TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Mikk Raba 179955

OPEN SOURCE LIBRARY FOR COLLECTION OF USER INTERACTION DATA ON WEB APPLICATIONS

Bachelor's thesis

Supervisor: Andres Käver
BSc

Tallinn 2020

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Mikk Raba 179955

AVATUD LÄHTEKOODIGA TEEK KASUTAJA TEGEVUSINFO KOGUMISEKS VEEBIRAKENDUSTEL

bakalaureusetöö

Juhendaja: Andres Käver
BSc

Tallinn 2020

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Mikk Raba

18.05.2020

Abstract

Standard realisation of modern web analytics tools is usage of JavaScript libraries on client side. Implementation of tracking is done by utilising global functions provided by those libraries. These functions initiate data sending to collection servers once they are called on user chosen events and provided with input data. This process is called "tagging" and the snippets of code added to website's source "tags". Changes of configuration or requirements to set tags may change daily according to platform development. Management of configuration and tags are responsibilities of IT development teams.

Common solution to minimise the dependency on IT development teams is opting to use tag manager systems. Tag manager systems provide a graphical user interface (GUI) that allows the user to configure on which events and conditions behavioural data is collected. These systems generate necessary code according to user set settings. The generated script is then loaded by a small tag manager's script asynchronously on page load from third party servers and the tags are "injected" into website. This process makes tag managers inherently a security risk because they inject code into website from another resource. In addition, business stakeholders are usually provided with access to these systems in order for them to manage tags by themselves. In this way, because tag managers generated code for user, they place code related responsibility on business side which is a conflict of interests and responsibilities.

Furthermore, companies can have more than one analytics tool that needs separate tagging. Configuration and maintenance in essence, no matter if tagging directly in code or using a tag manager system, is a manual process that is time consuming. At the same time, these tools have similar data requirements as they are similarly designed to provide information about user behaviour on the web.

This thesis aims to solve this problem by providing a library that collects visitor web interaction data once in a structured manner and allows to forward this data to one or multiple tools.

Annotatsioon

Avatud lähtekoodiga teek kasutaja tegevusinfo kogumiseks veebirakendustel

Tänapäeval turul olevad veebianalüütika tööriistad on tavapraktikas rakendatud kliendi brauseris JavaScript teekidena. Andmete kogumine on realiseeritud nende teekide serveeritud globaalsete funktsioonide väljakutsumisega õigetel sündmustel ja sisendiga veebilehe koodis. Seda protsessi nimetatakse *tagging'*uks ning selle tulemusel lisatud koodilõike *tag'*ideks. Nõuded nendele *tag'*idele võivad muutuda igapäevaselt vastavalt rakenduses tehtud muudatustele ja haldusvastutus langeb IT arendusmeeskondadele.

Levinud lahendusena on IT arendusressurssist sõltuvuse vähendamiseks kasutusel *tag'*ide haldussüsteemid (inglise keeles *tag-manager systems).* Need haldussüsteemid on graafilise kasutajaliidesega tööriistad, mille abil saab kasutaja seadistada, millistel sündmustel, sisendinfoga ja tingimustel peab tööriist andmeid veebilehelt koguma. Need süsteemid genereerivad seadistuse alusel koodi, mis laetakse lehe laadimisel asünkroonselt ühe kontrollskripti poolt kolmanda osapoole serverist. Seda protsessi nimetatakse ka koodi süstimiseks veebilehele. Haldussüsteemide üks eesmärk on langetada tehnilist barjääri *tag*'ide seadistamisel ning praktikas langeb nende süsteemidega töö äriliste üksuste valdusesse. Selle tulemusel tekib vastutuse konflikt, sest IT lahendused, mis tegelevad platvormi funktsionaalsusega koodi tasemel ei tohiks olla äriliste üksuste vastutus. Teisalt toovad haldussüsteemid endaga kaasa turvariske, sest koodi laadimisel kolmanda osapoole serverist puudub ka kontroll selle koodi üle.

Ettevõtetel võib esineda mitu analüütika tööriista ja mitu veebilehte. Nende seadistamine andmete kogumiseks, kas otse koodis või läbi haldussüsteemi, on manuaalne töö. Sellest tingituna ei kasutata tihti tööriistade maksimaalset potentsiaalil. Teisalt on analüütikatööriistade eesmärk sarnane, see tähendab koguda andmeid kasutaja tegevuste kohta ning tihti vajavad need tööriistad sisendina sarnaseid või identseid andmeid.

Käesolev töö püüab lahendada andmete kogumise duplitseerimist ja tööriistade seadistamist. Selleks pakutakse välja lahendus andmete ühekordseks kogumiseks struktureeritud kujul, mis võimaldaks saata neid andmeid edasi mitmele tööriistale ning mis vähendaks vajadust *tag'*ide lisamiseks

Töö annab ülevaate ajaloost, tänastest praktikatest ja lahendustest turul. Seejärel analüüsitakse turul olevate analüütikatööriistade toimimismehhanisme, nende erinevusi, puudujääke ning ettevõtete vajadusi. Analüüsi tulemusena pakutakse välja lahendus JavaScript teegi kujul ning realiseeritakse antud lahendus.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 71 leheküljel, kuute peatükki, 27 joonist ja ühte tabelit.

List of abbreviations and terms

AaaS	Analytics as a Service
AJAX	Asynchronous JavaScript and XML
CSS	Cascading Style Sheets
DOM	Document Object Model
ES	EcmaScript
GUI	Graphical User Interface
HTML	Hypertext Markup Language
JS	JavaScript
PWA	Progressive Web Application
SaaS	Software as a Service
SDK	Software Development Kit
SPA	Single Page Application
UI	User Interface
URL	Uniform Resource Locator
UX	User Experience
WWW	World Wide Web

Table of contents

List of figures

List of tables

Introduction

Digital analytics and web interaction data collection has become a priority for many companies in the recent years. With exponentially accelerating digitalisation and physical presence losing its relevance companies are in a race to provide services on digital channels. Simultaneously, customers have adapted and are expecting services to be available in digital channels. This brings a shift in the environment companies are set as there are no borders or distance in the internet and location does not necessarily bring any advantage. Instead, companies are forced to compete in providing best user experience on digital channel.

Good user experience is a result of understanding visitor needs. In order to have this understanding they need to collect customer interaction data for analysis. Nowadays, market is filled with hundreds of web analytics tools which aim to help companies make data driven decisions. Companies are investing large amounts of money into them and many of such tools are provided according to a software as a service (SaaS) or analytics as a service (AaaS) model.

Standard implementation of modern web analytics tools is done on the client-side using JavaScript libraries. These libraries provide global functions which generate requests to analytics collection endpoints. These functions are called at certain user events that the site's owner is interested with contextual input data. Code snippets responsible for this action are called "tags" and the process of adding tags is called "tagging".

The implementation of tags and configuration is meant to be maintained by IT development team. Maintenance of tags can scale quickly in workload and can be left competing with other developments in the backlog. In order to eliminate or lessen the dependency on IT development resource, a tag manager solution is often used. A tag manager system is a graphical user interface (GUI) that allows to configure the model, conditions and events on which data is collected. After configuration, tag manager system generates necessary code for tracking. This generated code is downloaded asynchronously on page load by tag manager snippet in the website’s head section. This process is also called “injection” of code.

Although tag managers shift management of tags to a GUI and lower first barrier of technical knowledge, they also have downsides. Tag managers are inherently a security risk because they allow injecting custom HTML and JS snippets and in standard practise serve generated code from third-party servers over which the user does not have control over. In addition, tag managers are usually left in full control of business side which raises a conflict of responsibility. Business units should not be responsible of site’s functionality especially if it consists of code management.

Many companies have multiple analytics tools on their website. The process of tagging, no matter if directly in code or using a tag manager, is in essence a manual process. Companies may also have multiple websites

All this can create a situation where return on investment is negative due to subpar or lagging tool implementation. At the same time analytics tools all aim to give understanding of the same user behavioural events and require similar input data gathered from those events.

1 Objectives and scope

The objective of this thesis is to analyse available web analytics tools, their implementation practises and requirements by business stakeholders and IT, in order to propose and develop a JavaScript library that would simplify implementation of web analytics related data collection, particularly user behavioural event tracking, and limit the need for tagging or extra configuration. The library would aim to allow website owners to collect information on events in a certain universal base structure with one script, allow same implementation on multiple websites in a quick manner, manage data collection with full governance on their own platform and potentially collect more data with default implementation.

This thesis strictly looks and provides a solution for web analytics. With rapid growth of mobile applications, sometimes the term “digital analytics” is used instead of “web analytics” and sometimes they are used interchangeably.

2 Background

In this section, in the beginning a brief overview of history is provided. Next, a more in-depth overview of surrounding environment is provided, covering field expectations, legal trends, solutions on the market and technical implementation practises. Lastly, short insight to alternatives is given.

2.1 History

With the spread of internet and progress of companies building online presences, an issue presented immediately – there was no understanding of who was the visiting customer or how many customers visited the website. As first remedy for this problem, in the beginning of World Wide Web (WWW), analysis of server log files that stored web transaction data were used to obtain some understanding. This approach allowed basic overview of website views and visits. Visits had also an alternative name "session" [1].

The approach of using log files had its limitations. With arrival of first search engines, web engine spiders started crawling websites making it difficult to understand the number of unique visitors. In addition to spiders, internet service providers started implementing dynamically assigned IP addresses adding to this problem. In order to solve this, a new technique of data collection called "page tagging" emerged. The new method was often complimented by a counter on the website presenting the number of visitors. Page tag was essentially a hidden pixel image element on the website, that once downloaded, stored the information of visit and additional parameters assigned to query of the image request. In addition, with the image downloaded, a cookie was also set to user's browser which allowed measuring user retention. [1] [2]

While this helped mitigate problems related to visitor uniqueness and was an important step further, such measurement was still limited because it did not allow to understand customer behaviour on a deeper level. There was no information on user interactional events such as user clicks and other mouse events [1]. Page tagging evolved quickly and with JS more information about customer interactions was added to the requests. JS libraries started emerging that instead of loading an image placed already to the HTML, generated a request to certain address passing more information and allowing more customizability.

First modern web analytics tool is considered to be made by a United States company WebTrends [1]. In 2015, Google started offering their web analytics platform *Google Analytics* after acquiring Urchin [3]. *Google Analytics* holds today a market share of 84 percent according to W3Techs surveys [4].

The original pixel placement and loading is still in use as a fallback in situations where no scripts are allowed or possible to use. For example, it is used with emails because the level of functionality of mail clients may differ dramatically [5] [6].

2.2 Current situation

In the past two decades, web and mobile related analytics has gone through extensive change. In 2016, data and analytics market size was 160 billion US dollars expecting to reach 220 billion in 2020. [7] Ravi et al. analyse in their review "Analytics in/for cloud-an interdependence: A review" that cloud computing has lowered the cost of data storage and computing resources and it is expected for analytics related cloud services to increase in sales in the forthcoming years. Many companies opt for on-demand Analytics services and "Analytics as a Service" has become a buzzword [8]. Consequently, almost all popular web analytics tools on the market today are cloud based. Cloud based solutions have multiple benefits. They can be easily scaled, allow quicker time to market, need no infrastructure, level the possibilities of big and small companies and demand less designated people from the company [9]. On the other hand, it must be pointed out that cloud solutions have their security challenges. Data breaches, compromised authentication, system vulnerabilities and Denial-of-Service (DoS) attacks are a few of many risks that a company may face when using cloud services [10]. On-premise solutions can offer viable options for companies that are more concerned regarding the security and location of their sensitive data. There are a few vendors that do offer on-premise solutions, for example WebTrends and Matomo [11].

Many vendors offer their services using freemium pricing model where limited package is offered for free. Freemium pricing model is also used by most popular *Google Analytics* but does include others, for example *Amplitude* and *Mixpanel* [12] [13] [14]. Using a tracking script search extension, for example Ghostery, on any of modern browsers and moving around the web, it can be seen that while *Google Analaytics* is almost a *de facto* standard, other tools are also often present.

2.2.1 The influence of growing regulation

In the past five years, data collection for analytics and digital marketing has undergone heavy regulation. Largest impact has been General Data Protection Regulation (GPDR) that went into effect on the 25th of May 2018 in the European Union. GDPR aims to regulate personal data usage and give customer more control over their personal data. In order to do this, GDPR forces companies to write privacy policies in clear and forward language that state how the data is used and demand for customer consent to use their data. This consent needs to be clear and silence is not considered an agreement. Furthermore, companies need to have clearly defined purposes and documentation for data processing. Most importantly, GPDR gives user rights to move, access, ask a copy and ask to delete the data. Every popular company has visitors from Europe on their website and is obligated to comply with the law in the same manner as companies residing in European Union [15].

Other regions around the world are also following with similar acts. In the United States, California Consumer Privacy Act (CCPA) took effect in state of California 1st of January 2020 and first part of Stop Hacks and Improve Electronic Data Security Act (SHIELD Act) has gone into effect in the New York State [16]. According to international law firm Hogan Lovells' 2019 "Asia Pacific Data Protection and Cyber Security Guide", China, India, Australia and other countries in the Asia-Pacific region already have laws implemented, are reviewing existing legislation or creating it inspired by the European GDRP directive [17].

The implementation of GDPR has also raised knowledge of data usage in the society which forces the businesses to consider reputational aspects in case of not being compliant in addition to possible fines. According to European Commission info sheet, by May 2019, 67% of Europeans had heard of GDRP and the knowledge of the existence of a public authority responsible for protecting user rights about personal data had grown 20 percentage points since 2015. Fines for not being compliant with GDPR have also been enacted. Biggest of them has been France fining Google with 50 million euros due to lack of gathering consent on advertisements [18].

In addition to GDPR, European Commission has proposed an update for ePrivacy Regulation. If GDPR protects personal data then ePrivacy Regulation aims to give users rights for privacy and confidentiality in their electronic communication when using their devices [19] [20].

Considering the ongoing process of ePrivacy Regulation, higher focus on data usage in the society and to the misuse of data by large international companies, the trend of regulation and expectations for levels of data security in the society is continuing to grow rapidly [21].

2.3 Analytics tools on the market

The number of different analytics tools has grown into hundreds [4]. Customer web and mobile platform interaction data has vast number of applications and this has resulted in a saturated market where there are many tools that differ in minor aspects. Scott Brinker, a marketing technologist, brings out in his blog that according to Netskope April 2017 study, marketing teams are using 91 cloud services on average [22]. Applications to data include but are not limited to, marketing analysis to understand the results of offering related efforts, user experience analysis, predictive behavioural analysis, personalisation of content and conversion analysis [1].

According to a research and advisory company Gartner's review, *SAS*, *Adobe Analytics* and *Google Analytics* are market leaders in web analytics. These tools are stand-alone and provide full solution incorporating data collection methods, storage and visualization features [23]. In addition, there are smaller companies on the market that also offer behavioural analytics and sometimes include a niche feature. As an example, *Mixpanel* is an event-based analytics service where the user must define the events fully by themselves. It offers extensive behavioural analytics visualization, segmenting, user flow funnelling and retention analysis but they also include user messaging feature that can be triggered based on result in the tool. Another tool, *KissMetrics*, aims to connect every action on the website or other platforms to single person. This behaviour is called "user profiling" and other tools also do offer similar features. For example, *Adobe Analytics* has "Cross-Device Analytics" feature, however they differ from *KissMetrics* as they do not place such behaviour as core principle [14] [24] [25].

Furthermore, there are tools that directly address single problems. Such tools address customer funnelling, heatmapping, screen recording, search engine analysis, A/B testing and many more.

2.3.1 Technical realisation and standard implementation

Standard implementation of web analytics tools is a JS library that is loaded asynchronously on page load. A tag is added to the header or footer of the website which includes a self-invoking function that begins the load process. For demonstration of such tag, implementation of *Google Analytics* core library analytics.js is included below (Figure 1). The 'UA-XXXX-Y' on line 12 references unique instance id which is available in the admin interface of the tool [26] [27] [28] [29].

```
1.    <!-- Google Analytics -->
2.    <script>
3.    (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r
4.    ]=i[r]||function(){
5.    (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new
6.    Date();a=s.createElement(o),
7.    m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parent
8.    Node.insertBefore(a,m)
9.    })(window,document,'script','https://www.google-
10.   analytics.com/analytics.js','ga');
11.
12.   ga('create', 'UA-XXXXX-Y', 'auto');
13.   ga('send', 'pageview');
14.   </script>
15.   <!-- End Google Analytics -->
```

Figure 1. Google Analytics analytics.js implementation tag.

These libraries include certain global functions which are called at pre-defined or user-defined events and that in turn generate requests to be sent to collection server. An example of such function can be found in above *Google Analytics* tag code sample. On line 13 a global function `ga` is called with "send" method description and "pageview" hit type description. This creates a request to collection servers [26] [27] [28] [29].

User also has an option to pass additional information to those functions matching a default or user-defined model. Below is an example of previous pageview hit type with additional page information.

```
ga('send', {
    'hitType': 'pageview',
    'page': '/home'
});
```

Figure 2. Sample of passing additional information to global analytics functions. In this example, the user has opted to send parameter "page" with value "/home".

Adobe Analytics is using instead of a global function a global variable `s` that has a method `t` for pageviews and `tl` method for other events. The `s` object is read directly by those functions once triggered. “AppMeasurement.js” is *Adobe Analytics’* core library which is downloaded by admin from the user interface.

```
<!-- Adobe Analytics -->
<!-- AppMeasurement.js configured with report suite id -->
<script src="AppMeasurement.js"></script>
<script>
s.pageName = '/home';
s.eVar1 = "eng";
s.events = "event1";
s.t();
</script>
```

Figure 3. Example of passing additional page information with pageview event with Adobe Analytics.

Traditional stand-alone full solution analytics tools collect certain data points without extra configuration. Standard out of the box collected information includes time spent on the site, URL of the website, browser and operating system information, referring site, screen size, geographical location of the user and more [26] [28].

To simplify data collection, modern web analytics implementation best practise advises to use a data layer on the website. Data layer is web analytics field specific semantical expression for a JS object that is placed usually in the `<head>` section of the website and that includes data variables which are collected. Server populates page and customer related values dynamically on page load. Other events modify data layer according to changes and customer interactions. For example, when a campaign visual is presented, the id of a the visual may be stored in the data layer and every consequent request would include this information to the request. Data layer can be also seen as analytical state management, especially in case of SPAs. It is meant to be as a singular or main source of information for analytics related methods. Sometimes a method does not expect input variables but instead is written to look for specific information from the data layer object. Data layer is often mapped one to one on the tag manager's preferences for easier system configuration. A minimalistic data layer example is demonstrated below.

```
<script>
  var _dataLayer = {
    user: {
      name: "User",
    },
    page: {
      path: "/home",
      section: "main"
    },
    campaign: {
      seen: []
    }
  }
</script>
```

Figure 4. Example of a minimalistic data layer in common analytics implementation.

2.3.2 Implementation using tag manager systems

Some vendors have additional tag manager systems available to use as middle layer between analytics tool or other vendor offered services [28] [30] [31]. Most popularly known is *Google Tag Manager* and company Tealium offers a *Tealium IQ*, a popular tag manager for enterprises. Tag manager is a system with a graphical user interface (GUI) that allows the user to configure the data model and conditions on which behavioural data is collected and sent to collection servers. Tag manager systems generate a script which matches user preferences. This configuration script is loaded asynchronously by a tag manager initialization tag that is implemented in a similar manner as regular libraries explained above. For visualization, *Google Tag Manager*'s tag is hereby demonstrated on Figure 5. The tag always downloads latest published configuration script.

```
<!-- Google Tag Manager -->
<script>(function(w,d,s,l,i){w[l]=w[l]||[];w[l].push({'gt
m.start':new Date().getTime(),event:'gtm.js'});var
f=d.getElementsByTagName(s)[0],j=d.createElement(s),dl=l!
='dataLayer'?'&l='+l:'';j.async=true;j.src='https://www.g
oogletagmanager.com/gtm.js?id='+i+dl;f.parentNode.insertB
efore(j,f);
  })(window,document,'script','dataLayer','GTM-XXXX');
</script>
<!-- End Google Tag Manager -->
```
Figure 5. Google Tag Manger's implementation tag.

Among configuration, the tag manager system includes tool's JS library according to version necessary. Tag managers differ from regular library usage from the aspect that they aim to remove dependency of development team and opt out triggering functions directly in website's codebase essentially eliminating the need of tagging in the code. The user sets specific events in the GUI determining when specific data is collected.

In addition to eliminating the dependency on development team, tag managers aim to be more than a simplification for analytics implementation. There are additional integration possibilities for other services by the tool's vendor. For example, *Google Tag Manager* allows to integrate similarly to analytics *Google Ads* conversion tracking. *Adobe Launch*, Adobe's tag manager allows integration of *Adobe Target*, an A/B testing tool by Adobe. Integrations can be also made with third-party services whose vendors have made it available and provided integration modules. Lastly, tag managers allow injection of custom HTML and JS snippets. From business side, tag managers allow skip potentially long release cycles and allow quicker change in tags [31] [30].

2.3.3 Server-side solutions

It was stated above that standard implementation of analytics tools is realised on the client-side. This does not limit the usage of data collection endpoints on the server side and some vendors provide specific software development kits (SDK) supporting languages such as PHP, Java and more. In addition, the prominence of package managers allows third parties to quickly share custom built solutions in case the vendor has not provided one [32] [27].

Server-side solutions provide an option in case data is only available on the server side. Another reason for server-side implementation can be data sensitivity. A company may wish to send information to collection servers but not make it visible on the client side. Furthermore, client-side tracking can be blocked by user thus it may be necessary to send business critical events server side to ensures the quality of the data.

On the other hand, it has to pointed out that server-side implementation may require more development resources. Most importantly, server-side implementation does not have access to behavioural events that are accessible on the client-side. Such events are for example click, scroll, mouseover, touch and many more. Server-side implementation can possibly hinder data collection and be a contradicting solution for single page applications (SPA) which update their content dynamically utilising AJAX and load a single HTML page. SPA model aims to have less unnecessary interaction with the server and at the same time support high level interactivity in fluid manner. Server-side implementation can also be lacking coverage in case of progressive web applications (PWA) that bring new capabilities such as push notifications, offline mode et cetera [33] [34] [35].

2.3.4 Alternatives

Analytics tool *Heap* aims to remove the dependency of tagging. The tools core system is built around "autocapture" functionality. This functionality collects data on every click, pageview, scroll and other events without manual tagging or tag manager configuration on the website. *Heap* allows quick initial installation but does need larger configuration on the tool's user interface [36] [37].

There are also simplified solutions on the market. Facebook offers their analytics pixel for the web and provides some analytical overviews focused on conversion result related to their platforms. They do include overview of pages visited, device splits et cetera that designated analytics tools provide [38] [39]. In addition, there are tools that rely on external databases of website usage and indexing capabilities. Such tools are generally focused on search engine optimization (SEO) but often provide some input to website usage. For example, one such tool is *SEMRush* [40].

On another side, there are data streaming tools which provide capable data collection functionality however provide no or limited analytics and visualization layer. Data streaming means sending data continuously in small pieces as the data is being created instead of large batches. Data streaming tools may often be suitable for large enterprises with higher level of analytical maturity, who aim to collect big data and have real-time analytics requirements. These tools include additional options in their data processing pipeline opposed to traditional stand-alone tools like *Google Analytics* or *Adobe Analytics*. These options include but are not limited to full data access, data reprocessing, data enrichment, raw data and possibility to choose storage location. Examples of such type of tools are *Segment* and *Snowplow.* These tools may be complimented by stream-processing tools such as *Apache Kafka* and data visualization tools like *Tableau* or *QlikSense* [41] [42]. Data streaming is very versatile and allows extensive data modelling however requires high level of competence in the organization. Uthayasankar Sivarajah et al. discuss in their study "Critical analysis of Big Data challenges and analytical methods" that processing vast amounts of data remains a challenge. They also point out that storage of large volumes of data is a challenge itself. Furthermore, complexity of handling data variety, data inconsistency, velocity of data inflow and variability of sources cannot be underestimated. Small and medium sized companies do not have the resource to implement or utilize data at such level nor have the number of platforms at which making such an investment would be rational. Lastly, data visualization and modelling require another step of investment while traditional analytics tools offer easy to use user interface with pre-defined reporting templates and need no special integration to data warehouse [43].

3 Analysis

In this chapter, similarities and differences of analytics tools are viewed in more detail. Drawbacks of current implementation methods are discussed from business, IT and analytics perspective. This analysis is used as input for deriving requirements for making a proposal of a solution. Lastly, technical challenges of proposed solution are discussed, and an overview of similar existing solutions is given.

Scope of this analysis is limited to analytics tools that provide full user behavioural tracking coverage and analytical views unless stated otherwise.

3.1 Technical similarities and differences of tools available

Stand-alone analytics tools mainly rely on client-side implementation utilising designated JS libraries that provide global variables and functions necessary for sending data to collection servers. The prevalent implementation of analytics tools on the client-side is due to getting access to user behavioural events only available on the client-side. Tools opt for usage of plain JS without additional libraries [26] [27] [28] [29].

As best practise, documentation of most analytics tools advises to implement a data layer object on the page that acts as single source for general platform, user and state related information [28] [29]. *Mixpanel* offers a different approach of registering super properties that are automatically assigned with every consequent "track" method call [27]. As standard, all parameters related to certain hit must be present in the request. There is no caching and merging functionalities available on the tools' side.

Traditionally analytics tools include „pageview" as a primary event and allow users to define additional custom events. Some tools, for example *Mixpanel* and *Amplitude,* require user to define all events. All analytics tools allow passing additional data with calls to collection server. This data is required to match pre-defined or user-defined data model. Tools may have some reserved property names, however it is usually possible to overwrite values of these properties if necessary [28] [32]. For demonstration of passing additional contextual information, “Apply” button click event tracking is implemented with different tools.

```
<script>
// Google Analytics call to server
ga('send', {
  hitType: 'event',
  eventCategory: 'CTA Click',
  eventAction: 'click',
  eventLabel: 'Apply'
});

// Adobe Analytics
s.linkTrackVars = "eVar2, events";
s.eVar2 = "Apply"; // eVar2 defined as "label" dimension
in Adobe Analytics UI
s.events = "event2"; // event2 named to "CTA click" metric
in Adobe Analytics UI
s.tl();

// Mixpanel
mixpanel.track(
    'CTA Click',
    {'label': 'Apply'}
);

// Amplitude
amplitude.getInstance().logEvent('cta_click', {
    'label': 'Apply'
});
</script>
```

Figure 6. Demonstration of click event tracking with different analytics tools.

Except for tool *Heap* that tries to collect data using tool’s core auto capture feature and not use tagging [37], collection of data is implemented by using globally available library methods and implementing tags on the website. These tags trigger data sending at appropriate times. One website may have hundreds of such tags and it is website owner’s responsibility to define needed events and see through the implementation of tags. It is important to point out that there may exist other solutions that do not need tagging of which the author is unaware of.

It is also website owner’s responsibility to provide additional contextual info for those tags. For example, in a case where website’s owner wishes to track video play button and also include contextual information of video title, play time and other information, website's owner must implement methods to provide that information and send that as input according to requirements of particular tool.

Tools also differ in their number of variables collected by default. Traditional stand-alone tools like *Google Analytics* and *Adobe Analytics* have an extensive list of data points that are collected automatically. Such variables usually include URL, referrer, marketing query parameters, system and browser information [28] [26] [29].

Some tools, for example *Amplitude*, allow batching multiple events and provide separate endpoints for those actions. In this case, *Amplitude* expects a manual post request where the payload follows certain structure [29].

Leading web analytics tools support usage of tag manager systems which allow tagging a web page using a graphical user interface. Most popular tag manager is *Google Tag Manager*. Adobe provides their own tag manager named *Adobe Launch*. A user defines their variables, triggers and tags in the GUI instead of adding code snippets to websites source. Tag managers support vendor specific implementation but because they allow usage of custom HTML and JS snippets, they are often times also used for handling other tools related tags.

3.2 Drawbacks of current analytics tools implementation methods

In this chapter, an analysis is conducted looking at the requirements and issues related to solutions and practises used today from the views of business stakeholders, IT development and analytics.

3.2.1 Business perspective

The number of stakeholders who depend on accurate data collection varies heavily depending on a company size. Main and primary stakeholders are often marketing and sales teams who wish to have good understanding regarding offering and sales related performance. Stakeholders may also include teams responsible of user experience (UX), design and development. User experience and design responsible teams require knowledge on which components are effective, well noticed and clicked on the website, or which device models are most popular. Device model requirements interest also development teams when making decisions on development frameworks where the understanding of technical stack of website's userbase is important because decisions based on it may set course of development for years.

High quality data collection is the basis for making data driven decisions. When waterfall methodologies in software development were prevalent, business side needed to provide requirements very long in advance. Analytical questions in their essence cannot be sometimes foreseen and possibly change quickly in time. In 2001, agile development framework manifesto was written [44] and according to 13th annual State of Agile Report by CollabNet, in 2019, 97% of organizations are practising agile development frameworks [45]. Agile development supports proposing hypothesis, development according to the hypothesis, measuring and learning from mistakes in a short time span. This allows to adjust quicker to change of requirements and increase efficiency. From one side agile development frameworks support learning and analysing results. On the other hand, agile development focuses on development of features that bring the customer immediate benefits. This means analytics can often be left competing for development resource. In addition, while time frame has been shortened, release cycles still exist. Meaning that no matter how short the release cycle, even when resource is available, business may need to wait till new tags go into production. This is a problem because analytics often relies on understanding what is the trend. To be able to provide a trendline, data must be collected in a longer time period and answers are not available immediately.

It is popular for business side to utilise tag managers to limit the dependency on development resource. While tag managers are very capable and allow quick changes and try to eliminate the need for programming skills, they require basic HTML and JS knowledge as the tag manager's GUI is just an abstraction for programming. In the process of configurating the tags, the usage of URL parts as variables and CSS selectors is standard practise. This however is a conflicting requirement because business side should not be responsible for any development aspects, especially for code that is generated by external tool.

Another issue is that business side is not close to regular development cycle. It may cause situations where development team makes changes that breaks currently active tag. Business side is forced to keep an overview of all of their set tags. This however can be a complicated process as there are no guarantees the tags are set most optimal way considering the business side HTML, JS and general web development knowledge may be limited. From here another logical error rises – does business side want to be responsible for possible errors created by a broken tag and do they have the competence to fix the problem.

In addition to HTML and JS, analytics tools setup requires understanding of HTTP requests and when something is sent because otherwise, for example when platform is a SPA, can result in information not sent at correct times or being duplicated falsely.

Furthermore, companies may have multiple platforms or multiple tools. Maintenance of these tools can quickly grow and result in demands for the business side which they have no resources allocated. Having business side manage analytics implementation can also scatter the responsibility as it is probable that sales and UX have different needs. That can lead also to overlap of implemented tags unless there is central governance set up.

Some of the problems stated above could be mitigated by review processes where no tag can go live without IT review. That would however conflict with the core premise of tag managers that is making quick changes without the need of IT resource. It cannot be guaranteed that IT can be immediately ready to take changes into review, especially considering agile methodologies include teams often work in sprints where the tasks are pre-selected.

Considering all this, it is in business' interest to not take responsibility for any services that generate code on company's websites and instead rely on IT teams. This however can cause an inner conflict of priorities when giving input to IT, assuming that products, marketing and other stakeholders do not want IT to spend too much time on implementing analytics as it limits IT time spent on new features that the customers are waiting. In order to mitigate analytics competing with developing features for the customer, a quickly implemented and managed solution is needed.

3.2.2 IT perspective

One of key aspects that IT needs to ensure is reliable web platform that functions at full capacity for all users at all times. Thus, IT is interested in having full control over implementations and system monitoring. As discussed before, to remove the need for manual tagging, tag managers are often used. Considering reliability and control, tag managers have multiple negative sides.

First, tag managers generate code outside of platform source and inject it from a third-party server. While tag mangers offer separate staging and development tags, there is always a possibility that production environment has some differences that creates an error at runtime. Debugging of such error may be more difficult as the tag manager is a system outside of normal development framework.

Another potential issue that can rise with using tag manager systems is uncontrolled use of global variables and functions. Although in regular development process, such conflicts should present themselves, the IT team can never verify to full extent. For demonstration, below is shown a simple code example that results in an error. The user has overwritten *Google Analytics* global function `ga` with a custom string but the function is still called out on line 3 as *Google Analytics* function.

```
 1.  <!-- Injected tag -->
 2.  <script>
 3.  var ga = "another value for ga";
 4.  // 'ga' variable used as needed
 5.  </script>
 6.  <!–- ... page code -->
 7.  <script>
 8.  // 'ga' called resulting an error
 9.  ga('send', 'pageview');
10.  </script>
```

Figure 7. Demonstration of global Google Analytics method overwritten resulting an error in later method call.

Simo Ahava, a Google Developer Expert for *Google Analytics* and *Google Tag Manager*, points to best practise to encapsulate all custom scripts injected with tag managers in an immediately self-invoking function because JavaScript scopes variables to their running context [46]. Sample below would ensure no errors are present.

```
 1.  <script>
 2.  (function() {
 3.      var ga = "another value for ga";
 5.      // 'ga' variable used as needed
 6.  })();
 7.  </script>
 8.  <script>
 9.  // ’ga’ called resulting in a successful request
10.  ga('send', 'pageview');
11.  </script>
```

Figure 8. Demonstration of script snippet encapsulated by an immediately self-invoking function allowing resulting in successful call later to global Google Analytics function with a same name.

It can also happen that core library is updated and contains a non-encapsulated global variable. In 2017, a released update for *Google Tag Manager* set a global variable `f`. This resulted in broken websites for all users who had defined `f` globally as something else. [47] Encapsulation mitigates some of these problems from tag manager's side, however developer can still overwrite analytics related global variable by accident.

Another reason why tag manager systems may not be suitable options is security. Because tag managers allow injecting custom code, they are inherently a security risk. Many areas, for example financial service providers, may not want to accept this risk.

Furthermore, companies can opt for multiple analytics tools. Often tag manager of one tool is used to also manage tags of other tools. This results in interacting through a middle layer which the secondary tool is not prepared for. *Mixpanel* brings this issue out in their frequently asked questions section, stating that while it is possible to implement *Mixpanel* through *Google Tag Manager*, it requires interactions through *Google Tag Manager*'s data layer that can lead to unexpected behaviour. [48]

Lastly, tag managers exist outside of release cycle unless a certain management process has been implemented in the company. As discussed during the business side analysis, this results in hindering the core power of tag manager system which is being out of release cycle and allowing quick changes. All in all, tag manager systems may not be viable options for many companies.

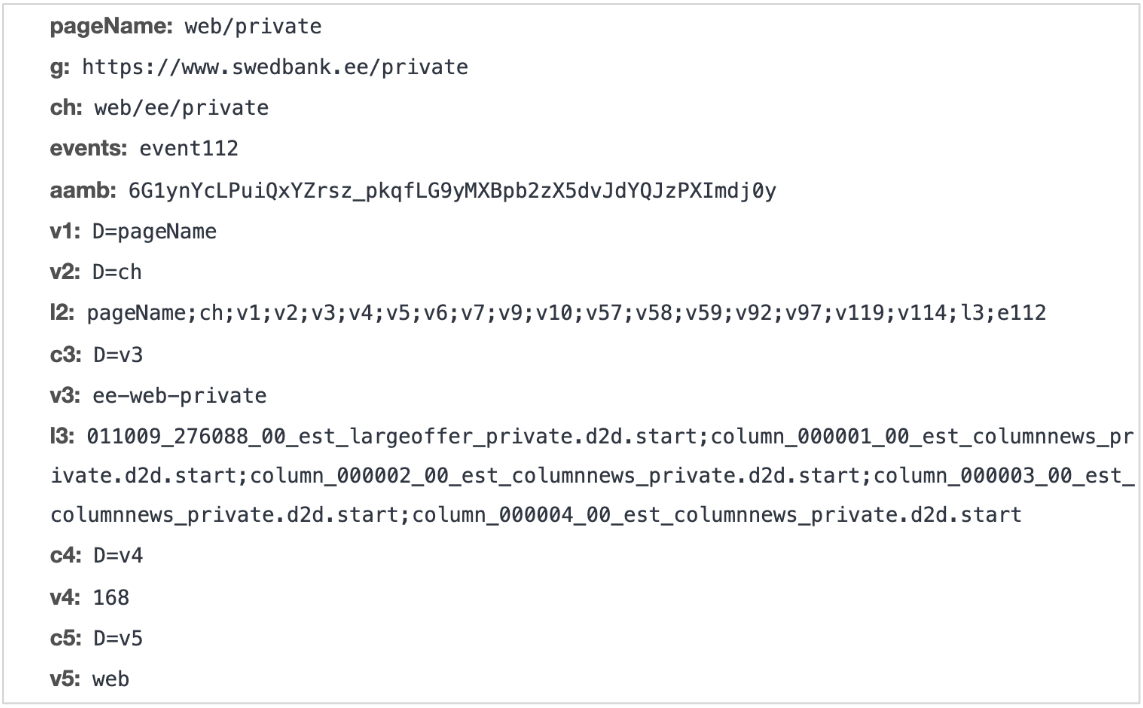
Alternative is opting for manual tagging and having a separate library copy available at company source. Companies may use multiple different tools and have duplicating tags for different tools on same components quickly creating unnecessary complexity and tags scattered around the code. To do manual tagging efficiently, IT must provide a custom framework or provide universal components, a user interface kit, that contains implementation of analytics triggers built in. The custom solution must easily support additional tools in the future. Developing such support for user interface kit or a custom analytics framework on a website can result in a large project. Creating such features for UI kit does not necessarily solve the problem for all company websites because company can be using different components and frameworks on platforms and the particular UI kit is not necessarily universal.

3.2.3 Web analytics perspective

Web analytics is a support function to business stakeholders and provide analytical views answering specific questions. Those questions may differ depending on stakeholder’s respective subject. Analytical views in core are however based on the same underlying customer behavioural data. To allow meaningful analysis, event data must also provide contextual information about the user, device, content presented, and elements involved in the interaction. An example could be measuring campaign effects on a product page. Sales and marketing question could be ”how many customers converted out of certain number of leads clicking on an apply button”. User experience team could ask on the same page if customers visiting from a mobile device use navigation menu compared to using the same apply button. To answer these questions certain necessary data points must be collected. Here are these data points listed on a high level:

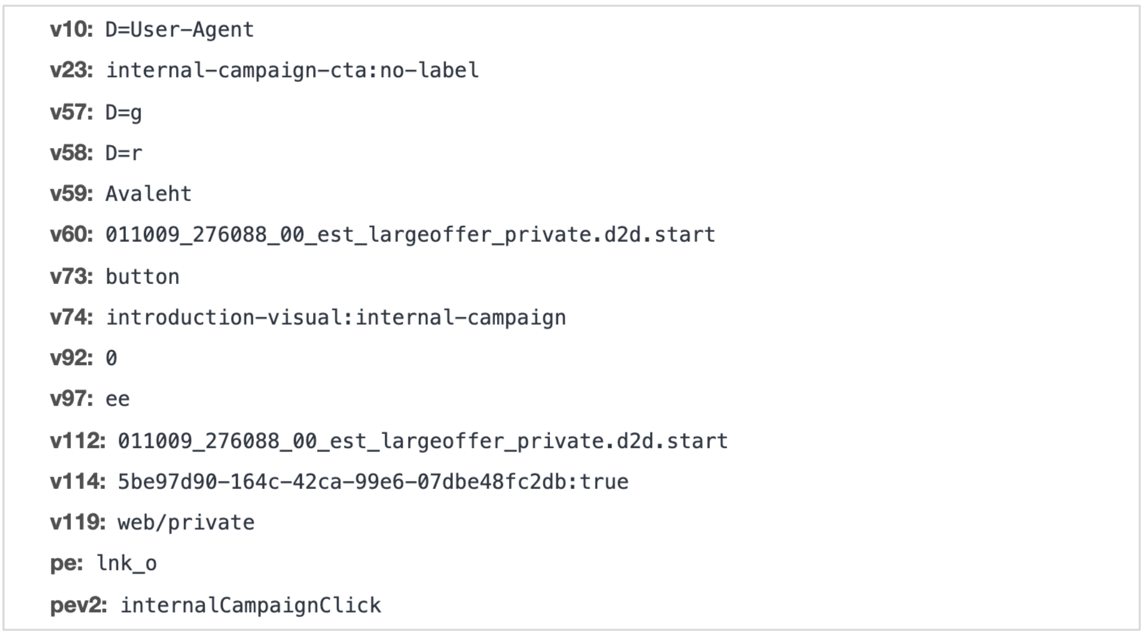
- Time of the visit
- Page information
- Visitor information
- Device information
- Behavioural events
    - Contextual information
        - Element type (button)
        - Element group (navigation, apply)
        - Element differentiation
- Marketing related events
    - Contextual information
        - Lead generating offer
        - Offer type (banner, email)

These derived requirements are also common in practise. For demonstration, outtakes of parameters collected on page load and click event from the website of Swedbank Estonia (Figure 9, Figure 10), largest bank in Estonia, and from the website of Pipedrive.com (Figure 11, Figure 12), a popular client relationship management tool service, are presented below.



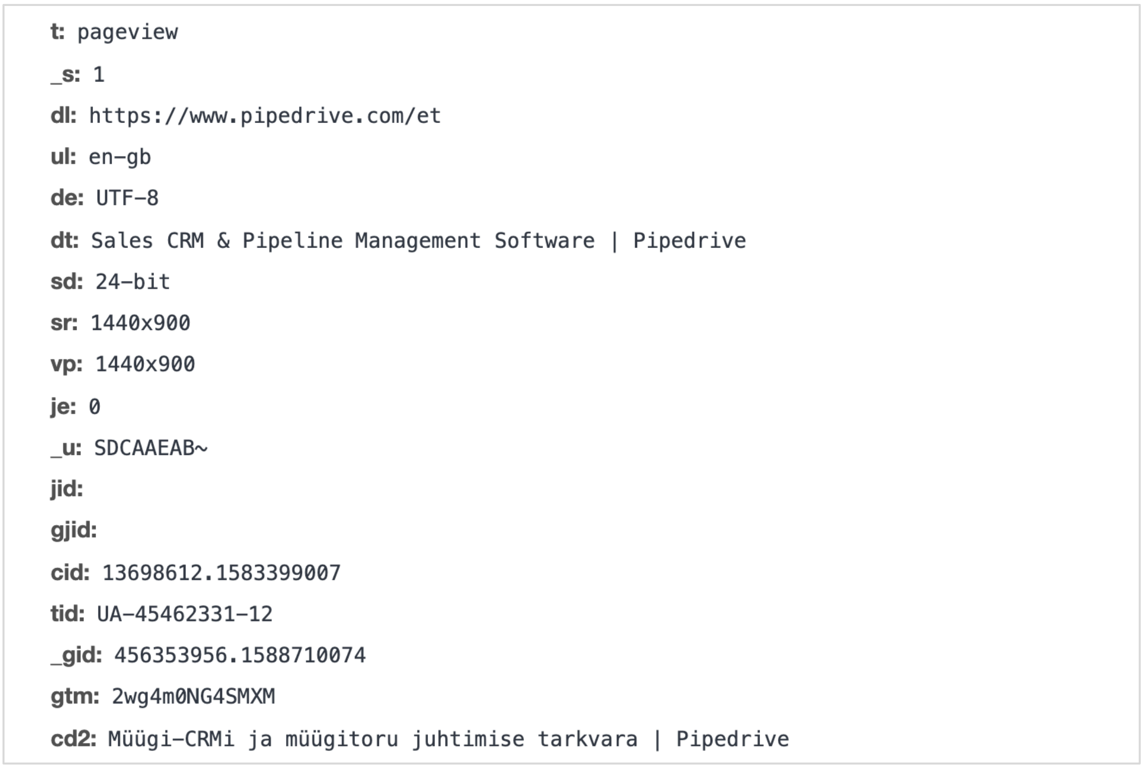
```
pageName: web/private
g: https://www.swedbank.ee/private
ch: web/ee/private
events: event112
aamb: 6G1ynYcLPuiQxYZrsz_pkqfLG9yMXBpb2zX5dvJdYQJzPXImdj0y
v1: D=pageName
v2: D=ch
l2: pageName;ch;v1;v2;v3;v4;v5;v6;v7;v9;v10;v57;v58;v59;v92;v97;v119;v114;l3;e112
c3: D=v3
v3: ee-web-private
l3: 011009_276088_00_est_largeoffer_private.d2d.start;column_000001_00_est_columnnews_pr
ivate.d2d.start;column_000002_00_est_columnnews_private.d2d.start;column_000003_00_est_
columnnews_private.d2d.start;column_000004_00_est_columnnews_private.d2d.start
c4: D=v4
v4: 168
c5: D=v5
v5: web
```

Figure 9. Screen capture from Google Chrome developer tools while visiting Swedbank Estonia’s website. Image shows a request made to Adobe Analytics collection server on page load.



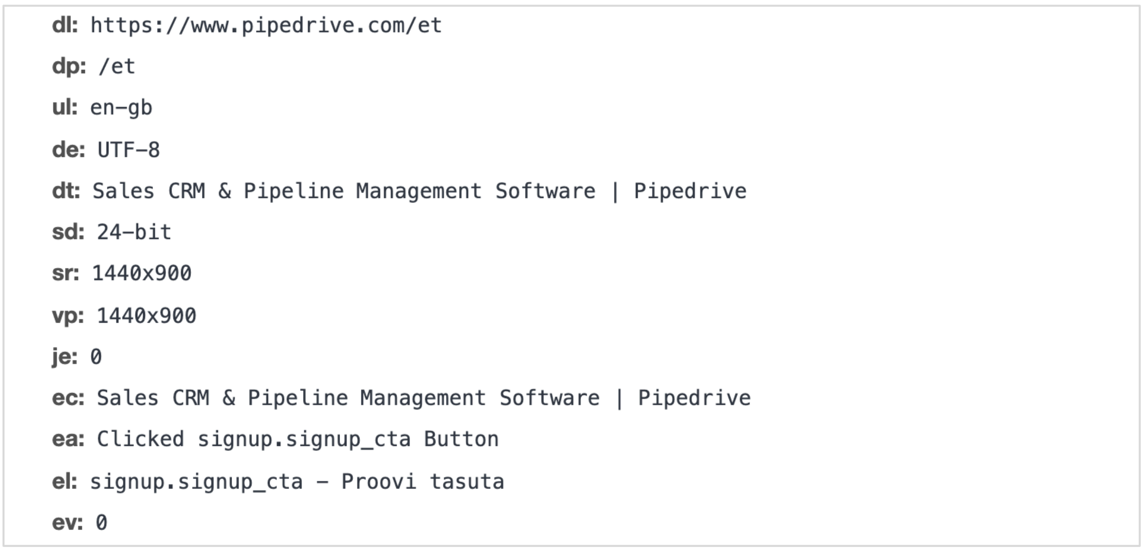
```
v10: D=User-Agent
v23: internal-campaign-cta:no-label
v57: D=g
v58: D=r
v59: Avaleht
v60: 011009_276088_00_est_largeoffer_private.d2d.start
v73: button
v74: introduction-visual:internal-campaign
v92: 0
v97: ee
v112: 011009_276088_00_est_largeoffer_private.d2d.start
v114: 5be97d90-164c-42ca-99e6-07dbe48fc2db:true
v119: web/private
pe: lnk_o
pev2: internalCampaignClick
```

Figure 10. Screen capture from Google Chrome developer tools while visiting Swedbank Estonia's website. Image shows a request made to Adobe Analytics collection server on a banner call to action button click on front page.



```
t: pageview
_s: 1
dl: https://www.pipedrive.com/et
ul: en-gb
de: UTF-8
dt: Sales CRM & Pipeline Management Software | Pipedrive
sd: 24-bit
sr: 1440x900
vp: 1440x900
je: 0
_u: SDCAAEAB~
jid:
gjid:
cid: 13698612.1583399007
tid: UA-45462331-12
_gid: 456353956.1588710074
gtm: 2wg4m0NG4SMXM
cd2: Müügi-CRMi ja müügitoru juhtimise tarkvara | Pipedrive
```

Figure 11. Screen capture from Google Chrome developer tools while visiting Pipedrive's website pipedrive.com. Image presents selection of parameters send to Google Analytics collection server on page load.

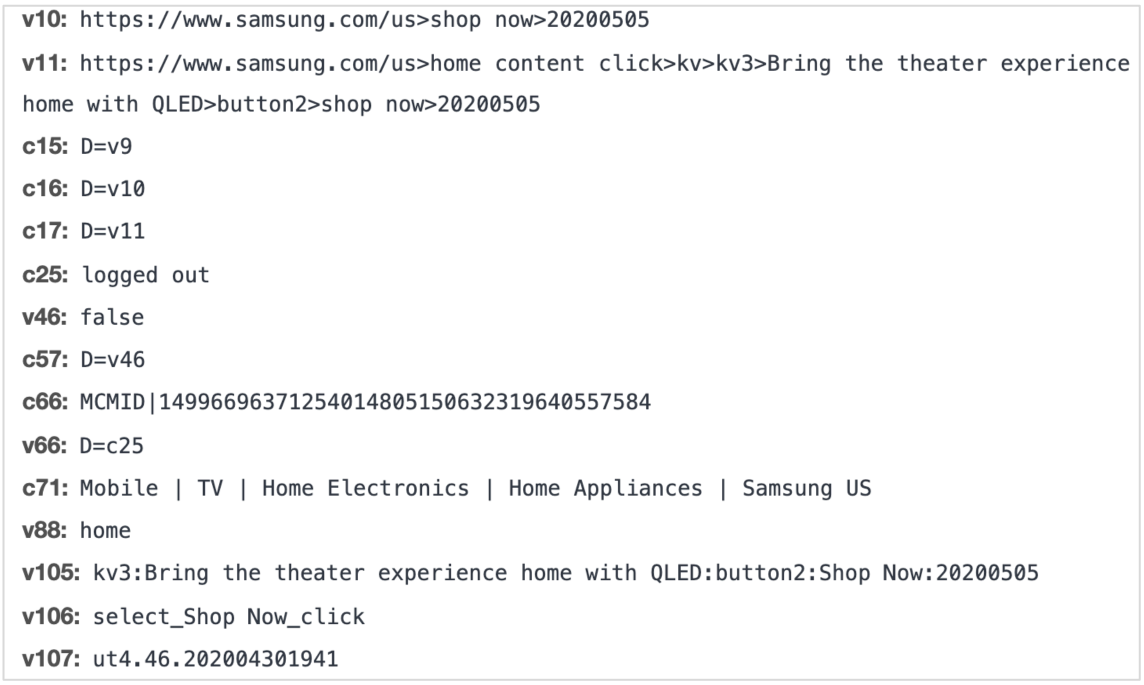


```
dl: https://www.pipedrive.com/et
dp: /et
ul: en-gb
de: UTF-8
dt: Sales CRM & Pipeline Management Software | Pipedrive
sd: 24-bit
sr: 1440x900
vp: 1440x900
je: 0
ec: Sales CRM & Pipeline Management Software | Pipedrive
ea: Clicked signup.signup_cta Button
el: signup.signup_cta - Proovi tasuta
ev: 0
```

Figure 12. Screen capture from Google Chrome developer tools while visiting Pipedrive's website pipedrive.com. Image presents selection of parameters send to Google Analytics collection server on click event.

On both of websites, general current location in website's hierarchy is collected among other information such as page title, screen resolution, site section et cetera. On Figure 9, Swedbank also collects loaded campaign advertisement values under variable "l3".

It can also be seen on Figure 10 and Figure 12 that click event includes additional contextual information such as that the event target was a button, what was the label of the particular button and which larger group or component this button was part of. Screen captures of full-length queries of requests are visible in Appendixes 1, 2, 3, 4 and 5. Similar examples can be found elsewhere around the web. Appendix 6 has a request example of a click on a header image button on official website of Samsung and Appendix 7 has a request example of an article view on New York Times website. Outtake of the request on Samsung's website is presented here.



```
v10: https://www.samsung.com/us>shop now>20200505
v11: https://www.samsung.com/us>home content click>kv>kv3>Bring the theater experience
home with QLED>button2>shop now>20200505
c15: D=v9
c16: D=v10
c17: D=v11
c25: logged out
v46: false
c57: D=v46
c66: MCMID|14996696371254014805150632319640557584
v66: D=c25
c71: Mobile | TV | Home Electronics | Home Appliances | Samsung US
v88: home
v105: kv3:Bring the theater experience home with QLED:button2:Shop Now:20200505
v106: select_Shop Now_click
v107: ut4.46.202004301941
```

Figure 13. Screen capture from Google Chrome's developer tools while visiting Samsung's website samsung.com/us. Image presents selection of parameters send to analytics collection server on front page banner button click.

As discussed before, some tools cover extensive list of page and marketing related data points in default configuration, however examining the real-world examples of companies and their implementation, it can be deducted that event tracking is core part of understanding customer behaviour on a website and to have meaningful understanding, tracking must also contain additional contextual information about interaction target elements that allows proper filtering and segmenting during data analysis.

3.3 Requirements

At first look, business side and IT have in many ways conflicting requirements with business side needing a way of making quick changes and coverage, opposed to IT needing efficiency, reliability and portability. However, an important aspect is that business side does not need the possibility to quickly change tags in production as this is a solution for secondary problem but not addressing the source problem. The source problem is that business side is missing some tags from the beginning and if all website components would be tracked on certain level with data quality ensured, this need would not exist. Considering this aspect, analysis above and real-world examples, certain requirements can be derived. Here is presented a table of high-level requirements from business perspective, IT perspective and what both sides share in common.

Table 1. Basic requirements to analytics implementation from business perspective, IT perspective and shared in common.

	Functional Requirements	**Non-Functional Requirements**
Business	Solution must send data on all business defined behavioural events Solution must collect all business defined contextual information Solution must be able to differentiate similar components in different parts of page	Solution must ensure high level data quality
IT	Solution must not cause conflicts with existing solutions Solution must eliminate tagging necessity Collect event related data without or minimal configuration Solution must support integration with different tools	Solution should be low maintenance Solution should be easily implemented Solution must be scalable Solution should be portable Solution must be secure Solution should be customizable
Common		Solution must be compliant with national requirements Solution should be low cost Solution should be reliable

To summarise, business side requires wide variety of information gathered from the website. Information gathered must contain sufficient contextual background to allow making accurate decision. IT requires a solution that would limit tagging and other configuration of data collection to minimal while allowing quick implementation, portability and reliability. Analytics requirements are common with business side with the exception that the data must be structured to allow easier management.

This table of requirements was shared with representatives of prominent business entities for additional verification. Respective representatives agreed with derived requirements. The conversations hereby are not disclosed.

3.4 Proposed solution

Taking into account defined requirements and analysis conducted above, it can be deducted that a solution which could provide structured data about interaction target elements on all commonly tracked events without additional tagging and minimal configuration would benefit all sides. Such solution would allow cleaner code, portability, scalability and would be quick to install while also ensuring necessary coverage.

The realisation is limited to JavaScript and should allow simple integration with analytics tools without dependencies. In addition, proposed solution should take into consideration privacy aspects and allow masking capabilities. Masking may also be beneficial for other than privacy related reasons, for example in aggregating interactions with particular link or element where related numbers are not particularly important from analytical aspect. An example of such case could be a link that directs to a file where the link label contains the date of the last update time of that file, however that date is not important for the user.

3.4.1 Existing similar solutions

Possibly similar solution is offered by company Keen as a tool, including also a visualization layer. They have published a "keen-tracking" package on Npmjs.com to support that tool. The library holds, according to documentation, variety of possibilities and offers auto tracking features based on robust data models. It is a fairly large library at 1.36 megabytes and is a paid solution that demands identification to be used. [49]

Adobe Analytics provides activity map feature that allows automatic click tracking. It aims to collect all clicked elements by finding link action and region information from DOM. The feature allows limited customizability and is *Adobe Analytics* specific and with limited functionality [50].

Members of *Google Analytics* developer platform team have unofficially created an "autotrack" package which is available at Npmjs.com. The goal of the package is to provide a set of plugins to *Google Analytics* that allow automatically track certain behavioural events. For example, plugin “outboundLinkTracker” automatically tracks links to external domains. Some plugins need additional data attributes to be implemented. The library is strongly coupled to *Google Analytics* core analytics.js library, does not work with other libraries nor *Google Tag Manager* and has not been maintained with last update three years ago [51] [52].

There are smaller packages which help or automate some aspects of click tracking with the help of data attributes or tracking specific elements such as YouTube videos [53] [54].

From integration aspect, David Wells has created a library “analytics”. *Analytics* is an abstraction layer for tracking events on website and aims to provide pluggable solution. The library is very lightweight at 13.2 kilobytes in minimized form and has large list of plugins already available including *Google Analytics, Google Tag Manager* and others. The library also provides number of pre-defined helpful event listeners a user can utilise to control the flow of data collection [55].

Analytics library does not provide out-of-the box collection of data. The user is responsible for providing input as is common with regular implementation. *Analytics* then sets parameters for other analytics tools using tool provided methods. User needs to provide mapping in case custom dimensions exist [55].

Analytics library can be potentially used as integration handler. This would allow reusing existing plugins already available and limit the need of creating new extension modules.

No other solutions that would aim to cover all behavioural tracking in a structured manner was found.

3.4.2 Technical challenges

Providing automatized data collection in a structured way has some challenges. While HTML is structured, there is no specific required way or order set for usage of elements and developers can combine them according to their preference. There are some exceptions to this, mostly lists and tables related elements, but these are limited [56]. Another challenge is understanding element’s particular usage by its parameters and surrounding elements. HTML5 does encourage semantic mark-up but it is not a rule and websites developer can opt for usage of only `<div>` elements [57]. It is relatively common to use `<header><footer>` and `<main>` elements but it is not guaranteed.

In addition to unknown structure of HTML, in some cases, elements are not used as they are designed but handled by JavaScript on the side in hidden mode. Common example of this in practise is the creation of custom checkbox elements out of other non `<input>` tags. A JavaScript event listener function is assigned to this custom element that delegates customer's input to hidden functional `<input>` element.

These challenges in mind, the library should be open to customization with minimal effort in order to provider extended capabilities. While this would mean some configuration, benefits of the functional aspects would still be present, eliminating tagging and making available universal data structure everywhere on collection. Furthermore, other mitigating actions could be performed. Templates for popular design and component frameworks such as Bootstrap4 and Material Design by Google could be included.

In case of auto tracking, DOM tree could be travelled upwards for a better candidate as interaction element. For example, a hypothetical case can be visualised where a `<a>` element is inside a `<li>` element in navigation menu and the `<a>` element covers nearly all of `<li>` element. The reason for this would be that `<li>` is used for menu items placement and have little margin. The user does not make a difference between those elements and interacts with the `<li>` and `<a>` element simultaneously. According to statista.com, mobile share of worldwide traffic in the fourth quarter of 2019 was 52,6%. With many companies understanding the importance of mobile, they are also updating UI to be ready for mobile user. According to UX Movement website, optimal size and spacing for mobile website buttons is 42 pixels to 72 pixels [58]. The allowed difference could be set at 10px on all sides. The goal of the process would be to rise the chance of finding an element with an identifier. Without a universal identifier it may be complicated to differentiate elements when, for example, there are multiple languages, meaning there are multiple labels for one button. The conditions could be looped multiple times in case such coverage exists and none of the identification is available on lower level.

Another question is finding labels. In most cases labels are often easily found as `innerText` property of interacted element. For dropdowns the labels are provided commonly as separate `<label>` tag and this has to be taken into consideration. Label value is vital for analytics to provide context thus ensuring the existence of it is essential. The value itself however is forgiving because the value should be visible for the user on the website thus it can be always searched in live for better understanding.

Lastly, considering performance, the user should have an option to limit the triggering rate of info. JavaScript events like “resize”, “scroll”, “mousemove” and many more are fired very often, and it can affect strongly browser performance.

4 Development of proposed library

In the first part of this chapter technical planning is done. Second half proceeds with the realisation of the solution.

4.1 Planning

First part describes in detail general frameworks and components used during development. Secondly, the main planned logic of library is viewed closer.

4.1.1 Frameworks and components used during development

Considering IT non-functional requirements from section three, to ensure reliability, library will be written using TypeScript. TypeScript is a superset of JavaScript that is transpiled (converted) into plain JavaScript. This is necessary because browsers are not able to interpret TypeScript.

TypeScript enables usage of types, static typing, interfaces, classes and other classical object-oriented programming paradigm possibilities. Usage of these features ensures reliability as it limits possible errors in code [59]. Support for TypeScript has grown quickly over the last years with many modern frameworks such as ReactJs and Angular supporting or using TypeScript by default [60] [61]. Study by Stackoverflow.com revealed that TypeScript is third most loved language by developers [62]. This is also presented in number of related questions on Stackoverflow.com. Share of TypeScript related questions have grown from 0,10% to ~1,70% in the past years [63].

In addition, TypeScript's compiler will allow defining specific target JavaScript version to which source code is transpiled without loss of functionality even when target JS version does not have all features available that may be used in the source code. To ensure that developed library supports older browsers, ECMAScript 5 (ES5) is chosen as target version. According to caniuse.com website, 98,16% of all browsers (tracked with cookies) have full support for ES5 [64]. In order to mitigate browser differences in ES5 standard implementation, polyfills will be used. Polyfill is a snippet of code that adds a function in case a web browser does not support it by default. They are generally used in case a feature is a part of established standard and not implemented by particular browsers' versions. Polyfills will allow usage of features during development regardless if a specific browser supports it [65].

Furthermore, for additional reliability and portability support, it has been tried to limit dependencies. Having lower number of dependencies allows quick installation, puts no additional import requirements on the site owner and lowers the need of regular maintenance.

For easier code management during development, the code will be separated into several files. These files are bundled together on production build of the library. This process will be handled by Webpack. Webpack is an open-source JavaScript module bundler that is also capable of transforming and packaging [66].

4.1.2 Logic, structure and principles

The logic and structure aim to follow common best practises and principles of software development. This means, all components of the library are designed with the aim to have single responsibility, they are kept simple and are open for extension.

The library will provide a singleton on initialisation after a registration function is called with additional configuration object. This instance will manage data collection related configuration, listeners and methods. Configuration object contains defined components and consumer callbacks. Additionally, it will be possible to set debug mode active, use masking features and manage rate of event triggering.

Components break into two categories - targets and containers. Target component is a low-level element that user directly interacts with. Examples of suitable element for a target component are a link, button and checkbox. User can provide selectors to include higher level element as a target. Every target has a name and selector and event that a listener is defined for.

A container is a higher-level element in the DOM tree that is also a parent for target component. One target component can have multiple parents thus a target component can have multiple containers. These containers will be collected in the same order as their hierarchy in the DOM tree. Examples of containers include 'header', 'footer', 'nav'.

During initialisation, event listeners are set according to targets set in the configuration. Once a listener is triggered, a tracker object is created. Tracker object's methods will read DOM and populate context info in pre-defined structure. This object is sent as input to a function that will trigger all suitable callbacks to forward data to consumers.

4.2 Implementation

In this chapter the realisation of planned solution is described. First, the handling of library instance registration and options are covered. Secondly, how the element is analysed and thirdly, how the data is forwarded. Library is named "Kollektor" which is derived from the English word "collection" where letter "c" is replaced with a "k".

4.2.1 User options and configuration

On library load using a `<script>` tag, a global object `_kollektor` will be made available. User must first call registration method of this object and pass options as input. The only requirement is to include at least one consumer. Other options are optional and library will automatically set default values in case an option is not specifically defined. The only exception to this behaviour is in situations when used template is defined as "custom". In such case, library will expect targets and containers among consumers, however the library can handle and revert to default when they are not present.

Once options are given, the `register` function immediately controls that at least one callback or consumer is defined. If this condition is true, Kollektor class instance creation method is called. Example of minimalistic setup is demonstrated below on Figure 14.

```
<script>
var bootstrapKollektor = _kollektor.register({
  template: "bootstrap4",
  consumers: [
      {
        name: "tool-1",
        map: {
          param1: 'collectedProperties.label'
        },
        handler: (event, data) => {
          console.log({
            message: "Callback to specific tool",
            payload: data
          })
        },
        events: ["click"]
      }
    ]
  });
bootstrapKollektor.track();
</script>
```

Figure 14. Minimalistic example of library registration. The user has defined the usage of "bootstrap4" template and provided one consumer.

After instance is successfully configurated the `track` method can be called. On `track` method call, event listeners are created according to loaded targets and tracking begins.

User also has an option to use one of pre-defined templates. On Figure 14 above, `bootstrap4` template is chosen. A demonstration of minimal custom template configuration that listens only clicks on `<a>` and `<button>` element is demonstrated below on Figure 15.

Among using custom template, in the sample below, user has turned on privacy masking and set certain excluded selectors. Privacy masking will mask all numbers longer than set number of digits. Default limit is 5 digits.

```
<script>
var myKollektor = _kollektor.register({
  template: "custom",
  isDebug: true,
  privacy: {
      masking: true,
      limit: 3,
      excludedSelectors: ["input[type='tel']"]
  },
  targets: [
    {
      name: "link",
      selector: "a",
      events: ["click"],
      labelAttribute: "data-label",
      identifierAttribute: "data-uid",
      condition: function(element) {
          return element.hasAttribute("someAttribute");
      }
    },
    {
      name: "button",
      selector: "button",
      events: ["click"]
    }
  ],
  containers: [],
  consumers: [
    {
      map: {
        param1: 'collectedPropreties.label'
      },
      handler: (event, data) => {
        console.log({
          message: "1st callback",
          payload: data
        })
      },
      events: ["click"]
    }
  ]
});
myKollektor.track();
</script>
```

Figure 15. Kollektor registration with a simple custom template. The user has provided two targets on "link" and "button". The user has also set debug and privacy settings on.

User also has the possibility to limit the processing rate of particular events or „debounce" them for a certain period of time. This behaviour can be set for all events at once or to specific events separately. Default template debounces „resize" and „scroll" events with a delay of 500 milliseconds. This means that listener's callback is not called until 500 milliseconds have passed from the last time the listener was triggered (Figure 16).

```
debounce: [
  {
    event: "resize",
    delay: 500
  },
  {
    event: "scroll",
    delay: 500
  }
],
scrollDistances: [25, 50]
```

Figure 16. Example of choosing debounced events and tracked scroll distances on configuration.

Lastly, user can populate `scrollDistances` property with certain percentage numbers to send scroll distance information (Figure 16).

Options `isDebug`, `privacy`, `debounce` and `scrollDistances` can also be overridden when using a non-custom template.

4.2.2 Interaction data collection

Interaction data collection is separated into two different aspects. Interaction tracking or cases where customer interacts with specific targets and scroll tracking.

In case of tracking interactions where targets are involved, once an event listener has been triggered, the listened element is passed to `analyseInteractionEvent` method that is responsible for pre-analysis. This method controls that a matching target in the options exits and that the element does not match excluded selectors. If the conditions are true, method proceeds with tracker object creation. On tracker object instance creation, constructor calls two main methods – `populateNativeProperties` and `populateData`. The first sets values for `InteractionTracker` class property `nativeProperties`. That property gives access to different values collected from the element without any modification. These are id, CSS classes, attributes, role, type, label, href, onclick and style.

The second method, `populateData` proceeds with internal analysis to populate data in a structured form. One of key actions is determining best candidate for data collection as it is not always the target itself. When conditional function is provided for the target, the method uses that as reference. In case there was no condition, method proceeds to analyse and choose which element would be suitable to provide as interacted element to tracker object creation.

Considering technical challenges analysis above regards user true interaction target, in situations where

1) the listener element covers 95% of the parent element
2) and is not smaller than 10 pixels in height or width from its parent
3) and the current element is missing an id property value

the method takes the parent element as interacted element. This check is repeated till mentioned conditions are not true.

Element's native properties, surrounding siblings and parents are processed to provide information under `collectedProperties` property. `collectedProperties` property values include type, label, identifier, action, containers, isLink, isOutBound and action.

Type matches the type property from matched target component provided in the options of Kollektor instance. Label matches calculated target element object `innerText` property value or dropdown `<label>` tag value. Identifier aims to be unique on the website. In default configuration "id" is chosen as source, but the user can provide their own when using a custom template. Action is a concatenation of event type and matched target name. Container is all containers found delimited by colon.

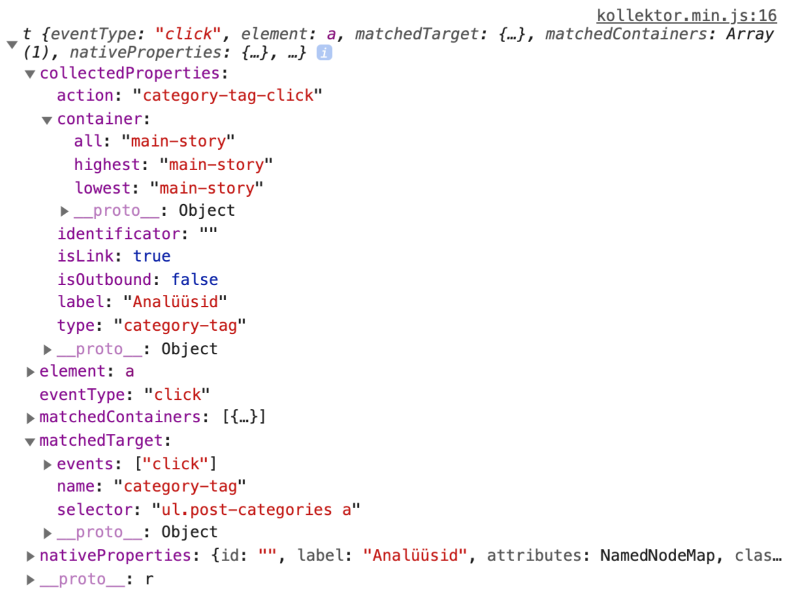
For label and identifier attributes, special attributes as source can be set. When not set, library makes additional check if the element is input. If is input, tries to find `<label>`

tag among siblings. As last resort, native property `innerText` is used. For universal identifier, id property is fallback. When `id` is not available but is outbound link, `href` attribute is collected. If no condition was suitable, value is returned as empty string.

The end result tracker object includes

- Set of default properties available under `nativeProperties` property
- Set of calculated properties available under `collectedProperties` property
- Additional original event type
- Used element
- Matched target from the loaded options given on registration
- Matched containers from the loaded options given on registration

An example of such object is presented on Figure 17.



```
                                                               kollektor.min.js:16
▼ t {eventType: "click", element: a, matchedTarget: {…}, matchedContainers: Array
  (1), nativeProperties: {…}, …}
  ▼ collectedProperties:
      action: "category-tag-click"
    ▼ container:
        all: "main-story"
        highest: "main-story"
        lowest: "main-story"
      ▶ __proto__: Object
      identificator: ""
      isLink: true
      isOutbound: false
      label: "Analüüsid"
      type: "category-tag"
    ▶ __proto__: Object
  ▶ element: a
    eventType: "click"
  ▶ matchedContainers: [{…}]
  ▼ matchedTarget:
    ▶ events: ["click"]
      name: "category-tag"
      selector: "ul.post-categories a"
    ▶ __proto__: Object
  ▶ nativeProperties: {id: "", label: "Analüüsid", attributes: NamedNodeMap, clas…
  ▶ __proto__: r
```

Figure 17. Tracker object properties available for forwarding.

The logic of scroll tracking follows the same basic steps as regular interaction tracking. When an event listener has been triggered, the listened element is passed to `analyseScrollEvent` method that is responsible for pre-analysis. This pre-analysis consists of calculation of scroll distance and control if any of the set scroll distances has been reached and that this scroll distance is not already sent before. If the conditions are true, scroll specific tracker object is created and sent to consumers. Number of collected values compared to interaction tracking is small and contains only “action” and “label” as properties in `collectedProperties` object. The scroll tracking tracker is open for future addition of parameters and methods if suggestions are made.

4.2.3 Forwarding data

Once event data has been analysed, the created tracker object is passed to `sendToAllConsumers` method. This method will trigger data forwarding to provided consumer callbacks. Method controls for every consumer that current event is also set in the events list of the consumer. If so, consumer provided mapping is read to create a necessary data object according to consumers requirements. Mapping consists of a map object, where keys are the names of properties consumer needs and values are tracker object properties in string representation. For demonstration, an example of simple map is shown for *Google Analytics*.

```
const map = {
  'event_category': 'collectedProperties.action',
  'event_label': 'collectedProperties.label'
}
```

Figure 18. Demonstration of a consumer map object that defines requirements of needed. In this example, a simple map for Google Analytics is presented.

The map object is sent as input to `mapToCallbackObject` function that creates a new object according to consumer needs (Figure 19). Created object is passed to callback methods, for example *Google Analytics* consumer. The `mapToCallbackObject` function can handle situations when a property is not presented and returns empty string in such cases. In case value type is `String`, value is also passed for privacy masking control.

```
/**
 * Creates an object with properties according to map
 * property in callback object.
 *
 * @param tracker tracker object with collected info.
 * @param cbObject custom callback map object.
 */
private mapToCallbackObject(tracker: ITracker, cbObject:
Map<string, string>): object {
    this.messenger.log("mapToCallbackObject()");
    const data: {[key: string]: any} = {};
    Object.entries(cbObject).forEach( ([k, v]) => {
        const val = v.split('.')
                        .reduce((a: object, b: keyof object)
                        => a[b], tracker) || "";
        data[k] = typeof(val) == "string"
        ? this.privacyManager
            .maskNumbersLongerThanLimit(val)
        : val;
    });
    return data;
}
```

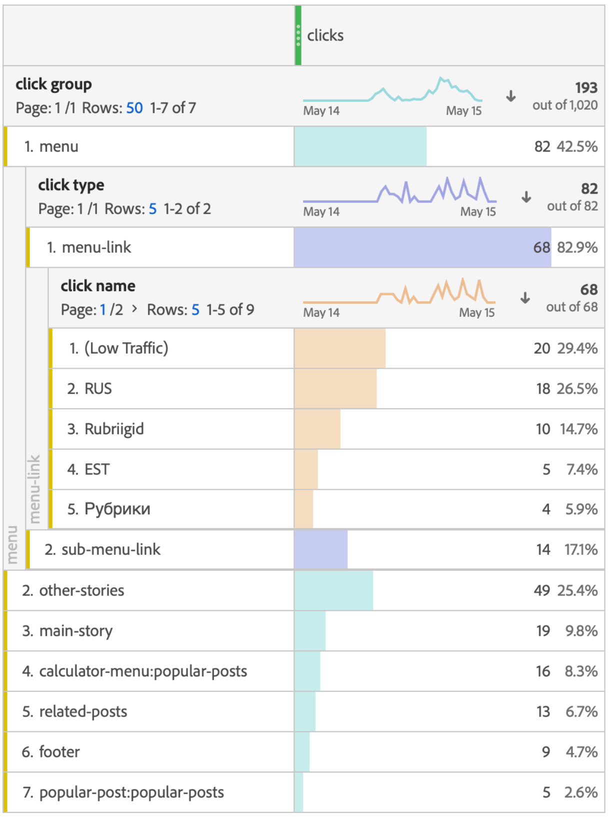
Figure 19. Function that provides an object with necessary values according to consumer provided map.

5 Result

As a result of this thesis a JavaScript library was developed that allows site owner collect web interaction target element data in structured manner in a single action and provide this data as input for all tools used on the site. The library and its source code are published under MIT licence on Github version control platform and also on Npmjs.com. Repository name on Github is *kollektor.js* [67] and package name on Npmjs.com *kollektor-js* [68].

The library provides a default configuration for quick data collection for common target elements and events. User can provide their own configuration for higher quality level data collection or edge cases. The configuration consists of targets, containers and consumers. In addition, the site owner can turn on privacy masking mode and provide CSS selectors to mark excluded elements. In case interaction target follows default configuration or no special conditions are provided by user, the library includes simple analysis to help collect more contextual information than particular element has, provided it is possible.

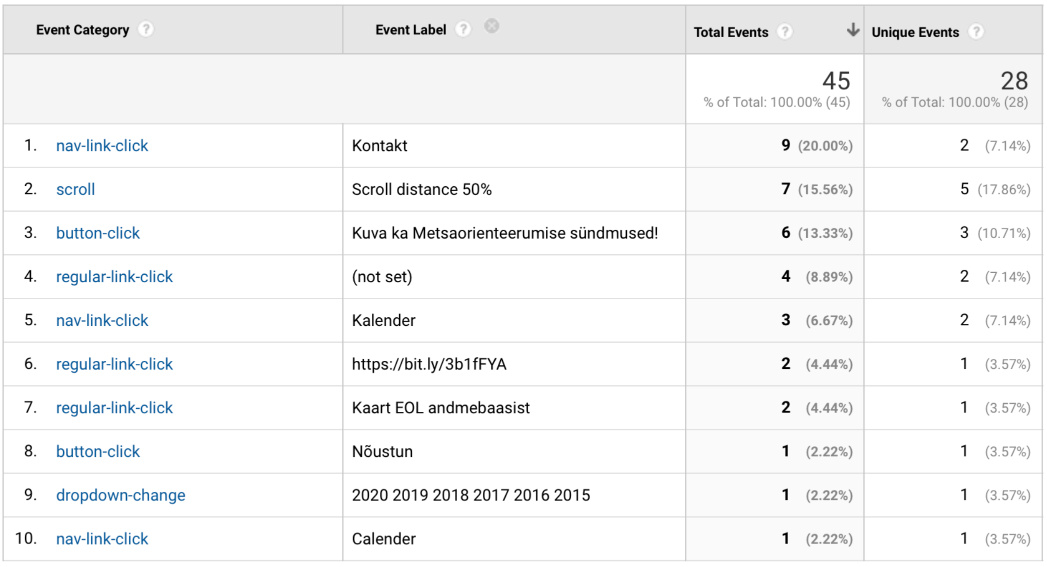
Library was implemented on two different websites for proof of concept. First proof of concept usage was set up on a website where main web analytics was done using *Adobe Analytics* and also *Google Analytics*. *Adobe Analytics* was set up via usage of *Adobe Launch*, Adobe's tag manager system. *Google Analytics* was implemented by using tool's core library directly. None of the systems were set to track any event data. The implementation was done using custom template. Below is shown an example of data collected after implementing the library.



	clicks
click group Page: 1 /1 Rows: 50 1-7 of 7	193 out of 1,020
1. menu	82 42.5%
click type Page: 1 /1 Rows: 5 1-2 of 2	82 out of 82
1. menu-link	68 82.9%
click name Page: 1 /2 › Rows: 5 1-5 of 9	68 out of 68
1. (Low Traffic)	20 29.4%
2. RUS	18 26.5%
3. Rubriigid	10 14.7%
4. EST	5 7.4%
5. Рубрики	4 5.9%
2. sub-menu-link	14 17.1%
2. other-stories	49 25.4%
3. main-story	19 9.8%
4. calculator-menu:popular-posts	16 8.3%
5. related-posts	13 6.7%
6. footer	9 4.7%
7. popular-post:popular-posts	5 2.6%

Figure 20. Example of collected data. Screen capture from Adobe Analytics.

Secondly, the library was also set up on a website that was using Bootsrap4. The website had existing *Google Tag Manager* set up, however did not track any user interactions other than pageview. The library was implemented with one consumer and using pre-defined template for Bootstrap4. Example of collected data is presented below.



	Event Category	Event Label	Total Events	Unique Events
			45 % of Total: 100.00% (45)	28 % of Total: 100.00% (28)
1.	nav-link-click	Kontakt	9 (20.00%)	2 (7.14%)
2.	scroll	Scroll distance 50%	7 (15.56%)	5 (17.86%)
3.	button-click	Kuva ka Metsaorienteerumise sündmused!	6 (13.33%)	3 (10.71%)
4.	regular-link-click	(not set)	4 (8.89%)	2 (7.14%)
5.	nav-link-click	Kalender	3 (6.67%)	2 (7.14%)
6.	regular-link-click	https://bit.ly/3b1fFYA	2 (4.44%)	1 (3.57%)
7.	regular-link-click	Kaart EOL andmebaasist	2 (4.44%)	1 (3.57%)
8.	button-click	Nõustun	1 (2.22%)	1 (3.57%)
9.	dropdown-change	2020 2019 2018 2017 2016 2015	1 (2.22%)	1 (3.57%)
10.	nav-link-click	Calender	1 (2.22%)	1 (3.57%)

Figure 21. Example of collected data using pre-defined Bootstrap4 template.

6 Future developments

The target differentiation of social media related links could be done automatically. To do this, URLs of links could be analysed using regular expressions provided by Karl Lorey on Github.com repository "Regular expressions for social media profiles” [69].

In addition, device differences could be considered more in automatic collection. For example, touch devices generally have larger elements. Furthermore, separate plugins could be developed for quick integrations with common tools.

From reliability aspect, automated testing during build process of library should be added in the future.

Lastly, in larger view, the automatic collection features could be improved not only by using relatively easy deductions but instead large-scale analysis. To do this, a large set of websites could be studied to find additional common trends and practises. Using this analysis, the library could try to do a pre-analysis of particular website in order to match the site to mapped similar other entities. This would lessen the need for configuration by user.

Summary

This thesis expanded on development of a JavaScript library that collects web interaction target element data in structured manner. As result of this thesis a working library was created and published on Github version control platform and Npmjs.com. The library allows quick implementation of event tracking and forwarding of data for all analytics tools used on the site.

The thesis consists of six sections – objective, background, analysis, realisation, result and future developments. Firstly, objectives of thesis are defined shortly. In the background section of the thesis, current solutions and practises are covered. In addition, brief history is given and an overview of surrounding environment, including legal aspects, is discussed.

In the analysis section, similarities and differences of the tools currently on the market are presented, the drawbacks of those solutions are viewed from business, IT and analytics perspective. Based on this analysis simple requirements are defined and compared to in practise samples. A solution is proposed as a result of the analysis.

In the realisation section, planning and development of the data collection library is covered. Following with results, collected data using the library is presented on two websites. Lastly, possible next steps are viewed in the future developments section.

All steps covered in this thesis provided a solution for the problem set in the objective part of the thesis. As a result of this thesis a library was provided that allows collection of web interaction data in a structured format with minimal configuration. The solution is quick to implement, allows full control and comes with default set up that provides data from first second.

Kokkuvõte

Käesoleva diplomitöö raames käsitleti JavaScript teegi loomist veebiaplikatsioonidel külastajate tegevusinfo kogumiseks. Töö tulemusel loodi lahendus, mis võimaldab minimaalse konfiguratsiooniga koguda veebilehe komponentide andmeid struktuursel kujul ning neid edastada samaaegselt mitmetele erinevatele veebianalüütika tööriistadele.

Diplomitöö koosneb kuuest peatükist – eesmärk, taust, analüüs, tehniline realisatsioon, tulemus ning tulevik. Esimeses peatükis kirjeldatakse lühidalt töö eesmärki. Teises peatükis antakse ülevaade tänastest turul olevatest lahendustest ning praktikatest. Lisaks on toodud välja ülevaade ümbritsevast keskkonnast, sealhulgas regulatsioonide suund maailmas ning lühidalt kaetakse ka veebianalüütika ajalugu.

Töö kolmandas peatükis uuriti esmalt turul olevate analüütikatööriistade paigaldusmetoodikate sarnasusi ja erinevusi. Seejärel analüüsiti levinud lahenduste puudusi äriliste üksuste, IT üksuste kui ka analüütikaüksuste vaatest. Analüüsi alusel tuletati erinevate osapoolte nõudmised andmete kogumise lahendusele ja kõrvutati nõudmisi reaalselt praktikas kogutavate andmetega tuntud veebilehtedel. Analüüsi tulemusel pakuti välja JavaScript teek, mis tegeleb komponentide andmete kogumisega kasutajate tegevustel struktuursel kujul. Lisaks anti ülevaade eksisteerivate sarnaste lahenduste osas ning analüüsiti pakutud lahenduse arenduslike väljakutseid.

Neljandas peatükis viidi läbi pakutud lahenduse tehniline planeering ja realisatsioon vastavalt teises osas teostatud analüüsile. Töö viiendas, tulemuste peatükis paigaldati teek kahele veebilehele. Viimaseks anti kuuendas peatükis lühike sissevaade võimalikele tuleviku arendustele.

Diplomitöö alguses püstitatud eesmärk leidis töö käigus lahenduse. Valminud teek publitseeriti Npmjs.com paketihaldus keskkonnas ning lähtekood laeti Github versioonihaldus keskkonda [67] [68]. Loodud teek võimaldab ettevõtetel ja eraisikutel kiirelt ja minimaalse seadistusega koguda kasutaja interaktsioonide andmeid ja neid edastada paljudele tööriistade. Teek on täies mahus lehe omaniku kontrolli all, kaotab

vajaduse ärilistel üksustel muretseda tehnilise paigalduse pärast ning võimaldab potentsiaalselt koguda rohkem andmeid vaikimisi seadistuses.

References

[1] J. G Zheng and S. Peltsverger, "Web Analytics Overview," in *Encyclopedia of Information Science and Technology, Third Edition*, IGI Global, 2015.

[2] ROI Revolution, “ollecting Web Data: A Look at Web Analytics Methodology,” ROI Revolution, [Online]. Available: https://www.roirevolution.com/blog/2006/05/collecting-web-data-a-look-at-web-analytics-methodology/. [Accessed 18 May 2020].

[3] The New York Times, “Google Acquires Urchin Software,” 29 March 2005. [Online]. Available: https://www.nytimes.com/2005/03/29/technology/google-acquires-urchin-software.html. [Accessed April 2020].

[4] W3Techs, “Usage statistics of traffic analysis tools for websites,” W3Techs, [Online]. Available: https://w3techs.com/technologies/overview/traffic_analysis. [Accessed 18 April 2020].

[5] Google, “Google Analytics Developer Guides - Measurement Protocol Overview,” Google, [Online]. Available: https://developers.google.com/analytics/devguides/collection/protocol/v1. [Accessed 15 April 2020].

[6] Adobe, “Analytics Implementation Guide - Implementing with hardcoded image requests,” Adobe, [Online]. Available: https://docs.adobe.com/content/help/en/analytics/implementation/other/hardcoded.html. [Accessed 15 April 2020].

[7] S. L. France and S. Ghose, “Marketing analytics: Methods, practice, implementation, and links to other fields,” *Expert Systems with Applications,* vol. 119, pp. 456-475, 1 April 2019.

[8] K. Ravi, Y. Khandelwal, R. Vadlamani and B. S. Krishna, “Analytics in/for cloud-an interdependence: A review,” *Journal of Network and Computer Applications,* vol. 102, pp. 17-37, 15 January 2018.

[9] M. Sean, Z. Li, S. Bandyopadhyay, J. Zhang and A. Ghalsasi, “Cloud computing — The business perspective,” *Decision Support Systems,* vol. 51, no. 1, pp. 176-189, April 2011.

[10] N. Subramanian and A. Jeyaraj, “Recent security challenges in cloud computing,” *Computers & Electrical Engineering,* vol. 71, pp. 28-42, October 2018.

[11] Matomo, “Matomo.org home page,” Matomo, [Online]. Available: https://matomo.org. [Accessed 19 April 2020].

[12] Google, “Google Analytics: Marketing Platform - Compare,” Google, [Online]. Available: https://marketingplatform.google.com/about/analytics/compare/. [Accessed 18 April 2020].

[13] Amplitude, “Product analytics for everyone,” Amplitude, [Online]. Available: https://amplitude.com/pricing?ref=nav. [Accessed 18 April 2020].

[14] Mixpanel, “Pricing,” Mixpanel, [Online]. Available: https://mixpanel.com/pricing/. [Accessed 18 April 2020].

[15] European Comission, “EU data protection rules,” 23 May 2018. [Online]. Available: https://ec.europa.eu/info/sites/info/files/data-protection-factsheet-changes_en.pdf. [Accessed 20 April 2020].

[16] State of California Department of Justice, “California Consumer Privacy Act (CCPA) Fact Sheet,” [Online]. Available: https://oag.ca.gov/system/files/attachments/press_releases/CCPA%20Fact%20Sheet%20%2800000002%29.pdf. [Accessed 20 April 2020].

[17] J. J. Lazzarotti, J. C. Gavejian, S. W. Silver, M. T. Costigan and D. A. Plummer, “New York SHIELD Act FAQs,” Jackson Lewis P.C, 11 March 2020. [Online]. Available: https://www.workplaceprivacyreport.com/2020/03/articles/new-york-shield-act/new-york-shield-act-faqs/. [Accessed 20 April 2020].

[18] European Commission, “Infographic - GDRP In Numbers,” 22 May 2019. [Online]. Available: https://ec.europa.eu/info/sites/info/files/infographic-gdpr_in_numbers.pdf. [Accessed 20 April 2020].

[19] Eurpean Commission, “ePrivacy factsheet,” Eurpean Commission, 24 August 2018. [Online]. Available:

https://ec.europa.eu/newsroom/dae/document.cfm?doc_id=41238. [Accessed 20 April 2020].

[20] European Commission, "Proposal for an ePrivacy Regulation," European Commission, 20 March 2020. [Online]. Available: https://ec.europa.eu/digital-single-market/en/proposal-eprivacy-regulation. [Accessed 20 April 2020].

[21] N. Lomas, "https://techcrunch.com/2020/01/06/facebook-data-misuse-and-voter-manipulation-back-in-the-frame-with-latest-cambridge-analytica-leaks/," TechCrunch, 6 January 2020. [Online]. Available: https://techcrunch.com/2020/01/06/facebook-data-misuse-and-voter-manipulation-back-in-the-frame-with-latest-cambridge-analytica-leaks/. [Accessed 20 April 2020].

[22] S. Brinker, "The average enterprise uses 91 marketing cloud services," 12 June 2017. [Online]. Available: https://chiefmartec.com/2017/06/average-enterprise-uses-91-marketing-cloud-services/. [Accessed 21 April 2020].

[23] M. Kihn, A. Sarner, A. Frank, C. Eubanks and L. F. Kune, "Magic Quadrant for Digital Marketing Analytics," Gartner, 2016.

[24] KissMetrics, "KissMetrics versus Google Analytics," KissMetrics, [Online]. Available: https://www.kissmetricshq.com/kissmetrics-vs-google-analytics/. [Accessed 19 April 2020].

[25] Adobe, "Cross-Device Analytics," Adobe, [Online]. Available: https://docs.adobe.com/content/help/en/analytics/components/cda/cda-home.html. [Accessed 6 May 2020].

[26] Google, "Google Analytics Developer Guides - Add analytics.js to Your Site," Google, [Online]. Available: https://developers.google.com/analytics/devguides/collection/analyticsjs. [Accessed 19 April 2020].

[27] Mixpanel, "Developer documentation - Implement Mixpanel," Mixpanel, [Online]. Available: https://developer.mixpanel.com/docs. [Accessed 19 April 2020].

[28] Adobe, "Analytics Implementation Guide - AppMeasurement for JavaScript," Adobbe, [Online]. Available:

https://docs.adobe.com/content/help/en/analytics/implementation/js/overview.html. [Accessed 19 April 2020].

[29] Amplitude, “Amplitude Developer,” Amplitude, [Online]. Available: https://developers.amplitude.com/. [Accessed 19 April 2020].

[30] Adobe, “Adobe Experience Platform Launch - Adobe Extensions,” Adobe, [Online]. Available: https://docs.adobelaunch.com/extension-reference/web. [Accessed 20 April 2020].

[31] Google, “Tag Manager Features,” Google, [Online]. Available: https://marketingplatform.google.com/about/tag-manager/features/#integrations. [Accessed 20 April 2020].

[32] Google, “Google Analytics Developer Guides - Working with the Measurement Protocol,” Google, [Online]. Available: https://developers.google.com/analytics/devguides/collection/protocol/v1/devguide. [Accessed 20 April 2020].

[33] M. Robins, “Server Side Tracking – What’s Old is New Agai,” Poplin Data, 4 November 2019. [Online]. Available: https://poplindata.com/data-collection/server-side-tracking/. [Accessed 20 April 2020].

[34] A. Mesbah and A. van Deursen, “A component- and push-based architectural style for ajax applications,” *Journal of Systems and Software,* vol. 81, no. 12, pp. 2194-2209, December 2008.

[35] C. Love, “Powerful Progressive Web App Features For Crazy App-Like Results,” Love2Dev, 21 February 2019. [Online]. Available: https://love2dev.com/blog/progressive-web-app-features/. [Accessed 21 April 2020].

[36] Heap, “Heap's home page,” Heap, [Online]. Available: https://heap.io/. [Accessed 21 April 2020].

[37] Heap, “Autocapture - Whitepaper,” Heap, [Online]. Available: https://heap.io/wp-content/uploads/2020/04/Heap_Autocapture_Whitepaper_V6.pdf. [Accessed 21 April 2020].

[38] Facebook, “Facebook for developers - Facebook analytics,” Facebook, [Online]. Available: https://developers.facebook.com/docs/analytics. [Accessed 20 April 2020].

[39] Facebook, “Facebook for developers - Implementing Facebook Analytics,” [Online]. Available: https://developers.facebook.com/docs/facebook-pixel/implementation. [Accessed 20 April 2020].

[40] SEMRush, “Features,” SEMRush, [Online]. Available: https://www.semrush.com/features/. [Accessed 20 April 2020].

[41] Snowplow, “Technology,” Snowplow, [Online]. Available: https://snowplowanalytics.com/products/technology/. [Accessed 20 April 2020].

[42] Snowplow, “How we compare,” Snowplow, [Online]. Available: https://snowplowanalytics.com/products/technology/. [Accessed 20 April 2020].

[43] U. Sivarajah, M. M. Kamal, Z. Irani and V. Weerakkody, “Critical analysis of Big Data challenges and analytical methods,” *Journal of Business Research,* vol. 70, pp. 263-286, January 2017.

[44] T. Dingsøyr, S. Nerur, V. Balijepally and N. B. Moe, “A decade of agile methodologies: Towards explaining agile software development,” *Journal of Systems and Software,* vol. 85, no. 6, pp. 1213-1221, June 2012.

[45] CollabNet, “13th Annual State of Agile Report,” 2019. [Online]. Available: https://stateofagile.com/#ufh-i-521251909-13th-annual-state-of-agile-report/473508. [Accessed 23 April 2020].

[46] S. Ahava, “#GTMTips: Restrict Custom HTML Tag Scope,” 18 October 2016. [Online]. Available: https://www.simoahava.com/gtm-tips/restrict-custom-html-tag-scope/. [Accessed 21 April 2020].

[47] S. Führinger and M. R Silva, “Tag Manager Help,” 20 July 2017. [Online]. Available: https://support.google.com/tagmanager/forum/AAAAnP_FwdIUraqi_IK65A/?hl=en. [Accessed 21 April 2020].

[48] Mixpanel, “Implementing Mixpanel with Google Tag Manager,” Mixpanel, [Online]. Available: https://help.mixpanel.com/hc/en-us/articles/360001355166-Implementing-Mixpanel-with-Google-Tag-Manager. [Accessed 21 April 2020].

[49] A. Pryka and D. Lacheta, “keen-tracking package,” Keen, [Online]. Available: https://www.npmjs.com/package/keen-tracking. [Accessed 27 April 2020].

[50] Adobe, “Analytics Tools Guide - Link tracking methodology,” Adobe, [Online]. Available: https://docs.adobe.com/content/help/en/analytics/analyze/activity-map/link-tracking/activitymap-link-tracking-methodology.html. [Accessed 23 April 2020].

[51] P. Walton, “autotrack,” [Online]. Available: https://www.npmjs.com/package/autotrack. [Accessed 23 April 2020].

[52] P. Walton, “Issues - Generic Actions - #117,” [Online]. Available: https://github.com/googleanalytics/autotrack/issues/117. [Accessed 23 April 2020].

[53] M. Moroshko, “react-event-tracker package,” [Online]. Available: https://www.npmjs.com/package/react-event-tracker. [Accessed 27 April 2020].

[54] Z. Martineau, “trakjs package,” [Online]. Available: https://www.npmjs.com/package/trak.js. [Accessed 27 April 2020].

[55] D. Wells, “Analytics Documentation,” [Online]. Available: https://getanalytics.io/. [Accessed 22 April 2020].

[56] w3schools.com, “HTML Element Reference,” w3schools.com, [Online]. Available: https://www.w3schools.com/TAGs/. [Accessed 23 April 2020].

[57] Html.com, “Html5,” Html.com, [Online]. Available: https://html.com/html5/. [Accessed 23 April 2020].

[58] Anthony, “Optimal Size and Spacing for Mobile Buttons,” Uxmovement.com, [Online]. Available: https://uxmovement.com/mobile/optimal-size-and-spacing-for-mobile-buttons/. [Accessed 27 April 2020].

[59] Microsoft, “TypeScript's Home page,” Microsoft, [Online]. Available: https://www.typescriptlang.org/index.html. [Accessed 22 April 2020].

[60] Google, “TypeScript configuration,” Angular, [Online]. Available: https://angular.io/guide/typescript-configuration. [Accessed 23 April 2020].

[61] Microsoft, “TypeScript's Documentation,” TypeScript, [Online]. Available: https://www.typescriptlang.org/docs/home.html. [Accessed 23 April 2020].

[62] Stack Overflow, “Developer Survey Results 2019,” Stack Overflow, [Online]. Available: https://insights.stackoverflow.com/survey/2019#most-loved-dreaded-and-wanted. [Accessed 23 April 2020].

[63] Stack Overflow, “Stack Overflow Trends,” Stack Overflow, [Online]. Available: https://insights.stackoverflow.com/trends?tags=typescript. [Accessed 23 April 2020].

[64] A. Deveria, “ECMAScript 5,” Can I Use, [Online]. Available: https://caniuse.com/#feat=es5. [Accessed 22 April 2020].

[65] D. A. Rauschmayer, Speaking JavaScript: An In-Depth Guide for Programmers, O'Reilly, 2014.

[66] Webpack, “Documentation - Concepts,” Webpack, [Online]. Available: https://webpack.js.org. [Accessed 23 April 2020].

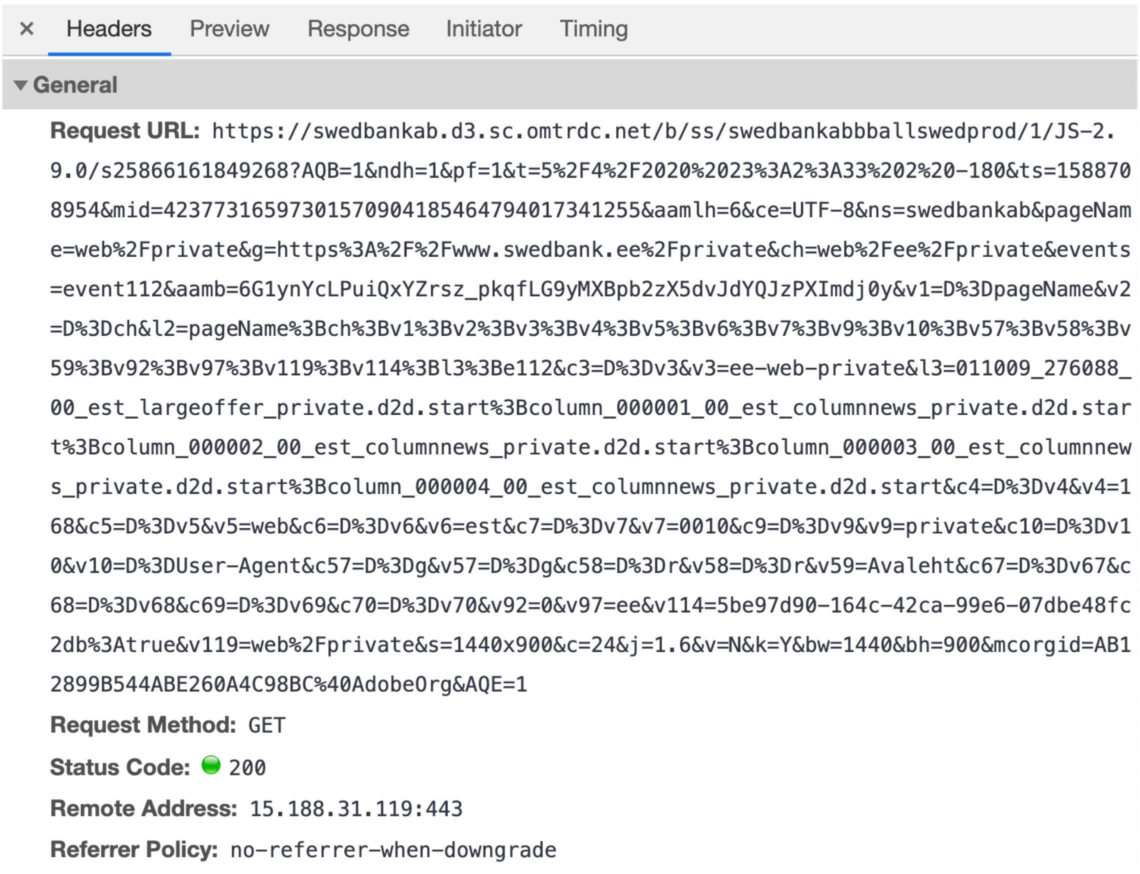
[67] M. Raba, “Repository of kollektor.js,” [Online]. Available: https://github.com/mikkraba/kollektor.js.

[68] M. Raba, “Package kollektor-js,” [Online]. Available: https://www.npmjs.com/package/kollektor-js.

[69] K. Lorey, “Regular expressions for social media profiles,” [Online]. Available: https://github.com/lorey/social-media-profiles-regexs. [Accessed 13 May 2020].

[70] Apple Inc, “UI Design Dos and Don’ts,” Apple Inc, [Online]. Available: https://developer.apple.com/design/tips/. [Accessed 27 April 2020].

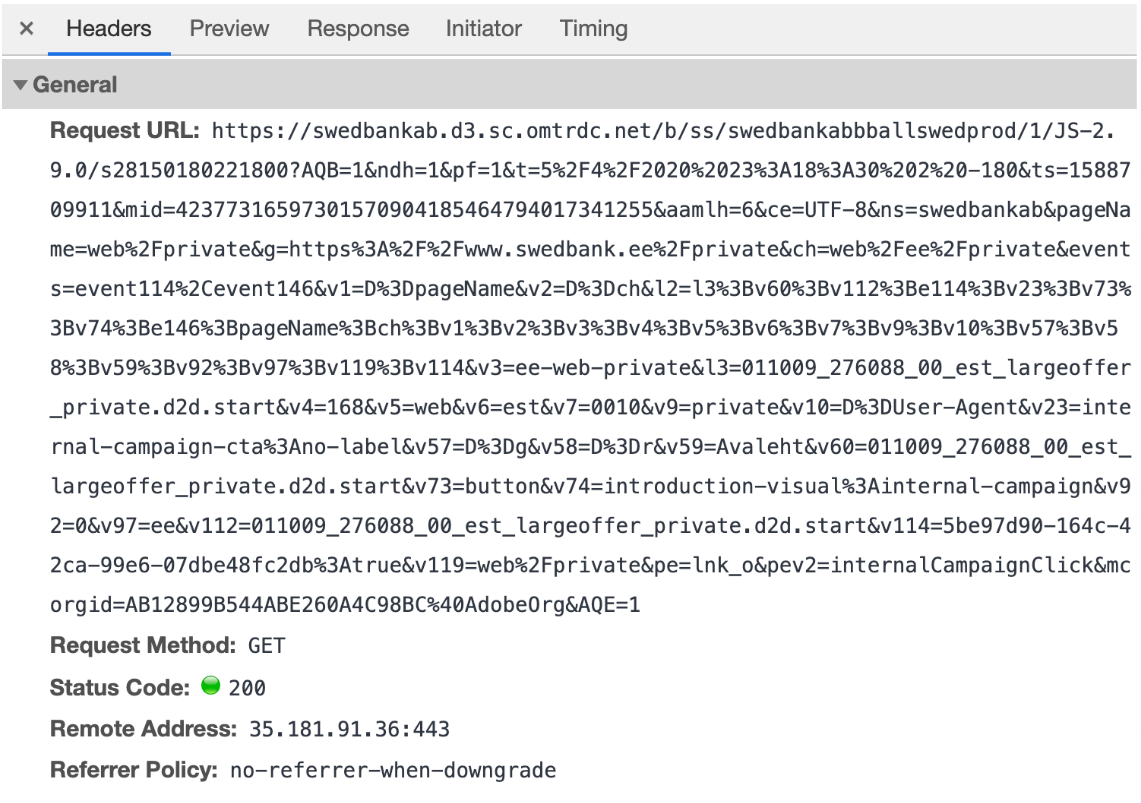
Appendix 1 – Request made to analytics collection server on Swedbank’s website on pageview



```
Request URL: https://swedbankab.d3.sc.omtrdc.net/b/ss/swedbankabbballswedprod/1/JS-2.
9.0/s25866161849268?AQB=1&ndh=1&pf=1&t=5%2F4%2F2020%2023%3A2%3A33%202%20-180&ts=158870
8954&mid=42377316597301570904185464794017341255&aamlh=6&ce=UTF-8&ns=swedbankab&pageNam
e=web%2Fprivate&g=https%3A%2F%2Fwww.swedbank.ee%2Fprivate&ch=web%2Fee%2Fprivate&events
=event112&aamb=6G1ynYcLPuiQxYZrsz_pkqfLG9yMXBpb2zX5dvJdYQJzPXImdj0y&v1=D%3DpageName&v2
=D%3Dch&l2=pageName%3Bch%3Bv1%3Bv2%3Bv3%3Bv4%3Bv5%3Bv6%3Bv7%3Bv9%3Bv10%3Bv57%3Bv58%3Bv
59%3Bv92%3Bv97%3Bv119%3Bv114%3Bl3%3Be112&c3=D%3Dv3&v3=ee-web-private&l3=011009_276088_
00_est_largeoffer_private.d2d.start%3Bcolumn_000001_00_est_columnnews_private.d2d.star
t%3Bcolumn_000002_00_est_columnnews_private.d2d.start%3Bcolumn_000003_00_est_columnnew
s_private.d2d.start%3Bcolumn_000004_00_est_columnnews_private.d2d.start&c4=D%3Dv4&v4=1
68&c5=D%3Dv5&v5=web&c6=D%3Dv6&v6=est&c7=D%3Dv7&v7=0010&c9=D%3Dv9&v9=private&c10=D%3Dv1
0&v10=D%3DUser-Agent&c57=D%3Dg&v57=D%3Dg&c58=D%3Dr&v58=D%3Dr&v59=Avaleht&c67=D%3Dv67&c
68=D%3Dv68&c69=D%3Dv69&c70=D%3Dv70&v92=0&v97=ee&v114=5be97d90-164c-42ca-99e6-07dbe48fc
2db%3Atrue&v119=web%2Fprivate&s=1440x900&c=24&j=1.6&v=N&k=Y&bw=1440&bh=900&mcorgid=AB1
2899B544ABE260A4C98BC%40AdobeOrg&AQE=1
Request Method: GET
Status Code: ● 200
Remote Address: 15.188.31.119:443
Referrer Policy: no-referrer-when-downgrade
```

Figure 22. Screen capture from Google Chrome developer tools. On image is presented a request made to analytics collection server on Swedbank.ee website on page load.

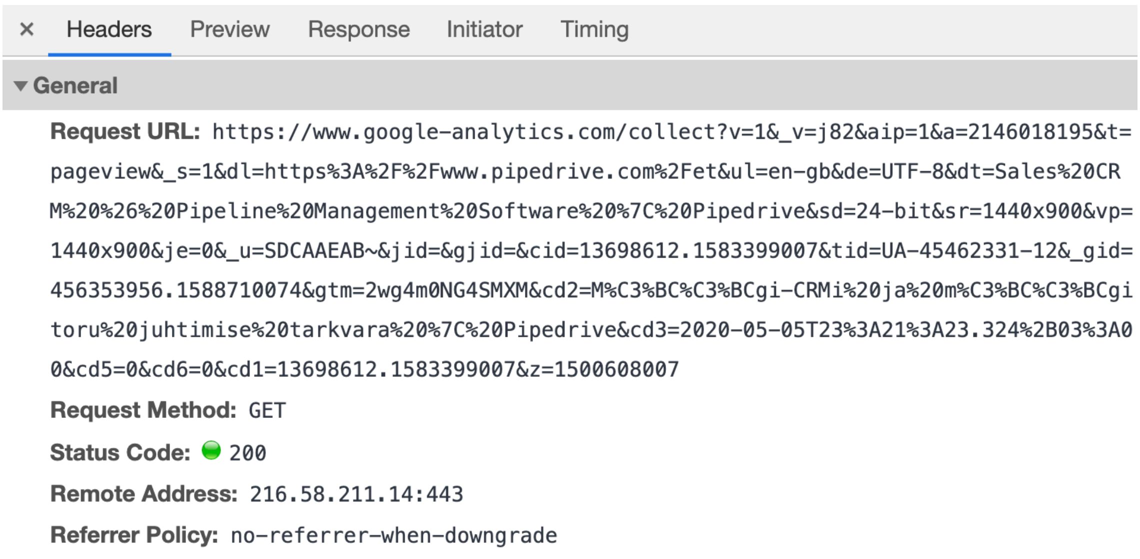
Appendix 2 – Requests made to analytics collection server on Swedbank’s website on click



```
Request URL: https://swedbankab.d3.sc.omtrdc.net/b/ss/swedbankabbballswedprod/1/JS-2.
9.0/s28150180221800?AQB=1&ndh=1&pf=1&t=5%2F4%2F2020%2023%3A18%3A30%202%20-180&ts=15887
09911&mid=42377316597301570904185464794017341255&aamlh=6&ce=UTF-8&ns=swedbankab&pageNa
me=web%2Fprivate&g=https%3A%2F%2Fwww.swedbank.ee%2Fprivate&ch=web%2Fee%2Fprivate&event
s=event114%2Cevent146&v1=D%3DpageName&v2=D%3Dch&l2=l3%3Bv60%3Bv112%3Be114%3Bv23%3Bv73%
3Bv74%3Be146%3BpageName%3Bch%3Bv1%3Bv2%3Bv3%3Bv4%3Bv5%3Bv6%3Bv7%3Bv9%3Bv10%3Bv57%3Bv5
8%3Bv59%3Bv92%3Bv97%3Bv119%3Bv114&v3=ee-web-private&l3=011009_276088_00_est_largeoffer
_private.d2d.start&v4=168&v5=web&v6=est&v7=0010&v9=private&v10=D%3DUser-Agent&v23=inte
rnal-campaign-cta%3Ano-label&v57=D%3Dg&v58=D%3Dr&v59=Avaleht&v60=011009_276088_00_est_
largeoffer_private.d2d.start&v73=button&v74=introduction-visual%3Ainternal-campaign&v9
2=0&v97=ee&v112=011009_276088_00_est_largeoffer_private.d2d.start&v114=5be97d90-164c-4
2ca-99e6-07dbe48fc2db%3Atrue&v119=web%2Fprivate&pe=lnk_o&pev2=internalCampaignClick&mc
orgid=AB12899B544ABE260A4C98BC%40AdobeOrg&AQE=1
Request Method: GET
Status Code: ● 200
Remote Address: 35.181.91.36:443
Referrer Policy: no-referrer-when-downgrade
```

Figure 23. Screen capture from Google Chrome developer tools. On image is presented a request made to analytics collection server on Swedbank.ee website when a banner button is clicked.

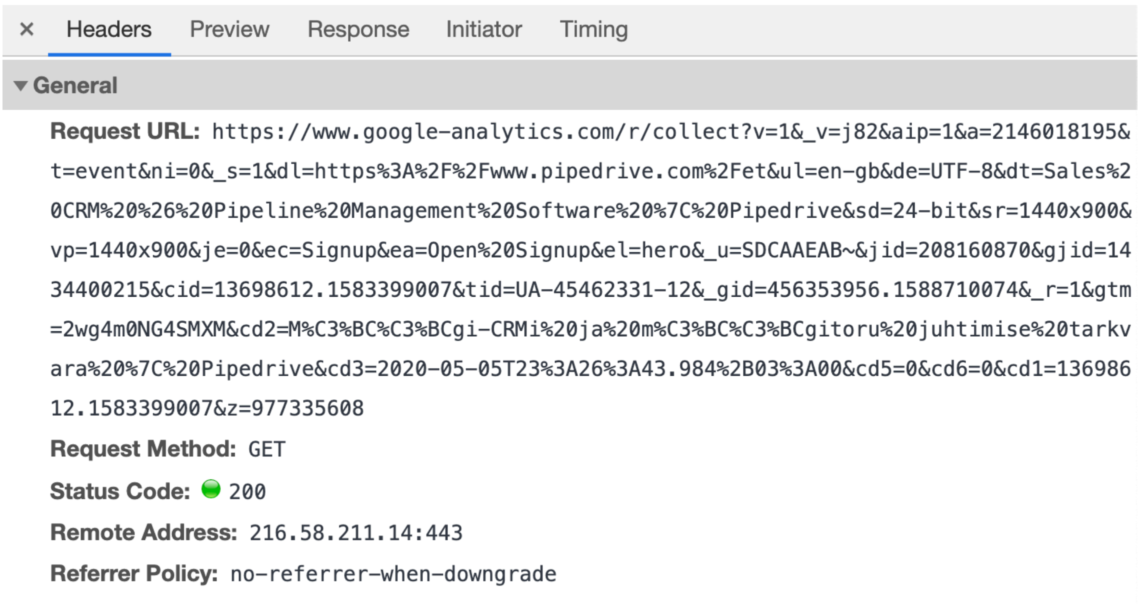
Appendix 3 – Requests made to analytics collection server on Pipedrive’s website pageview



```
Request URL: https://www.google-analytics.com/collect?v=1&_v=j82&aip=1&a=2146018195&t=
pageview&_s=1&dl=https%3A%2F%2Fwww.pipedrive.com%2Fet&ul=en-gb&de=UTF-8&dt=Sales%20CR
M%20%26%20Pipeline%20Management%20Software%20%7C%20Pipedrive&sd=24-bit&sr=1440x900&vp=
1440x900&je=0&_u=SDCAAEAB~&jid=&gjid=&cid=13698612.1583399007&tid=UA-45462331-12&_gid=
456353956.1588710074&gtm=2wg4m0NG4SMXM&cd2=M%C3%BC%C3%BCgi-CRMi%20ja%20m%C3%BC%C3%BCgi
toru%20juhtimise%20tarkvara%20%7C%20Pipedrive&cd3=2020-05-05T23%3A21%3A23.324%2B03%3A0
0&cd5=0&cd6=0&cd1=13698612.1583399007&z=1500608007
Request Method: GET
Status Code: ● 200
Remote Address: 216.58.211.14:443
Referrer Policy: no-referrer-when-downgrade
```

Figure 24. Screen capture from Google Chrome developer tools. On image is presented a request made to analytics collection server on Pipedrive.com website when page is loaded.

Appendix 4 – Request made to analytics collection server on Pipedrive’s website on click



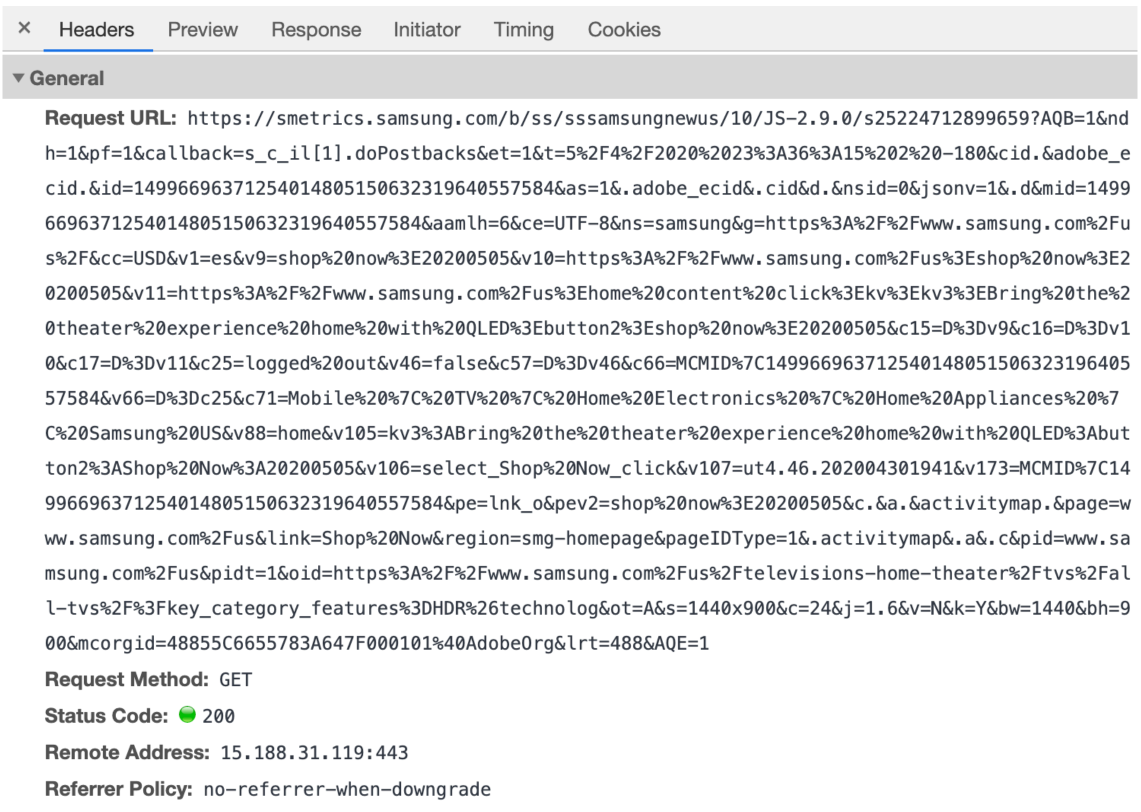
× Headers Preview Response Initiator Timing

▾ General

```
Request URL: https://www.google-analytics.com/r/collect?v=1&_v=j82&aip=1&a=2146018195&
t=event&ni=0&_s=1&dl=https%3A%2F%2Fwww.pipedrive.com%2Fet&ul=en-gb&de=UTF-8&dt=Sales%2
0CRM%20%26%20Pipeline%20Management%20Software%20%7C%20Pipedrive&sd=24-bit&sr=1440x900&
vp=1440x900&je=0&ec=Signup&ea=Open%20Signup&el=hero&_u=SDCAAEAB~&jid=208160870&gjid=14
34400215&cid=13698612.1583399007&tid=UA-45462331-12&_gid=456353956.1588710074&_r=1&gtm
=2wg4m0NG4SMXM&cd2=M%C3%BC%C3%BCgi-CRMi%20ja%20m%C3%BC%C3%BCgitoru%20juhtimise%20tarkv
ara%20%7C%20Pipedrive&cd3=2020-05-05T23%3A26%3A43.984%2B03%3A00&cd5=0&cd6=0&cd1=136986
12.1583399007&z=977335608
Request Method: GET
Status Code: ● 200
Remote Address: 216.58.211.14:443
Referrer Policy: no-referrer-when-downgrade
```

Figure 25. Screen capture from Google Chrome developer tools. On image is presented a request made to analytics collection server on Pipedrive.com website when a banner button is clicked.

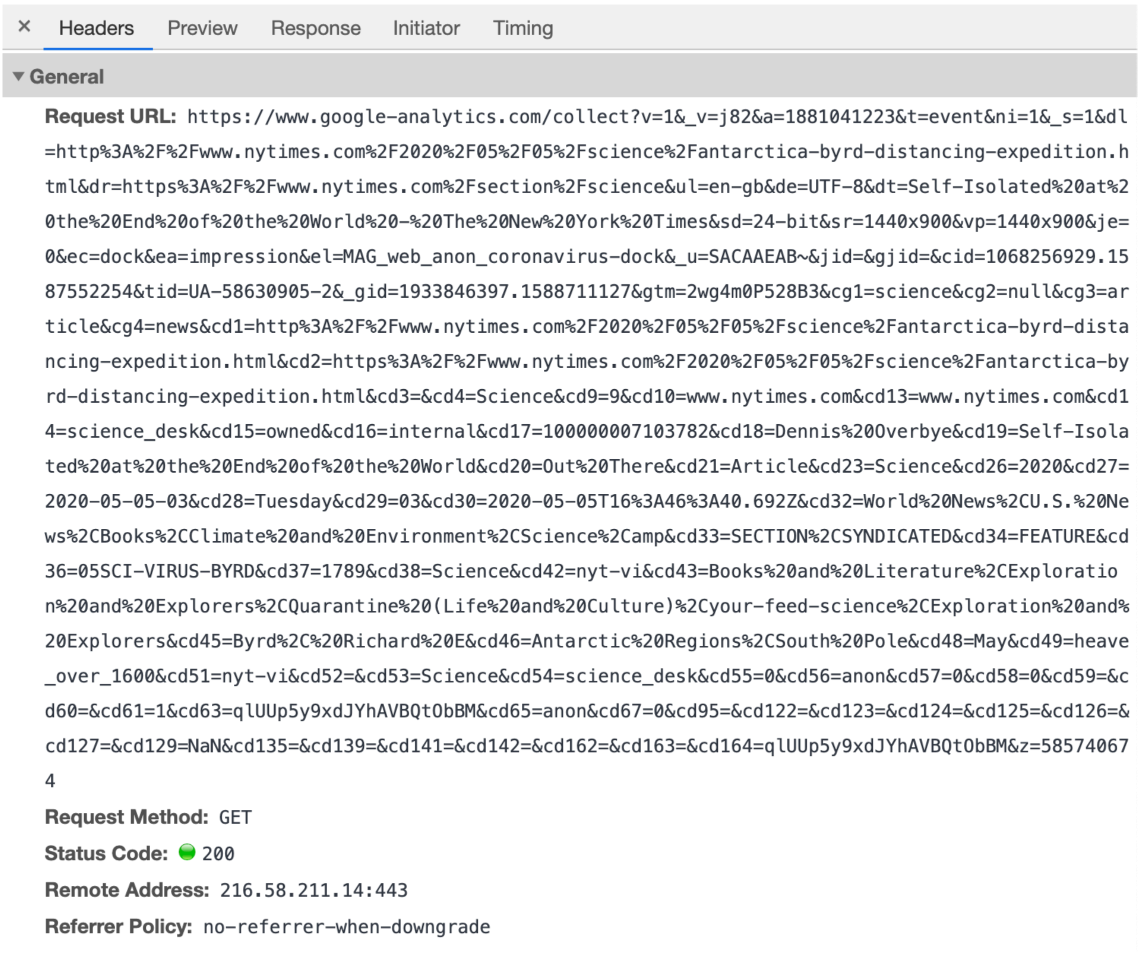
Appendix 5 – Request made to analytics collection server on Samsung's website on click



```
Request URL: https://smetrics.samsung.com/b/ss/sssamsungnewus/10/JS-2.9.0/s25224712899659?AQB=1&nd
h=1&pf=1&callback=s_c_il[1].doPostbacks&et=1&t=5%2F4%2F2020%2023%3A36%3A15%202%20-180&cid.&adobe_e
cid.&id=14996696371254014805150632319640557584&as=1&.adobe_ecid&.cid&d.&nsid=0&jsonv=1&.d&mid=1499
6696371254014805150632319640557584&aamlh=6&ce=UTF-8&ns=samsung&g=https%3A%2F%2Fwww.samsung.com%2Fu
s%2F&cc=USD&v1=es&v9=shop%20now%3E20200505&v10=https%3A%2F%2Fwww.samsung.com%2Fus%3Eshop%20now%3E2
0200505&v11=https%3A%2F%2Fwww.samsung.com%2Fus%3Ehome%20content%20click%3Ekv%3Ekv3%3EBring%20the%2
0theater%20experience%20home%20with%20QLED%3Ebutton2%3Eshop%20now%3E20200505&c15=D%3Dv9&c16=D%3Dv1
0&c17=D%3Dv11&c25=logged%20out&v46=false&c57=D%3Dv46&c66=MCMID%7C149966963712540148051506323196405
57584&v66=D%3Dc25&c71=Mobile%20%7C%20TV%20%7C%20Home%20Electronics%20%7C%20Home%20Appliances%20%7
C%20Samsung%20US&v88=home&v105=kv3%3ABring%20the%20theater%20experience%20home%20with%20QLED%3Abut
ton2%3AShop%20Now%3A20200505&v106=select_Shop%20Now_click&v107=ut4.46.202004301941&v173=MCMID%7C14
996696371254014805150632319640557584&pe=lnk_o&pev2=shop%20now%3E20200505&c.&a.&activitymap.&page=w
ww.samsung.com%2Fus&link=Shop%20Now&region=smg-homepage&pageIDType=1&.activitymap&.a&.c&pid=www.sa
msung.com%2Fus&pidt=1&oid=https%3A%2F%2Fwww.samsung.com%2Fus%2Ftelevisions-home-theater%2Ftvs%2Fal
l-tvs%2F%3Fkey_category_features%3DHDR%26technolog&ot=A&s=1440x900&c=24&j=1.6&v=N&k=Y&bw=1440&bh=9
00&mcorgid=48855C6655783A647F000101%40AdobeOrg&lrt=488&AQE=1
Request Method: GET
Status Code: ● 200
Remote Address: 15.188.31.119:443
Referrer Policy: no-referrer-when-downgrade
```

Figure 26. Screen capture from Google Chrome developer tools. On image is presented a request made to analytics collection server on Samsung.com website when a banner button is clicked.

Appendix 6 – Request made to analytics collection server on article view on The New York Times website



```
Request URL: https://www.google-analytics.com/collect?v=1&_v=j82&a=1881041223&t=event&ni=1&_s=1&dl
=http%3A%2F%2Fwww.nytimes.com%2F2020%2F05%2F05%2Fscience%2Fantarctica-byrd-distancing-expedition.h
tml&dr=https%3A%2F%2Fwww.nytimes.com%2Fsection%2Fscience&ul=en-gb&de=UTF-8&dt=Self-Isolated%20at%2
0the%20End%20of%20the%20World%20-%20The%20New%20York%20Times&sd=24-bit&sr=1440x900&vp=1440x900&je=
0&ec=dock&ea=impression&el=MAG_web_anon_coronavirus-dock&_u=SACAAEAB~&jid=&gjid=&cid=1068256929.15
87552254&tid=UA-58630905-2&_gid=1933846397.1588711127&gtm=2wg4m0P528B3&cg1=science&cg2=null&cg3=ar
ticle&cg4=news&cd1=http%3A%2F%2Fwww.nytimes.com%2F2020%2F05%2F05%2Fscience%2Fantarctica-byrd-dista
ncing-expedition.html&cd2=https%3A%2F%2Fwww.nytimes.com%2F2020%2F05%2F05%2Fscience%2Fantarctica-by
rd-distancing-expedition.html&cd3=&cd4=Science&cd9=9&cd10=www.nytimes.com&cd13=www.nytimes.com&cd1
4=science_desk&cd15=owned&cd16=internal&cd17=100000007103782&cd18=Dennis%20Overbye&cd19=Self-Isola
ted%20at%20the%20End%20of%20the%20World&cd20=Out%20There&cd21=Article&cd23=Science&cd26=2020&cd27=
2020-05-05-03&cd28=Tuesday&cd29=03&cd30=2020-05-05T16%3A46%3A40.692Z&cd32=World%20News%2CU.S.%20Ne
ws%2CBooks%2CClimate%20and%20Environment%2CScience%2Camp&cd33=SECTION%2CSYNDICATED&cd34=FEATURE&cd
36=05SCI-VIRUS-BYRD&cd37=1789&cd38=Science&cd42=nyt-vi&cd43=Books%20and%20Literature%2CExploratio
n%20and%20Explorers%2CQuarantine%20(Life%20and%20Culture)%2Cyour-feed-science%2CExploration%20and%
20Explorers&cd45=Byrd%2C%20Richard%20E&cd46=Antarctic%20Regions%2CSouth%20Pole&cd48=May&cd49=heave
_over_1600&cd51=nyt-vi&cd52=&cd53=Science&cd54=science_desk&cd55=0&cd56=anon&cd57=0&cd58=0&cd59=&c
d60=&cd61=1&cd63=qlUUp5y9xdJYhAVBQtObBM&cd65=anon&cd67=0&cd95=&cd122=&cd123=&cd124=&cd125=&cd126=&
cd127=&cd129=NaN&cd135=&cd139=&cd141=&cd142=&cd162=&cd163=&cd164=qlUUp5y9xdJYhAVBQtObBM&z=58574067
4
Request Method: GET
Status Code: ● 200
Remote Address: 216.58.211.14:443
Referrer Policy: no-referrer-when-downgrade
```

Figure 27. Screen capture from Google Chrome developer tools. On image is presented a request made to analytics collection server on The New York Times website when an article is viewed.