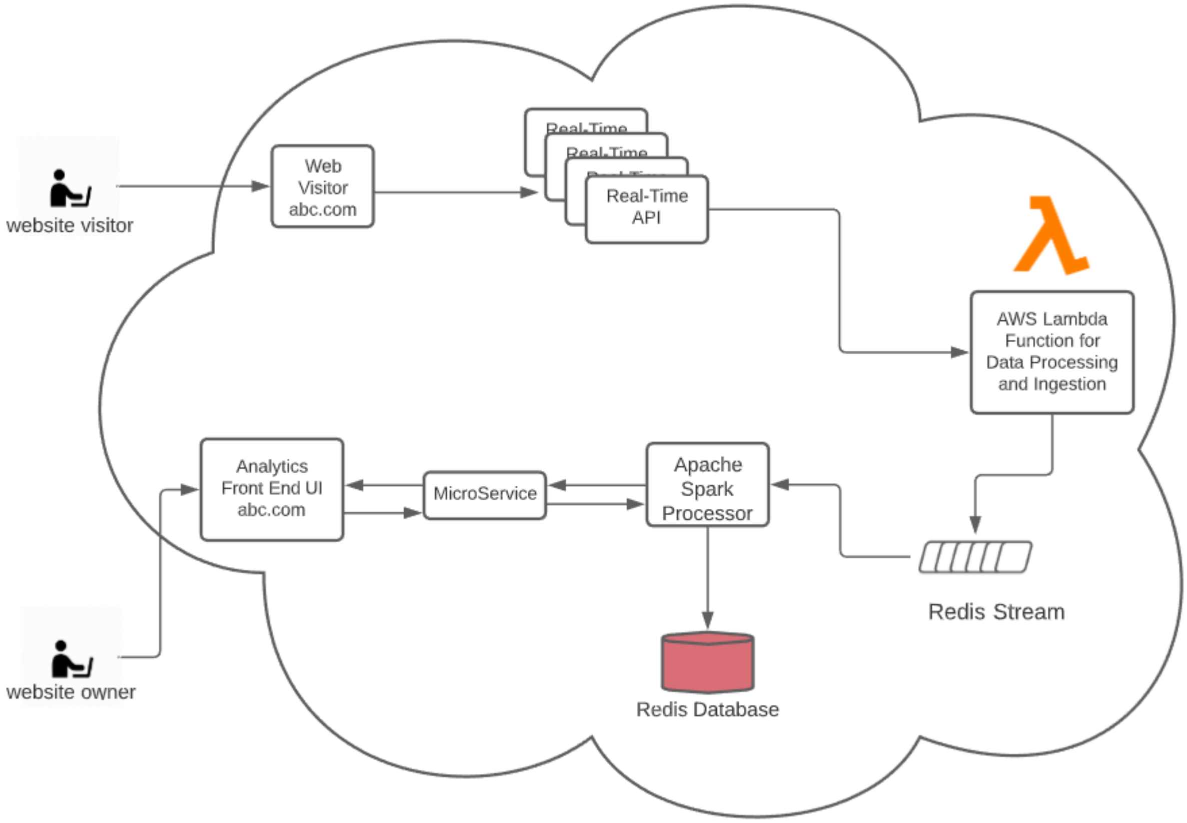
Google Analytics-Like Backend Design

The design of our google-analytics like backend system is a cloud-based solution. The beauty of this is that it is entirely server less, and as such, we don't need to worry about provisioning hardware or any other infrastructure related tasks. Each component can be auto tweaked for greater capacity or scalability.



The first step involves a web user visiting a website, let's call it abc.com. We expect the owner of abc.com to have installed a JavaScript code snippet on their site to track these web visits, among other events they wish to monitor. The code snippet installed on the site calls some sort of a real time API gateway, with information about the underlying event added into the data being sent (such as time of event trigger, browser used, is a returning visitor, website click, button click, etc.). This API gateway then calls a lambda function that we have setup. The purpose of this lambda function is to process the events and categorize them accordingly, and ultimately append them to a redis stream which models a log data structure.

Redis Streams was chosen for this design for 2 primary reasons:

- This built in data structure stream in redis can easily handler upwards of one million writes and reads each second. Therefore, the requirement of handling large volumes of operations is handled.

- Redis streams also orders data automatically by time, therefore the requirement to reprocess historical data in case of bugs in the processing logic can also be done.

The consumer that consumes the data from the redis streams is a structured streaming API in Apache Spark. There already exists a redis and spark joint library, so this integration is easily configured. Apache Spark will continually consume these redis streams, perform some processing, and store the results back in a redis NoSQL key-map database. This key-map database offers ultra-fast querying capabilities also performed by Apache Spark. Lastly, we have a micro service that sits right in front of our website owner's Google Analytic-like UI platform, which is called on demand. For example, when the website owner wants to retrieve analytics for the number of abandoned cart events on his site, the micro service would call our Apache Spark application, which acts as a SQL interface to our Redis data, and query the information on the fly.