Clevertap Website Sdk Integration

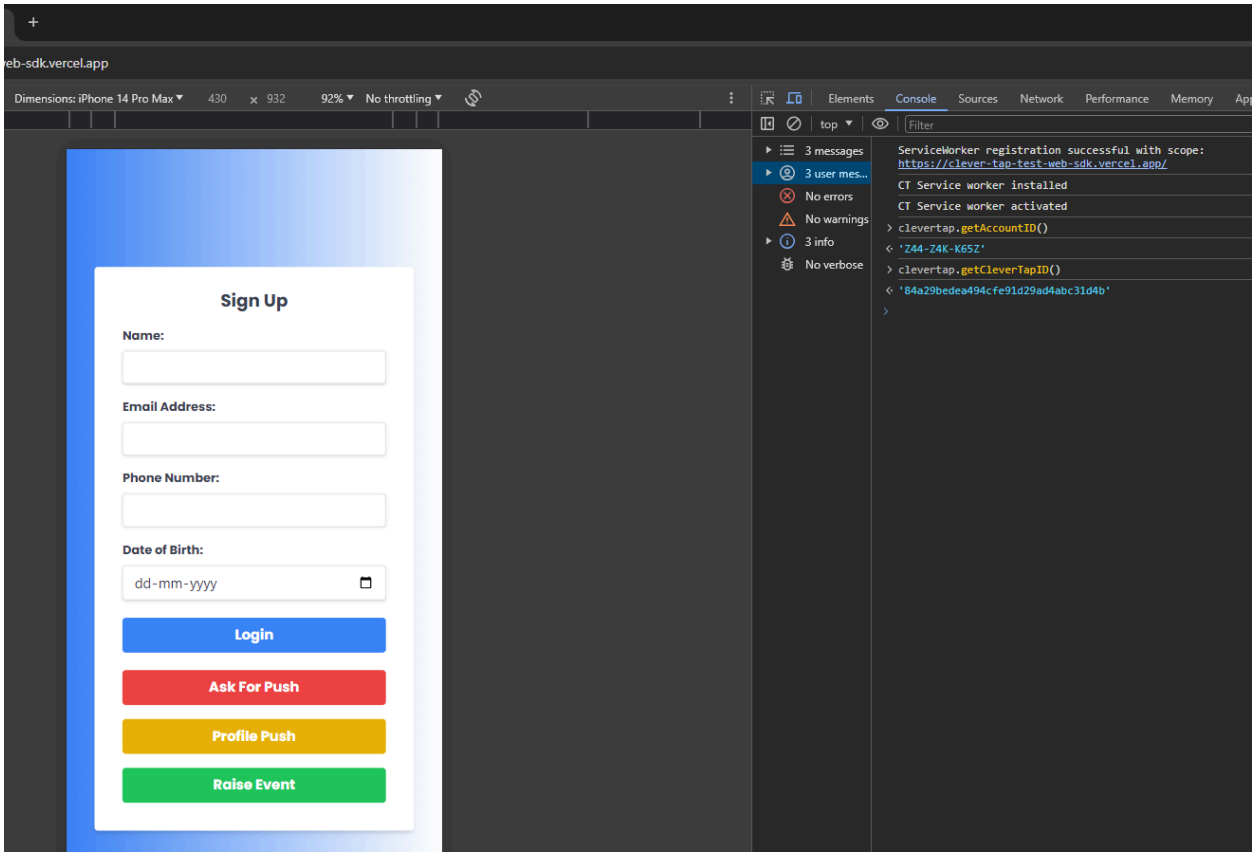
Introduction

CleverTap is a customer retention platform that provides the functionality to integrate app analytics and marketing. The platform helps customers increase user engagement in three ways:

- Tracks actions users are taking and analyzes how people use the product.
- Segment users based on their actions and run targeted campaigns to these segments.
- Analyze each campaign to understand their effect on user engagement and business metrics.

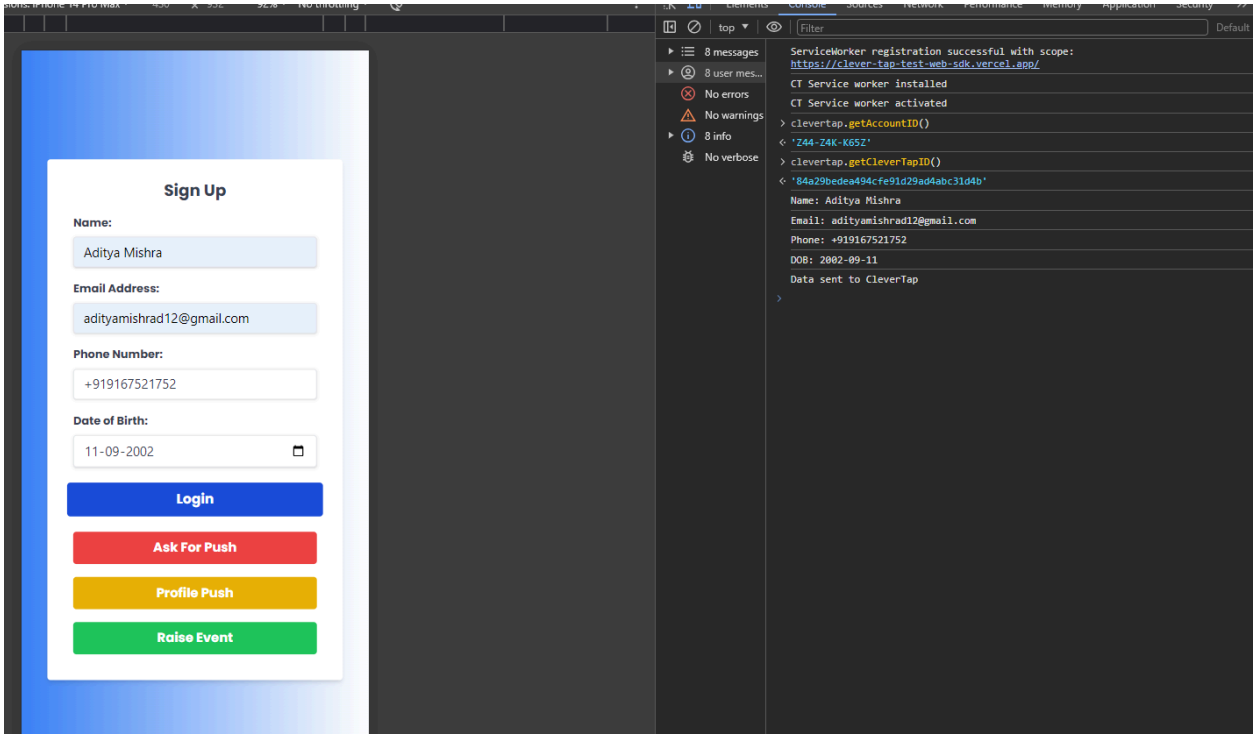
Working of CleverTap Integration

1.Verifying CleverTap Integration:

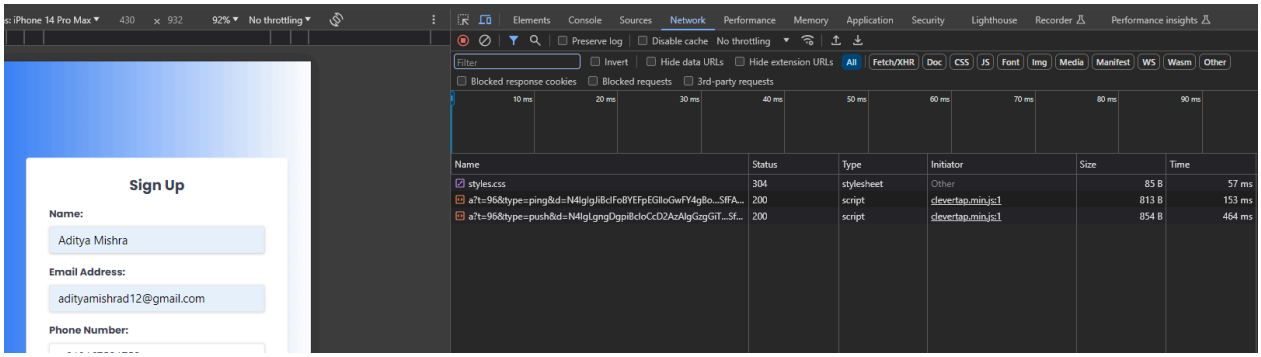


The first step in demonstrating how your website works is to confirm that CleverTap has been successfully integrated. This can be done by using the `clevertap.getAccountId` and `clevertap.getCleverTapID` methods. These methods retrieve the account ID and CleverTap ID respectively, which are unique identifiers provided by CleverTap. If these methods return valid IDs, it indicates that CleverTap has been correctly integrated into your website.

2.User Login Tracking:

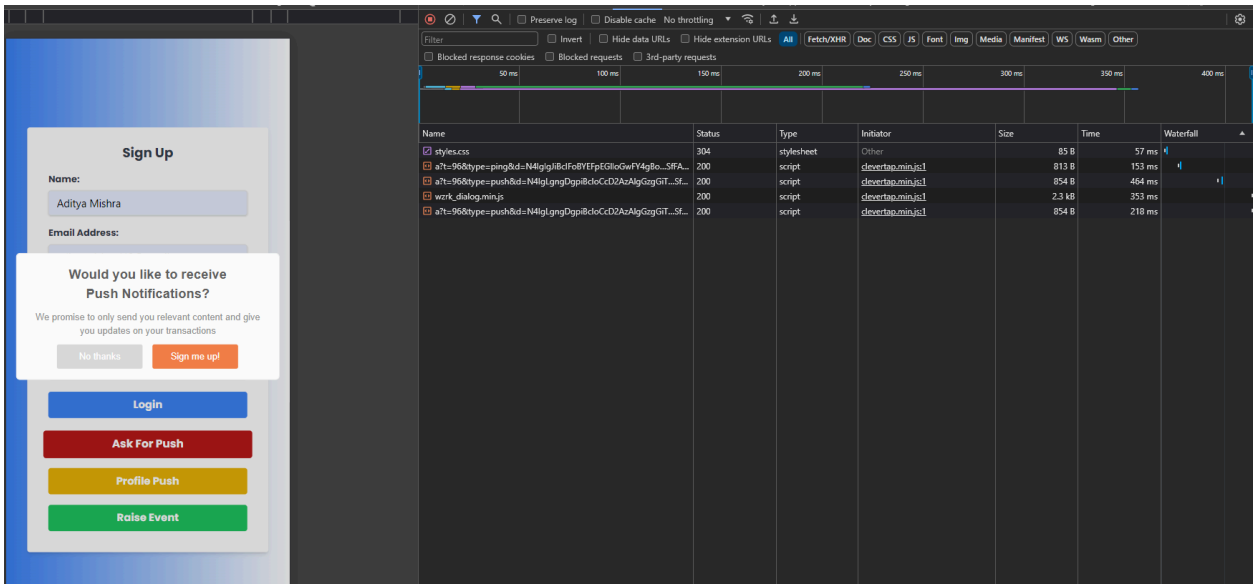


The second step involves tracking user logins. When the Login button is clicked, a CleverTap function is called. This function pushes user data (including name, email, phone, and date of birth) to CleverTap.



You can confirm that the data has been successfully sent by checking your network console. If it displays a 200 OK status, this means the data has been accepted by the server. Additionally, you can view the sent data in your console. This allows you to verify that the correct information is being tracked.

3.Push Notification Request




```
Push notification request sent to CleverTap
Name: Aditya Mishra
Email: adityamishrad12@gmail.com
Phone: +919167521752
DOB: 2002-09-11
Data sent to CleverTap
```

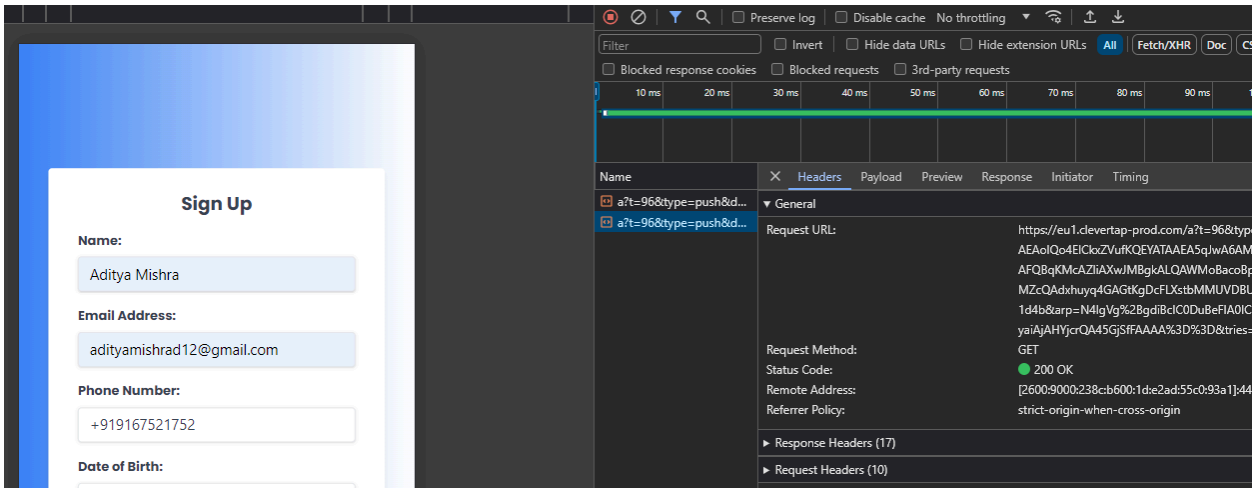
The third step involves asking the user if they would like to receive push notifications. This is done by adding an “Ask For Push” button to your website. When this button is clicked, a CleverTap function is called.This function sends a push notification request to the user with a title, a message body, and two buttons: one to accept (‘Sign me up!’) and one to reject (‘No thanks’). If the user doesn’t respond within 5 seconds, the request will be sent again. The ‘Sign me up!’ button is colored ‘#f28046’ for visibility.

4.Profile Data Push:

The fourth step involves updating the user’s profile data in CleverTap. This is done by adding a “Profile Push” button to your website. When this button is clicked, a CleverTap function is called.

This function pushes the user's data (including name, email, phone, and date of birth) to CleverTap.

```
Profile data pushed to CleverTap
Name: Aditya Mishra
Email: adityamishrad12@gmail.com
Phone: +919167521752
DOB: 2024-05-09
Data sent to CleverTap
```



This retrieves the values of the name, email, phone, and date of birth fields, and pushes this data to CleverTap. This allows CleverTap to keep track of the user's profile data, which can be used for personalized marketing and user engagement.

5.Event Tracking

```
Event pushed to CleverTap
Name: Aditya Mishra
Email: adityamishrad12@gmail.com
Phone: +919167521752
DOB: 2024-05-09
Data sent to CleverTap
```

The fifth step involves tracking user interactions or events. For this, you need to use CleverTap's event.push() function when the Event button is clicked. This function

allows you to add various properties to a custom event, which can be of different types such as date-time, string, integer, and float.

6.Enabling Logs:

Lastly, to verify your integration, you can enable logs in your browser console. This can be done by entering the command sessionStorage['WZRK_D'] = ";. This will allow you to see detailed logs of the CleverTap functions in your console, helping you debug and verify that everything is working as expected.

Implementation

CleverTap is integrated into the website using its SDK, added to the HTML via a script tag. User interactions with specific HTML elements are tracked as events in CleverTap through JavaScript event listeners. The website's design is handled by Tailwind CSS, ensuring a consistent user experience. This setup enables valuable data collection and analysis of user interactions.

Setup and Integration

In this section, we delve deeper into the code employed for integrating CleverTap into a web application. The JavaScript code enables tracking of user interactions and facilitates sending push notifications.

CleverTap Object Initialization

The clevertap object is initialized with several empty arrays and properties. These will be used to store event data, user profile data, account data, region data, notification data, and privacy settings.

```
var clevertap = {
  event: [],
  profile: [],
  account: [],
  onUserLogin: [],
  region: "",
  notifications: [],
  privacy: [],
};
```

Account and Privacy Settings

The `clevertap.account` array is populated with the CleverTap account ID. The `clevertap.privacy` array is updated with privacy settings, including whether to opt out of data collection and whether to use the user's IP address for data collection.

```
clevertap.account.push({ id: "Z44-Z4K-K65Z" });
clevertap.privacy.push({ optOut: false });
clevertap.privacy.push({ useIP: false });
```

CleverTap SDK Integration

The CleverTap SDK is integrated into the web application by creating a new script element, setting its source to the CleverTap SDK URL, and appending it to the document.

```
(function () {
  var wzrk = document.createElement("script");
  wzrk.type = "text/javascript";
  wzrk.async = true;
  wzrk.src =
    ("https:" == document.location.protocol
      ? "https://d2r1yp2w7bby2u.cloudfront.net"
      : "http://static.clevertap.com") + "/js/clevertap.min.js";
  var s = document.getElementsByTagName("script")[0];
  s.parentNode.insertBefore(wzrk, s);
})();
```

User Login Event Tracking

An event listener is added to the signup form. When the form is submitted, the event listener prevents the default form submission, retrieves the values of the name, email, phone, and date of birth fields, and logs a user login event in CleverTap.

```
document
  .getElementById("signupForm")
  .addEventListener("submit", function (event) {
    event.preventDefault();

    var name = document.getElementById("name").value;
    var email = document.getElementById("email").value;
    var phone = document.getElementById("phone").value;
    var dob = document.getElementById("dob").value;

    console.log("Name:", name);
```

```
    console.log("Email:", email);
    console.log("Phone:", phone);
    console.log("DOB:", dob);

    var phoneRegex = /^\+\d+$/;
    if (!phoneRegex.test(phone)) {
      alert("Phone number should be formatted as +[country code][number]");
      return;
    }

    clevertap.onUserLogin.push({
      Site: {
        Name: name,
        Email: email,
        Phone: phone,
        DOB: new Date(dob),
      },
    });

    console.log("Data sent to CleverTap");
  });
```

Profile Data Push

An event listener is added to a button. When the button is clicked, the event listener retrieves the values of the name, email, phone, and date of birth fields, and pushes the profile data to CleverTap.

```
document
  .getElementById("profilePush")
  .addEventListener("click", function (event)
    var name = document.getElementById("name").value;
    var email = document.getElementById("email").value;
    var phone = document.getElementById("phone").value;
    var dob = document.getElementById("dob").value;
    var phoneRegex = /^\+\d+$/;
    if (!phoneRegex.test(phone)) {
```

```
      alert("Phone number should be formatted as +[country code][number]");
      return;
    }

    try {
      clevertap.profile.push({
        Site: {
          Name: name,
          Email: email,
          Phone: phone,
          DOB: new Date(dob),
        },
      });

      console.log("Profile data pushed to CleverTap");
    } catch (error) {
      console.error("Error pushing profile data to CleverTap:", error);
    }
  });
```

Push Notification Request

An event listener is added to a button. When the button is clicked, the event listener sends a push notification request to CleverTap.

```
document
  .getElementById("askForPush")
  .addEventListener("click", function (event) {
    try {
      // call the CleverTap function
      clevertap.notifications.push({
        titleText: "Would you like to receive Push Notifications?",
        bodyText:
          "We promise to only send you relevant content and give you
updates on your transactions",
        okButtonText: "Sign me up!",
        rejectButtonText: "No thanks",
        askAgainTimeInSeconds: 5,
```

```
        okButtonColor: "#f28046",
      });

      console.log("Push notification request sent to CleverTap");
    } catch (error) {
      console.error(
        "Error sending push notification request to CleverTap:",
        error
      );
    }
  });
```

Event Tracking

An event listener is added to a button. When the button is clicked, the event listener defines a set of event properties and pushes an event to CleverTap.

```
document
  .getElementById("eventButton")
  .addEventListener("click", function (event) {
    // define the event properties
    var eventProperties = {
      "Property 1": "Value 1", // string property
      "Property 2": 123, // integer property
      "Property 3": 45.67, // float property
      "Property 4": new Date(), // date-time property
    };

    try {

      clevertap.event.push("Event button clicked", eventProperties);

      console.log("Event pushed to CleverTap");
    } catch (error) {
      console.error("Error pushing event to CleverTap:", error);
    }
  });
```

This code provides a comprehensive example of how to integrate CleverTap into a web application, track user interactions, and send push notifications. It demonstrates the use of various CleverTap functionalities, including event tracking, user profile tracking, and push notifications. It also highlights the importance of correctly formatting data and handling errors. The code is well-structured and easy to understand, making it a valuable resource for anyone looking to integrate CleverTap into their web application.

Troubleshooting

Issue 1: Not Being Able to See the Event Push

If you're not being able to see the event push, it might be due to the server you're using. CleverTap's push event functionality requires a secure context. This means that your website needs to be served over HTTPS, or from localhost during development.

Solution

Try using a secure server, either a localhost for development or an HTTPS server for production. This is because push notifications require a secure context as they involve service workers, which have the ability to intercept network requests and modify responses. Due to these powerful capabilities, service workers are only available to secure origins (HTTPS sites, basically) to prevent misuse of the API.

Issue 2: Incorrect Import of Service Worker

Another common issue is the incorrect import of the service worker. Service workers are a crucial part of the push notification functionality, and if they're not correctly imported, it can lead to issues.

Solution

Ensure that the service worker is correctly imported. Check the path of the service worker file in your JavaScript code. It should be relative to the origin, not the app directory. Also, make sure that the service worker file is being served correctly by your server.

Issue 3: Adblocker Chrome Plugins Interfering with CleverTap SDK

If you have ad blocker plugins installed on your Chrome browser, they might interfere with the CleverTap SDK. This can result in errors when calling the CleverTap SDK inside the console.

Solution

To test if this is the issue, try using Chrome's Incognito mode. Incognito mode disables most plugins by default, including ad blockers. If the CleverTap SDK works correctly in Incognito mode, then the issue is likely due to an adblocker plugin. In this case, you can either disable the ad blocker plugin or add an exception for your website in the adblocker's settings.

Important Links

- **Github Website** - https://github.com/Ciriously/CleverTap_Test_WebSDK
- **Deployed Website** - https://clever-tap-test-web-sdk.vercel.app/

- **References** - https://developer.clevertap.com/docs/getting-started
- **Video Demo** -