Add event listeners in Prometheus

Using this method, we can only track the event in our local system by adding an extension.

Here's a step-by-step guide on how you can create a browser extension to monitor user interactions on the website:

1. **Create a Browser Extension**: Develop a browser extension using HTML, CSS, and JavaScript. This extension will inject code into the website's DOM to track user interactions.
2. **Inject Tracking Code**: Inject JavaScript code into the website's DOM using the browser extension. This code will track user interactions such as button clicks, page visits, scrolling, etc., and send this data to your backend server for further processing.
3. **Send Data to Backend Server**: Once the tracking data is collected, send it to your backend server using AJAX requests. The backend server will receive this data and perform the necessary operations like counting the number of people visiting projects, blogs, filling forms, etc.
4. **Process Data and Display in Grafana**: On the backend server, process the received data and store it in a database or any other suitable storage mechanism. You can then use Grafana to visualize this data by connecting Grafana to your database and creating dashboards to display the desired metrics.

Here's a basic example of how you can implement this approach:

**Manifest.json**

{

    "manifest\_version": 3,

    "name": "Website Interaction Tracker",

    "version": "1.0",

    "permissions": [

      "activeTab",

      "scripting"

    ],

    "content\_scripts": [

      {

        "matches": ["https://portfolio.aftabs.co/\*"],

        "js": ["content.js"]

      }

    ]

  }

**Content.js**

// Log that the script is running

console.log('Content script is running');

// Track button clicks

const submitButton = document.querySelector('button[type="submit"]');

if (submitButton) {

  submitButton.addEventListener('click', function() {

    sendDataToServer('form-submit');

    console.log('Form submit button clicked');

  });

} else {

  console.log('Submit button not found');

}

// Track link clicks

document.querySelectorAll('a').forEach(link => {

  link.addEventListener('click', function() {

    const url = this.getAttribute('href');

    if (url.includes("blogs")) {

      sendDataToServer('visit-blogs');

      console.log('Visited blogs');

    } else if (url.includes("projects")) {

      sendDataToServer('visit-projects');

      console.log('Visited projects');

    } else if (url.includes("resume")) {

      sendDataToServer('visit-resume');

      console.log('Visited resume');

    }

  });

});

// Track scrolling

window.addEventListener('scroll', function() {

  sendDataToServer('scrolling');

  console.log('Scrolled');

});

function sendDataToServer(interactionType) {

  const timestamp = new Date().toISOString();

  // Send data to backend server using AJAX request

  fetch('http://localhost:3005/track-interaction', {

    method: 'POST',

    headers: {

      'Content-Type': 'application/json'

    },

    body: JSON.stringify({ interactionType, timestamp })

  })

  .then(response => {

    console.log('Interaction tracked successfully:', interactionType, timestamp);

  })

  .catch(error => {

    console.error('Error tracking interaction:', error);

  });

}

**Server.js**

const express = require('express');

const bodyParser = require('body-parser');

const cors = require('cors');

const client = require('prom-client');

const app = express();

const PORT = 3005; // Ensure this matches your Prometheus target

// Enable CORS for all routes

app.use(cors());

// Create a Registry which registers the metrics

const register = new client.Registry();

// Create custom metrics

const formSubmitCounter = new client.Counter({

    name: 'form\_submits\_total',

    help: 'Total number of form submits',

});

const visitBlogsCounter = new client.Counter({

    name: 'visit\_blogs\_total',

    help: 'Total number of visits to blogs',

});

const visitProjectsCounter = new client.Counter({

    name: 'visit\_projects\_total',

    help: 'Total number of visits to projects',

});

const visitResumeCounter = new client.Counter({

    name: 'visit\_resume\_total',

    help: 'Total number of visits to resume',

});

const scrollingCounter = new client.Counter({

    name: 'scrolling\_total',

    help: 'Total number of scroll events',

});

// Register the custom metrics

register.registerMetric(formSubmitCounter);

register.registerMetric(visitBlogsCounter);

register.registerMetric(visitProjectsCounter);

register.registerMetric(visitResumeCounter);

register.registerMetric(scrollingCounter);

// Endpoint to receive data from extension

app.use(bodyParser.json());

app.post('/track-interaction', (req, res) => {

    const { interactionType, timestamp } = req.body;

    console.log('Received interaction:', interactionType, 'at', timestamp);

    // Increment the appropriate counter based on interaction type

    switch (interactionType) {

        case 'form-submit':

            formSubmitCounter.inc();

            console.log('formSubmitCounter incremented');

            break;

        case 'visit-blogs':

            visitBlogsCounter.inc();

            console.log('visitBlogsCounter incremented');

            break;

        case 'visit-projects':

            visitProjectsCounter.inc();

            console.log('visitProjectsCounter incremented');

            break;

        case 'visit-resume':

            visitResumeCounter.inc();

            console.log('visitResumeCounter incremented');

            break;

        case 'scrolling':

            scrollingCounter.inc();

            console.log('scrollingCounter incremented');

            break;

        default:

            console.log('Unknown interaction type');

            break;

    }

    res.sendStatus(200);

});

// Expose the metrics at the /metrics endpoint

app.get('/metrics', async (req, res) => {

    res.setHeader('Content-Type', register.contentType);

    res.send(await register.metrics());

});

app.listen(PORT, () => {

    console.log(`Server is running on port ${PORT}`);

});

**Packages need to be installed:**

npm i express

npm i body-parser

npm i prom-client

npm i cors

In promethus.yaml file we need to add the exporter we created:

global: scrape\_interval: 15s

scrape\_configs:

- job\_name: 'user\_interactions'

static\_configs: - targets: ['localhost:3005']

Then start promethus and start Grafana and try the metrices appearing in localhost:3005/metrics

**SETUP:**

 Install **and Load the Browser Extension**:

* Open your browser (e.g., Chrome).
* Go to chrome://extensions/.
* Enable **Developer mode**.
* Click **Load unpacked** and select the directory containing your manifest.json and content.js files.

 Open **the Target Website**:

* Go to https://portfolio.aftabs.co/.
* Open the browser console (F12) to see the logs.

 Interact **with the Website**:

* Click on the submit button.
* Click on blog links, project links, and resume links.
* Scroll the page.

 Verify **Backend Server Logs**:

* Check the terminal where the server is running to see if it logs the received interactions and increments the counters.