description...

[✓ Commit directly to the main branch]

Create a new branch for this commit and start a pull request Learn more about pull requests

Use Control + Shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.

Use Control + Space to trigger autocomplete in most situations.


github.com/sonamcc/Sonam-Pwa/actions/runs/14133275163
sonamcc / Sonam-Pwa
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
Re-run all jobs
Summary
Jobs
deploy
Run details
Usage
Workflow file

← Deploy static content to Pages

Create static.yml #1

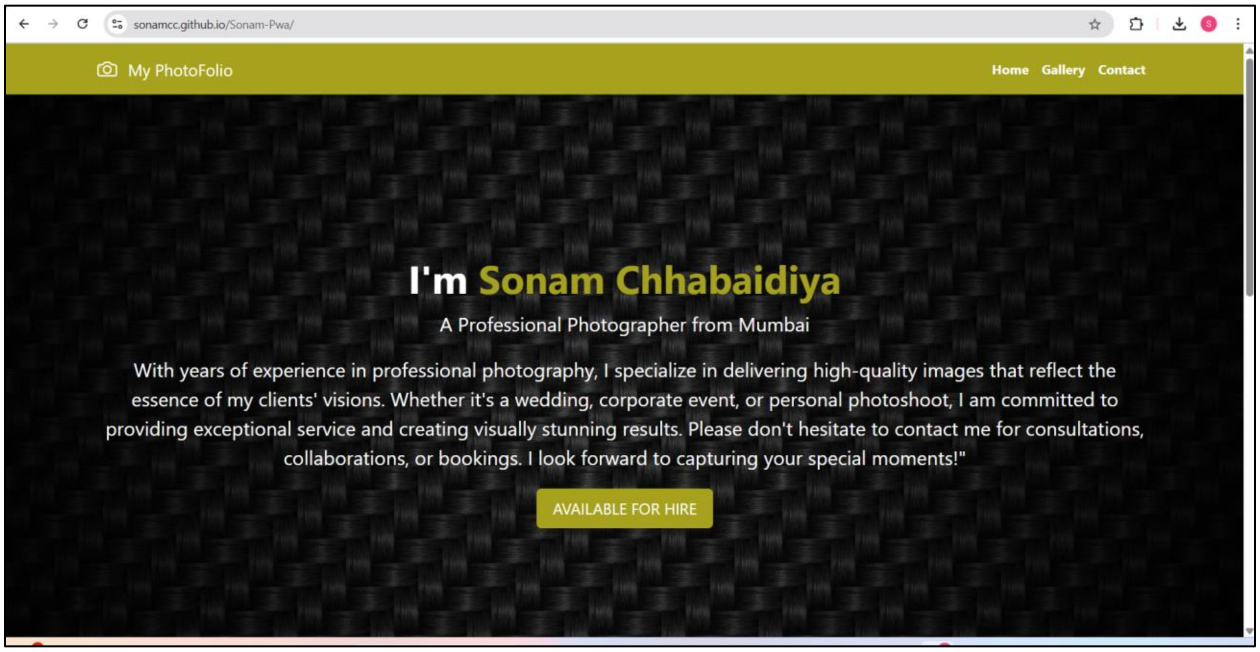
Triggered via push 2 minutes ago	Status	Total duration	Artifacts
sonamcc pushed cc9e4cd main	Success	22s	1

static.yml
on: push

deploy 14s
https://sonamcc.github.io/Sonam-Pwa/

Artifacts
Produced during runtime

Name	Size	Digest



I'm Sonam Chhabaidiya

A Professional Photographer from Mumbai

With years of experience in professional photography, I specialize in delivering high-quality images that reflect the essence of my clients' visions. Whether it's a wedding, corporate event, or personal photoshoot, I am committed to providing exceptional service and creating visually stunning results. Please don't hesitate to contact me for consultations, collaborations, or bookings. I look forward to capturing your special moments!"

Experiment 11

AIM: - To use google Lighthouse PWA Analysis Tool to test the PWA functioning.

Theory: - Google Lighthouse is an open-source tool that audits web applications based on multiple key parameters, including performance, accessibility, Progressive Web App (PWA) implementation, and best practices. It provides a detailed, automated report that helps developers optimize their websites efficiently. Unlike traditional manual audits, which can take days or even weeks, Lighthouse generates insights within minutes.
One of the key advantages of Lighthouse is its ease of use—no complex setup is required. Simply run it on a webpage or provide a URL, and it will generate an extensive performance report.

Key Features and Audit Metrics
Lighthouse can audit both desktop and mobile versions of a webpage. The core evaluation criteria include:

1. Performance
This metric measures how efficiently a webpage loads and displays content.
Key factors influencing the performance score include:

- Page load speed – How quickly the page becomes visible to the user.
- First Contentful Paint (FCP) – The time taken for the first piece of content to appear.
- Largest Contentful Paint (LCP) – The time taken for the main content to fully load.
- Cumulative Layout Shift (CLS) – Measures how visually stable a page is (i.e., avoiding unexpected shifts in content).
- Time to Interactive (TTI) – The time it takes for the page to become fully functional.

Lighthouse assigns a score from 0 to 100 based on percentile rankings, where:

- 100 → Top 2% of websites (98th percentile)
- 50 → Around the 75th percentile
- Lower scores → Indicate areas that need optimization

2. Progressive Web App (PWA) Score (Mobile)
With the rise of PWAs, modern web applications aim to provide a native app-like experience. Lighthouse evaluates the PWA implementation based on Google's Baseline PWA Checklist, which includes:

- Service Worker implementation – Ensuring offline support and background synchronization.
- App Manifest compliance – Providing metadata for better mobile integration.
- Viewport configuration – Optimizing mobile responsiveness.
- Performance in script-disabled environments – Ensuring the page functions even when JavaScript is disabled.

A high PWA score indicates that the application meets essential PWA criteria and provides an app-like user experience.

3. Accessibility
Accessibility ensures that web applications are usable by individuals with disabilities. Lighthouse audits a webpage based on:

- ARIA attributes – Enhancing accessibility through attributes like aria-required.
- Text alternatives for media – Ensuring audio and visual content is accessible.
- Semantic HTML – Proper use of , , , and other elements that improve screen-reader compatibility.

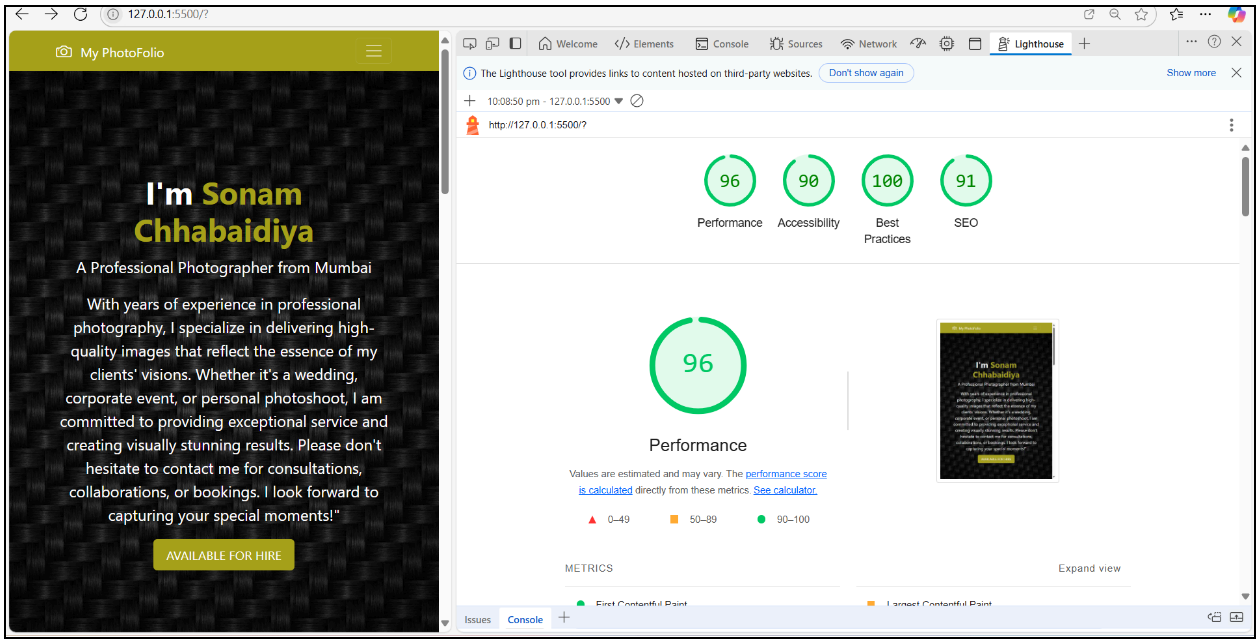
Unlike other metrics, accessibility checks follow a pass/fail approach—if a necessary feature is missing, it significantly impacts the score. A higher accessibility score ensures inclusivity for users with visual or cognitive impairments

Manifest.json

```
{
    "name": "PhotoFolio",
    "short_name": "PhotoFolio",
    "start_url": "/",
    "display": "standalone",
    "background_color": "#090909",
    "theme_color": "#a7a21d",
    "icons": [
      {
        "src":
"https://bootstrapmade.com/content/demo/PhotoFolio/assets/img/gallery/gall
ery-1.jpg",
        "sizes": "192x192",
        "type": "image/png"
      },
      {
        "src":
"https://bootstrapmade.com/content/demo/PhotoFolio/assets/img/gallery/gall
ery-2.jpg",
        "sizes": "512x512",
        "type": "image/png"
      }
    ]
  }
```

Sonam chhabaidiya/ Palak Chanchlani / Mahi jodhani D15A 09/05/21

Output:



127.0.0.1:5500/?

My PhotoFolio

I'm Sonam Chhabaidiya

A Professional Photographer from Mumbai

With years of experience in professional photography, I specialize in delivering high-quality images that reflect the essence of my clients' visions. Whether it's a wedding, corporate event, or personal photoshoot, I am committed to providing exceptional service and creating visually stunning results. Please don't hesitate to contact me for consultations, collaborations, or bookings. I look forward to capturing your special moments!"

AVAILABLE FOR HIRE

The Lighthouse tool provides links to content hosted on third-party websites. Don't show again Show more

10:08:50 pm - 127.0.0.1:5500

http://127.0.0.1:5500/?

96 Performance
90 Accessibility
100 Best Practices
91 SEO

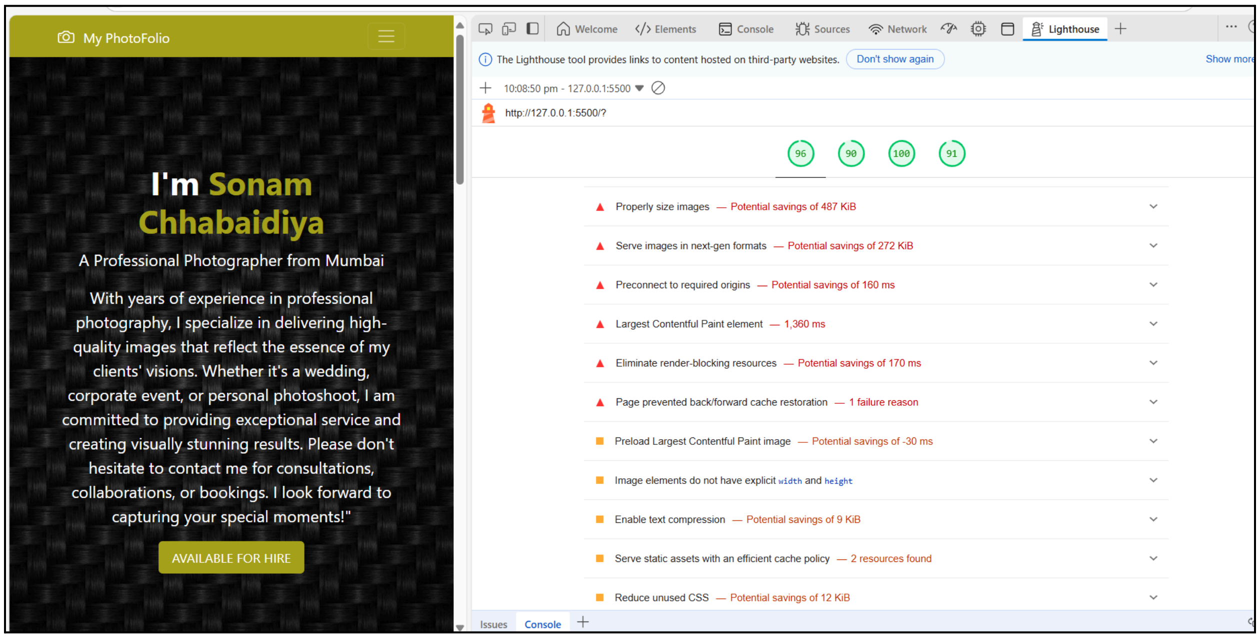
96

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

METRICS Expand view



The Lighthouse tool provides links to content hosted on third-party websites. Don't show again

10:08:50 pm - 127.0.0.1:5500

http://127.0.0.1:5500/?

96 90 100 91

▲ Properly size images — Potential savings of 487 KiB
▲ Serve images in next-gen formats — Potential savings of 272 KiB
▲ Preconnect to required origins — Potential savings of 160 ms
▲ Largest Contentful Paint element — 1,360 ms
▲ Eliminate render-blocking resources — Potential savings of 170 ms
▲ Page prevented back/forward cache restoration — 1 failure reason
■ Preload Largest Contentful Paint image — Potential savings of -30 ms
■ Image elements do not have explicit `width` and `height`
■ Enable text compression — Potential savings of 9 KiB
■ Serve static assets with an efficient cache policy — 2 resources found
■ Reduce unused CSS — Potential savings of 12 KiB