**Exception Handling, Logging and Analytics Tools**

|  |
| --- |
|  |
| **Document Version: 1.0** |
|  |



Table of Contents

[*Exception Handling* 3](#_Toc531622932)

[*Logging and Analytics* 4](#_Toc531622933)

[*Splunk* 4](#_Toc531622934)

[1.0 *Introduction* 4](#_Toc531622935)

[2.0 *Tracking mechanism used by Splunk* 4](#_Toc531622936)

[3.0 *How to Implement Splunk* 5](#_Toc531622937)

[*Google Analytics* 7](#_Toc531622938)

[4.0 *Introduction* 7](#_Toc531622939)

[5.0 *Usage* 7](#_Toc531622940)

[6.0 *Example of various tracking provided by Google Analytics* 7](#_Toc531622941)

[6.1 Page Tracking 7](#_Toc531622942)

[6.2 Event Tracking 8](#_Toc531622943)

[6.3 Social Interactions 8](#_Toc531622944)

[6.4 App / Screen Tracking 9](#_Toc531622945)

[6.5 User Timings 9](#_Toc531622946)

[6.6 Exception Tracking 10](#_Toc531622947)

[6.7 User Tracking 10](#_Toc531622948)

[7.0 *How to Implement Google Analytics* 10](#_Toc531622949)

[8.0 *Pros and Cons of Google Analytics* 11](#_Toc531622950)

[9.0 *Limits and Quotas (for free version)* 12](#_Toc531622951)

[*Mixpanel* 13](#_Toc531622952)

[10.0 *Introduction* 13](#_Toc531622953)

[11.0 *Usage* 13](#_Toc531622954)

[12.0 *Example of various tracking provided by Mixpanel* 13](#_Toc531622955)

[12.0 mixpanel.track 13](#_Toc531622956)

[12.1 mixpanel.track\_links 13](#_Toc531622957)

[12.2 mixpanel.track\_forms 13](#_Toc531622958)

[12.3 mixpanel.register\_once 14](#_Toc531622959)

[12.4 mixpanel.identify 14](#_Toc531622960)

[12.5 mixpanel.get\_distinct.id 14](#_Toc531622961)

[13.0 *How to Implement Mixpanel* 15](#_Toc531622962)

[14.0 *Pros and Cons of MixPanel* 16](#_Toc531622963)

[15.0 *Limits and Quotas (for free version)* 17](#_Toc531622964)

[*Recommended Tool* 18](#_Toc531622965)

[Google Analytics 18](#_Toc531622966)

# Exception Handling

Angular app

Web API

Google Analytics (Logging & Analytics framework)

Operation Request

Backend Processing

Exception

Encapsulate exception into JSON response

JSON Response

Log Events & Exceptions

Page displaying friendly error message

1. During a normal operation at Web API side, when an exception is encountered, the Try-Catch mechanism will capture the exception.
2. Custom Exception classes are used on the API side to wrap up the exception using the below JSON syntax.

{

ErrorCode:<Custom Defined>,

FriendlyMessage:<Friendly message to show on screen>,

ErrorType:<CustomError or SystemError>,

Stacktrace:<Stacktrace / Detailed Error>,

ExceptionMessage:<System generated Exception Message>

}

1. Web APIwill return JSON response to client side.
2. Based on the ErrorType and FriendlyMessage received on the response, the Angular app will either redirect the user to the Error Page showing the custom error message or if it is a business exception, the same will be shown on the screen as validation error or so.

For e.g. User attempts to login to UTICA portal.

He/she enters incorrect credentials, in that case Web API service will raise Custom Business exception i.e. “**Incorrect Credentials** **Entered**” and send JSON response to client side.

1. At front end, the Google Analytics will also be used for logging the JSON received by passing in all the JSON information into google analytics object to the analytics service.
2. Web API service will send different **ErrorType** like “**CustomError”** for **Custom Errors caught as Business Rule Exceptions** and “**SystemError”** for **System generated Errors**.

On Custom error, front end will receive same value for JSON field FriendlyMessage and ExceptionMessage. And for System Error, front end will receive actual System Exception message in ExceptionMessage and Friendly Message in field FriendlyMessage.

# Logging and Analytics

# Splunk

# Introduction

Splunk gives users real-time answers to meet customer expectations and business goals. One can use Splunk to connect machine data and gain insights into opportunities and risks for the business. It can help leverage artificial intelligence powered by machine learning for actionable and predictive insights.

# Tracking mechanism used by Splunk

**Javascript -**

var SplunkLogger = require("splunk-logging").Logger;

var config = {

token: "your-token-here"

};

var logger = new SplunkLogger(config);

// Payload to send to HTTP Event Collector.

var payload = {

message: {

temperature: "70F",

chickenCount: 500

},

severity: "info",

metadata: {

source: "chicken coop",

sourcetype: "httpevent",

index: "main",

host: "farm.local",

}

};

// The callback is only used if maxBatchCount=1, or batching thresholds have been exceeded.

logger.send(payload, function(err, resp, body) {

if (err) {

console.log("error:", err);

}

// If successful, body will be {text: 'Success', code: 0}

console.log("body", body);

});

For the config, the token is a mandatory property which needs to be generated at the Splunk portal and used here.

**.NET –**

**var** traceSource = **new** TraceSource("MyLogger");

traceSource.Switch.Level = SourceLevels.All;

traceSource.Listeners.Clear();

traceSource.Listeners.Add(**new** HttpEventCollectorTraceListener(

uri: **new** Uri("https://splunk-server:8088"),

token: "205A7CE0-24B6-44CD-9299-333E29BBBCF1");

traceSource.TraceEvent(TraceEventType.Information, "Hello world");

Various logging (severity) levels supported by Splunk are –

|  |
| --- |
|  |
| **Name** | **Type** | **Description** |
| DEBUG | string |  |
| INFO | string |  |
| WARN | string |  |
| ERROR | string |  |

Default value: INFO

# How to Implement Splunk

Splunk can be implemented both on the Angular (front end) side as well as on the Web API layer (back end).

**Prerequisites for Splunk logging for Javascript** –

1. Splunk Enterprise 6.3.0 or later, or Splunk Cloud.
2. Node.js – 0.10 or later
3. npm (latest version)
4. HTTP Event Collector

**To enable HTTP Event Collector, do the following** –

* For Splunk Enterprise, go to Settings > Data inputs > HTTP Event Collector, and then click Global Settings. Click the Enable button, and then click Save. For more information, see "[Enable HTTP Event Collector](http://docs.splunk.com/Documentation/Splunk/latest/Data/UsetheHTTPEventCollector#Enable_the_HTTP_Event_Collector)" in the Getting Data In manual.
* Create at least one HTTP Event Collector token: Go to Settings > Data inputs > HTTP Event Collector and click the New Token button. Proceed through the Add Data workflow until you've successfully created a token.

**Install Splunk logging for JavaScript, run the following from the command line –**

npm install --save splunk-logging

**Installing Splunk logging for .NET packages -**

### Using NuGet Package Manager

1. In Visual Studio, open the project whose activity you want to log to Splunk Enterprise.
2. On the Tools menu, point to Library Package Manager, and then click Manage NuGet Packages for Solution.
3. In the Manage NuGet Packages window, click Online from the list on the left, and then enter Splunk into the Search Online field in the upper-right corner. The Splunk logging for .NET library packages appear in the list.
4. Click the Install button for the logging library you want to install.
5. In the Select Projects window, select the checkboxes next to the projects in which you want to install the package, and then click OK.

The Package Manager adds the logging library you chose and its dependencies to your project. Because both main libraries require the common library, you'll see Splunk.Logging.Common in the list of references in the Solution Explorer, along with either Splunk.Logging.SLAB or Splunk.Logging.TraceListener, depending on which one you installed.

# Google Analytics

# Introduction

Google Analytics is a tool that helps you keep track of your User Activity. This includes how many users entered your website in a specific time-span, where those users came from, which pages they visited, etc. It has even expanded to how well your site’s speed is doing and how much social signal your site has garnered.

In short, Google Analytics is all about real data in your website. Data that you can translate into useful business intelligence and SEO strategy for your online campaigns.

# Usage

When you add either of these tracking snippets to your website, you send a pageview for each page your users visit. Google Analytics processes this data and can infer a great deal of information including:

* The total time a user spends on your site.
* The time a user spends on each page and in what order those pages were visited.
* What internal links were clicked (based on the URL of the next pageview).

In addition, the IP address, user agent string, and initial page inspection analytics.js does when creating a new tracker is used to determine things like the following:

* The geographic location of the user.
* What browser and operating system are being used.
* Screen size and whether Flash or Java is installed.
* The referring site.

# Example of various tracking provided by Google Analytics

## Page Tracking

Page tracking allows you to measure the number of views you had for a particular page on your website.

ga('send',{  
 hitType:'pageview',  
 page:location.pathname  
});

## Event Tracking

Events are user interactions with content that can be tracked independently from a web page or a screen load. Downloads, mobile ad clicks, gadgets, Flash elements, AJAX embedded elements, and video plays are all examples of actions you might want to track as Events.

ga('send', {

  hitType: 'event',

  eventCategory: 'Videos',

  eventAction: 'play',

  eventLabel: 'Fall Campaign'

});

## Social Interactions

We can use social interaction analytics to measure the number of times users click on social buttons embedded in webpages. For example, you might measure a Facebook "Like" or a Twitter "Tweet".

ga('send', {

  hitType: 'social',

  socialNetwork: 'Facebook',

  socialAction: 'like',

  socialTarget: 'http://abcd.com'

});

## App / Screen Tracking

Measuring screen views allows you to see which content is being viewed most by your users, and how are they are navigating between different pieces of content.

ga('send', 'screenview', {

  'appName': 'myAppName',

  'screenName': 'Home'

});

## User Timings

User timings allow developers to measure periods of time using the analytics.js library. This is particularly useful for developers to measure the latency, or time spent, making AJAX requests and loading web resources.

ga('send', {

  hitType: 'timing',

  timingCategory: 'JS Dependencies',

  timingVar: 'load',

  timingValue: 3549

});

## Exception Tracking

Exception tracking allows you to measure the number and type of crashes or errors that occur on your property.

ga('send', 'exception', {

   'exDescription': err.message,

   'exFatal': false

});

## User Tracking

Using User ID, we can unify the sessions that a single user is making using different devices, into one user in Google Analytics.

Not only does it make your user count more accurate, it will also help you understand how people *really* interact with your website. Imagine someone finding your website on a mobile device, then going to a desktop to finalize a purchase.

ga('set', 'userId', USER\_ID);

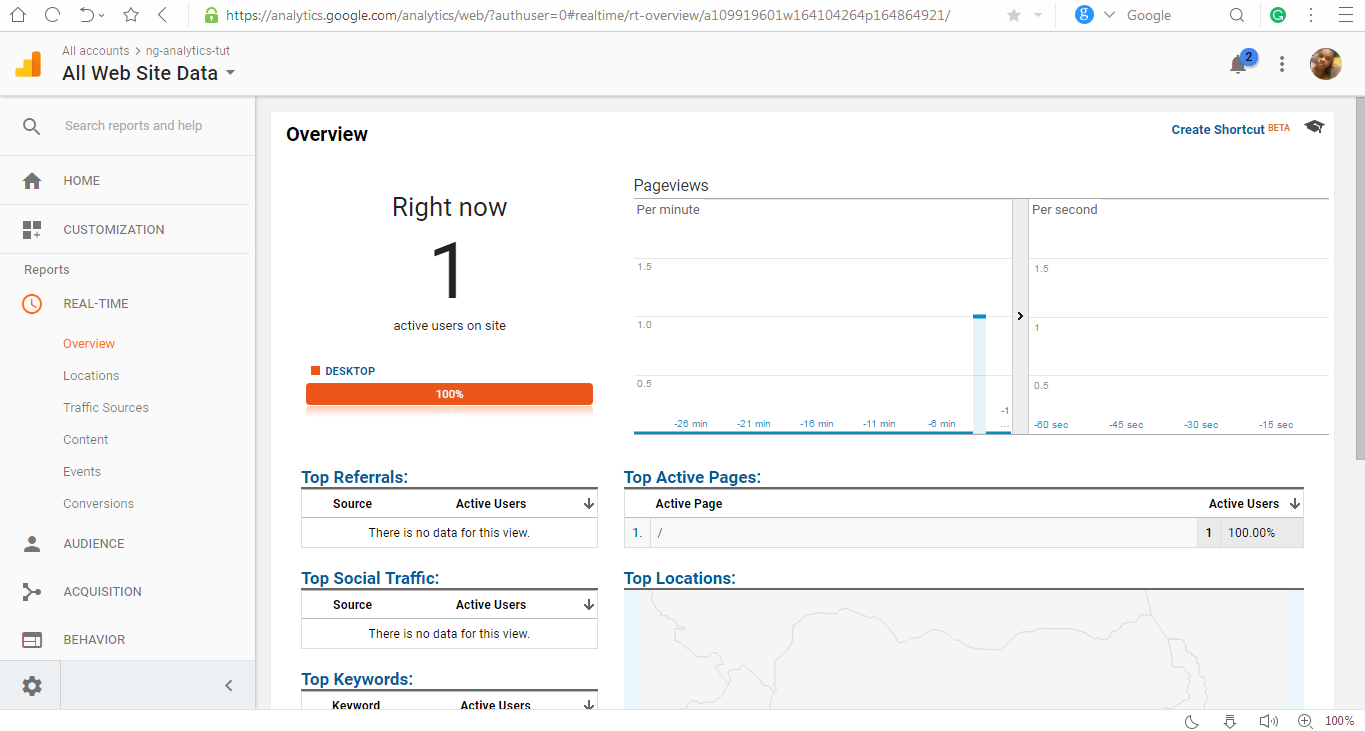
# How to Implement Google Analytics

* 1. First, we need to sign up and register our site in Google Analytics Website.
  2. Fill up all the require details and then click on “Get Tracking ID”.
  3. Copy your Tracking code and paste the analytics code below the *<app-root>* tag.

/\*\* index.html \*\*/  
...  
<app-root></app-root>  
<script>  
 (function(i, s, o, g, r, a, m) {  
 i['GoogleAnalyticsObject'] = r;  
 i[r] = i[r] || function() {  
 (i[r].q = i[r].q || []).push(arguments)  
 }, i[r].l = 1 \* new Date();  
 a = s.createElement(o),  
 m = s.getElementsByTagName(o)[0];  
 a.async = 1;  
 a.src = g;  
 m.parentNode.insertBefore(a, m)  
 })(window, document, 'script', '[https://www.google-analytics.com/analytics.js'](https://www.google-analytics.com/analytics.js%27), 'ga');

ga('create', 'UA-XXXXX-X', 'auto');// add your tracking ID here.  
ga('send', 'pageview');  
</script>  
...

* 1. In the Google Analytics Dashboard we can see something like this.



# Pros and Cons of Google Analytics

**Pros**

1. The free component of Google Analytics’ freemium based model allows you to utilize the basics of the application for free.
2. Google Analytics is a revolutionary tool business owner can utilize to figure out where the traffic is originating. In addition, it allows business owners to find out what keyword terms were used to be directed to the web page.
3. Google Analytics allows you to analyze your content performance.
4. The usage of Google Analytics is not restricted to your computer. You can use Google Analytics on mobile devices including Androids and iPhones.
5. It’s free and easy to install.

**Cons**

1. The premium component of Google Analytics’ freemium based model requires users to subscribe to the premium version if their traffic volume is high (one that exceeds the below given limits and quotas). The premium version costs $150,000/year (USD).
2. The usage of Google Analytics requires constant training because the user interface is constantly being updated. The changes are sometimes substantial as new features are being introduced and old features are either being combined or completely discarded. Users are required to keep up with Google Analytics’ constant updates.
3. All visits to a website must execute a JavaScript file. Visits that do not execute a JavaScript file do not get counted.

# Limits and Quotas (for free version)

The following limit applies to Web Property / Property / Tracking ID:

* **10 million** hits per **month** per **property**

If we go over this limit, the Google Analytics team might contact you and ask you upgrade to Analytics 360 or implement client sampling to reduce the amount of data being sent to Google Analytics.

The following limits apply to gtag.js, analytics.js, Android SDK, iOS SDK, and the Measurement Protocol:

* **200,000** hits per **user** per **day**
* **500** hits per **session**

If we go over either of these limits, additional hits will not be processed for that session or day, respectively. These limits apply to Analytics 360 as well.

The following limit applies to ga.js, mobile snippets, and any other legacy tracking library:

* **500** hits per **session**

If we go over this limit, additional hits will not be processed for that session. This limit applies to Analytics 360 as well.

# Mixpanel

# Introduction

**Mixpanel** is a business analytics service company. It tracks user interactions with web and mobile applications and provides tools for targeted communication with them. Its tool-set contains in-app A/B tests and user survey forms. Data collected is used to build custom reports and measure user engagement and retention.

# Usage

Mixpanel works with web applications, SaaS, but also supports mobile apps.

* **Mixpanel** is an event-centric platform where tracking is not automated and is based on events.
* **Mixpanel** is used to track events, create sales funnels and see trends.
* **Mixpanel** is more targeted than Google Analytics and the data they provide offers lots of possibilities.

# Example of various tracking provided by Mixpanel

## mixpanel.track

Track an event. This is the most important and frequently used Mixpanel function.

// track an event named 'Registered'

mixpanel.track('Registered', {'Gender': 'Male', 'Age': 21});

## mixpanel.track\_links

Track clicks on a set of document elements. Selector must be a valid query. Elements must exist on the page at the time track\_links is called.

// track click for link id #nav

mixpanel.track\_links('#nav', 'Clicked Nav Link');

## mixpanel.track\_forms

Track form submissions. Selector must be a valid query.

// track submission for form id 'register'

mixpanel.track\_forms('#register', 'Created Account');

## mixpanel.register\_once

Register a set of super properties only once. This will not overwrite previous super property values, unlike register().

// Register a super property for the first time only

mixpanel.register\_once({'First Login Date': new Date().toISOString()});

## mixpanel.identify

Identify a user with a unique ID instead of a Mixpanel randomly generated distinct\_id. If the method is never called, then unique visitors will be identified by a UUID generated the first time they visit the site.

## mixpanel.get\_distinct.id

Returns the current distinct id of the user. This is either the id automatically generated by the library or the id that has been passed by a call to identify().

// set distinct\_id after the mixpanel library has loaded

mixpanel.init('YOUR PROJECT TOKEN', {

loaded: function(mixpanel) {

distinct\_id = mixpanel.get\_distinct\_id();

}

});

# How to Implement Mixpanel

1. Install the Mixpanel Library (npm install --save mixpanel-browser)
2. Copy and paste the following code snippet inside the <head> and </head>tags to install the Mixpanel JavaScript Library. Note that you need to insert your project token in place of ”YOUR TOKEN" (do not delete the quotation marks) in the last line of the snippet.

<!-- start Mixpanel -->

<script type="text/javascript">

(function(e,a){if(!a.\_\_SV){var b=window;try{var c,l,i,j=b.location,g=j.hash;

c=function(a,b){return(l=a.match(RegExp(b+"=([^&]\*)")))?l[1]:null};g&&c(g,"state")&&(i=JSON.parse(decodeURIComponent(c(g,"state"))),

"mpeditor"===i.action&&(b.sessionStorage.setItem("\_mpcehash",g),history.replaceState(i.desiredHash||"",e.title,j.pathname+j.search)))}

catch(m){}vark,h;window.mixpanel=a;a.\_i=[];a.init=function(b,c,f){functione(b,a){varc=a.split(".");2==c.length&&(b=b[c[0]],a=c[1]);b[a]=function()

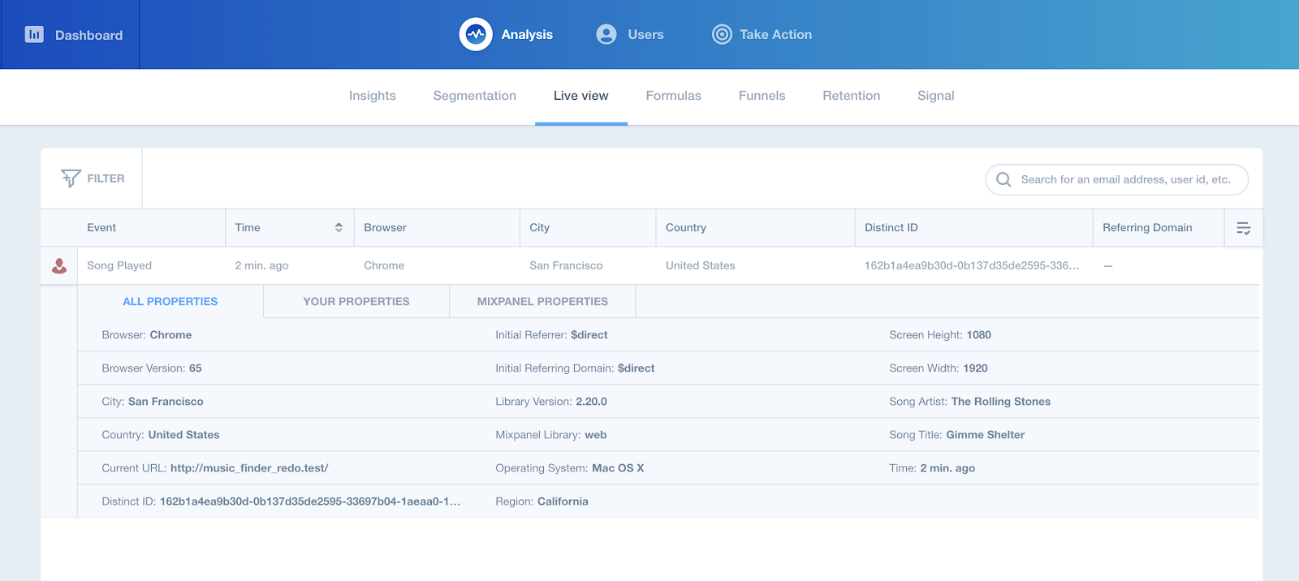
{b.push([a].concat(Array.prototype.slice.call(arguments,0)))}}vard=a;"undefined"!==typeoff?d=a[f]=[]:f="mixpanel";d.people=d.people||[];d.toString=

function(b)mixpanel.init("YOUR TOKEN");

</script>

<!-- end Mixpanel -->

1. Add the necessary tracking events to track each event.
2. In the MixPanel dashboard you will something like this.



# Pros and Cons of MixPanel

**Pros**

1. Mixpanel allows users to easily define events to track, ensuring a flexibility that promotes focused and meaningful analytics.
2. Mixpanel equips its users with tools that use statistical methodologies from modern data science, removing the need to write complicated code.
3. Mixpanel provides customer support to all Mixpanel users, regardless of plan type.
4. Mixpanel Messages make sending targeted messages easy and provides robust analysis on the recipients of those targeted messages.
5. Mixpanel offers an intuitive machine learning user interface that uses modeling to uncover concealed insights within a data set.

**Cons**

1. Too expensive to be used along with useful features like Google Tag Manager.
2. Slightly trickier to implement in comparison to google analytics.
3. Less flexibility in setting up user flows (able to map different paths/ decision points branching from the same flow.
4. Mixpanel charges based on data points which can encourage skimping on important data points.
5. Takes a while to get used to the interface.

# Limits and Quotas (for free version)

* **Event Properties**

There are technically no limits on the number of Event properties you can send to Mixpanel. However, the maximum number of properties you will be able to access in the Mixpanel UI drop-downs is 1500. Additionally, there is a limit of 255 properties per event so while you can have different events containing different properties, each event is limited to 255 properties. These limits include [Mixpanel default properties](https://help.mixpanel.com/hc/en-us/articles/115004613766-What-properties-do-Mixpanel-s-libraries-store-by-default-" \t "_blank) and [Mixpanel special/reserved Properties](https://help.mixpanel.com/hc/en-us/articles/115004602703-Special-or-reserved-properties" \t "_blank). It is important to note that if you find yourself above 1500 total event properties and/or 255 properties per event, it is likely you will want revisit the naming of your properties as discussed in this [article](https://help.mixpanel.com/hc/en-us/articles/115004545983-How-should-I-name-my-events-and-properties-).

* **People Properties**

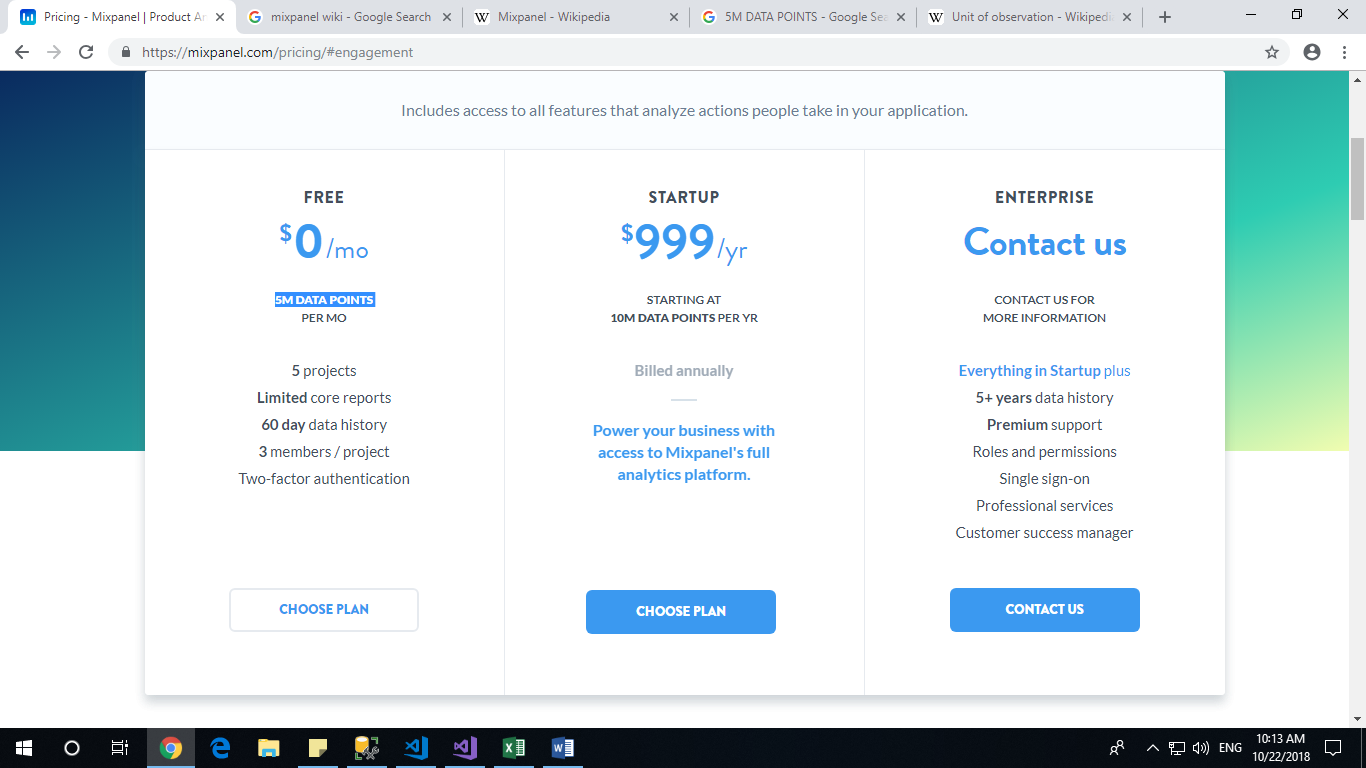
Mixpanel People Profiles can contain up to 255 properties each -- attempts to add more than 255 properties will fail. Additionally, only the 255 most common People Properties will be accessible in the Mixpanel UI drop-downs. As with Event Properties, these limits also include [Mixpanel default People Properties](https://help.mixpanel.com/hc/en-us/articles/115004613766-What-properties-do-Mixpanel-s-libraries-store-by-default-" \t "_blank).

While Event data (including Event Properties) are immutable, People Properties can be removed using the [$unset engage operation](https://mixpanel.com/help/reference/http#people-analytics-updates) if you find yourself close to the 255 per profile limit.

* **File Size character limit**

The following limits on field size apply when inputting fields for properties in Mixpanel:

* 255 characters for string properties
* 255 characters for each item in list properties



# Recommended Tool

## Google Analytics

Mixpanel is too expensive to be used along with useful features like Google Tag Manager and from implementation perspective, it is slightly trickier to implement in comparison to Google Analytics.

Also, considering the free version limits and quotas of Google analytics and looking at the volume of data and logs that would be processed, **Google analytics is the recommended tool to use**.