ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

Building a Real Time Analytics Reporting System Dashboard using Google superProxy

Pallavi Taneja

B.Tech Student, Dept. of Information Technology, Indira Gandhi Delhi Technical University for Women, Delhi

ABSTRACT: A Real Time Analytics System monitors and reports all the activities of the system happening the very moment. Besides a Core Reporting System, there is a need to build a system that is congruous with real-time for a wide variety of reasons. Albeit, A Core-Reporting system provides reports of various different metrics along different dimensions, but the reports are statistical averages, total, variance and deviations etc. over a given period of time. This raises a very crucial need for a Real Time System to vouch for the accuracy of the core-reporting data and also provide an intelligent analytical system. Real time analytics reports can more accurately provide a medium to connect the developers, Business Analysts etc. to the users and help them come up with an efficient analysis of the content and pave a right direction towards more advancement of the system that would be beneficial to both the ends. In this paper, an approach to design a real time reporting dashboard has been proposed using powerful Google Technologies.
The developed system will also be maintained and managed through a web server application that would be used to create API queries, clear request errors etc.

KEYWORDS: Real-Time Analytics; Core Reporting; Metrics; Dimensions; Google Technologies

I. INTRODUCTION

A Real Time Analytics System can be propounded as a dynamic processing system that instantly processes the hits and generates the prompt reports of the dynamic data. Here, the word 'Dynamic' has a very profound meaning. Dynamic means to query the data present in Random Access Memory rather than the data stored in physical storage that serves the core reporting. What actually raises a Real-Time Analytics desideratum? The answer to this involves a wide variety of needs. Real Time can form a primordial basis for the Core Reporting by being a debugger. Real time can actually affirm the changes made to the system are congruous or not. There is no sense in considering the statistical averages and means when the changes to system are not triggered properly. Real time reports can play a very crucial role in attesting the performance of a newly developed system and its inchoate features[8]. It can also help one to ameliorate the quality of content of the system and make updates according to the reports. Moreover, It will also contribute to a better analysis of the referrals for the website. Real time analytics can provide various active reports owing to an early and efficient development of a system. For instance, It can provide an active user count, active locations, active browsers, active devices, active pages etc. Now, It would be a cinch for one to analyse these active reports and resolve compatibility, accuracy, portability, efficiency issues pertaining to the website or app.

A. *Core Reporting and its Google Analytics API:*

A Core-Reporting takes a few hours to process the hits before they are ready to be used by variegated metrics and dimensions. But after processing a lot of dimensions are available in core reporting.

A Core Reporting API reports the data gathered by Google Analytics tracker code after performing a statistical analysis on the collected data. It then provides a wide gamut of obtaining the reports along different dimensions for different metrics. Each API request establishes a relation between the User obtained through authorization credentials and Profile ID obtained through Google Analytics Configuration hierarchy. An API query comprises of metrics and dimensions apart from profile ID. A metric is a measurement of an activity of a user whereas a dimension involves a categorization of the metrics for different criteria. For instance to get the pageviews for users belonging to different countries, the metric would be 'pageviews' and the dimension would be 'country'. After the query is made to the API, the API returns the data in the form of a table with headers specifying the data types and the column names.

Apart from metrics and dimensions[1] there is a great flexibility of specifying the sorting order of dimensions & metrics, apply filters to dimensions & metrics, specify the start-date and end date and many more. Quota Policy is

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

immanent to all the Google APIs so as to maintain a fair distribution of system resources. The specified quota limit is 10 QPS and 50,000 requests per day per project[1]. A Core Reporting system can be scaled up to constructing Custom Dimensions in Google Analytics and query them through the Core Reporting API.

B. *Real Time and its Google Analytics API:*

A Real-Time Analytics processes and continuously updates the hits and makes them readily available to be used by all dimensions and metrics. For any authenticated and authorized user, it allows you to query the real time data and frame reports in order to monitor the activities of your property in real time.

There is great variety of dimensions and metrics in the Real- Time API[2] as well though the range is lesser than that of Core Reporting. For instance, this API allows you to query an active user count, active users for different locations, browsers, devices, pages etc. as the dimensions. The Quota Policy for this API is same as that of other APIs of Google. The Real Time API so far does not take into account the concept of Custom dimensions thereby reducing a flexibilities in the usage of API.

There are a wide number of ways in which we can generate and use real-time reports. We can use them analyse the users' response to the changes made to the web application, monitor the real time traffic sources on the website through direct and indirect sources, verify and validate the working of the analytics tracking code and vouch for the changes made in the tracking code. There are a few limits to the real time API[7]. Changes to filtered data pertaining to a view may take up a few hours to get reflected in the reports. Therefore, It has been advised to use unfiltered queries to get the real-time responses so far. Batching of the data from sources like mobile can also cause delays of few minutes in the response. Another limitation is Campaign Attribution. This phenomenon occurs due to UserID Analytics or Universal Analytics by virtue of which it takes into consideration a single session for a particular UserID. Due to this, during a particular session a user is counted under a direct referral and stopped being counted under a specific campaign or indirect referral.

II. REAL TIME DASHBOARD OR WIDGET WITH GOOGLE ANALYTICS SUPERPROXY

A. *Google Analytics superProxy:*

Google Analytics superProxy provides an impressive way of sharing the Google Analytics data publicly[3]. It providesthe data in different formats: CSV, TSV, JSON-Table formatand JSON. This data once available can be used to constructa real time dashboard or widget by powering the charts withthe data in different formats. The data is obtained after anAPI query is made to the Google analytics superProxy. Ithas a very distinguished feature of specifying relative datesthat allows one to comprehend the when a particular querywas requested the last like 3days, 5hrs., 45 mins ago. Itautomatically refreshes the data of the reports on the basis ofthe refresh interval specified by the user for a particular query.To speed up the response time, it supports data caching which also contributes to the increased efficiency of the system. Once the Application is hosted, superProxy allows admin of the application, just like any other admin of a web server to control and manage the queries. It provides a great flexibility of supporting Multiple-Users each with different queries. It also allows admin to refresh a particular query on ad-hoc basis apart from automatic refreshing.

superProxy allows you to configure error logging where in the errors are logged for each API query in superProxy. There is a limit of 10 error responses exceeding which might lead to a suspension of query responses from the superProxy until the errors are fixed. During this period the superProxy returns the last error-free response on being requested. Some of the common error responses are unable to fetch URL, Quota Limit Exceed, Authorisation errors, python errors etc. It also provides a facility to change the time zone and also modify the query response URL by enabling "anonymised" responses.

B. *OAuth 2.0:*

OAuth 2.0 protocol of Google[4] is recommended for authentication and authorization by the Google APIs. OAuth 2.0 establishes an interaction between the Google Account and
Google APIs. In order to get an access token, a ClientID has to be created on Google Developer's Console. This ClientID will help the application to request the Google Authorization for getting access tokens. Now, the client will handle the response from the authorization server, extract an access token from it, and send the token to Google API intended to be used. Once, a token is obtained, it is saved and whole process of token management and token

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

refreshment begins. The token will be used for authentication to make requests to the API and it would give a response in return, with the requested data.

The Scenario and Working of the OAuth2[4] can be easily explained using a sequential diagram in Fig. 1:

1) The web server application will request a token from the Google Server through a query URL.
2) When the Google Server receives the request, It will extract the information to determine the type of access request being made. It will check the authenticity of the user and the user content after which it will return an authorization code.
3) The application will use the authorization code response to get an access token and a refresh token from the Google server.
4) Now, the Application will save both the tokens and use the access token to query the Google APIs. After the access token gets expired, the application will use the refresh token to get a new access token from Google servers.

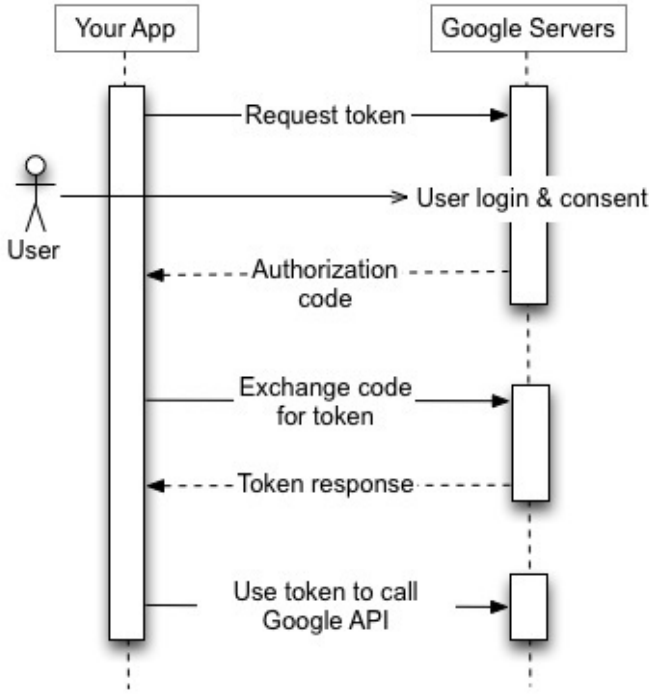


Fig. 1. Sequential Diagram depicting the interaction By web-server application and Google server to get an access token
Source: https://developers.google.com/identity/protocols/OAuth2

C. *Dimensions and Metrics of Real Time API*

	Dimensions	Metrics
Users	rt:userType	rt:activeUsers
Time	rt:minutesAgo	
Traffic Sources	rt:referralPath, rt:campaign, rt:source, rt:medium	
Goal Conversions	rt:goalID	rt:goalXXvalue, rt:goalAll Value
Platform/Device	rt:browser, rt:deviceCategory, rt:browserVersion	
Geo	rt:country, rt:city, rt:region, rt:longitude, rt:lattitude	
Page Tracking	rt:pagePath, rt:pageTitle	rt:pageviews
App Tracking	rt:appName, rt:appVersion, rt:screenName	rt:screenViews
Event Tracking	rt:eventAction, rt:eventCategory, rt:eventLabel	rt:totalEvents

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

D. *Creating End-point URLs with superProxy*

Once we get the data from the access token, there is a need to parse the data so as to extract the information to be used and displayed. There is a need to transform the data and convert it to a format that can be used to power the dashboard charts. We want the represented data on the dashboard to be accessed by all those who have permissions to access the web application. The script for building dashboard can be kept and managed on the server side by keeping the query responses on the database or data store on the server side. Automated and regular refreshing of the data is another need. Also, we need caching of data for fast responses and efficiency. All these complexities will now be managed by Google Analytics superProxy instead of webserver after the first end point URL query has been created.
Refer to Fig.2 that elucidates the entire scenario.
Configuration of Google Analytics superProxy correctly with the web application is seminal job. It involves a series of caveats. First of all, Authentication and Authorization of Google Analytics superProxy is required. This implies that the web application has been granted access and authorization to access the Google analytics data. The web interface for authorization is provided by the superProxy. Once, the authentication flow is completed, the application will get the access tokens and superProxy will automatically manage the entire process of token refreshing. This token will now enable our web application to communicate directly with Google Analytics server and Google APIs via a superProxy interface. The management of responses from Google Analytics through the token will be catered by the superProxy itself. Once, it receives the data it will automatically store the data response on a data store. Not only this, the entire management of automatic refreshing of the data in the data store will be handled by the superProxy.
superProxy will make the raw data table containing Google Analytics data available in four formats by doing some transformations pertaining to each format. The available formats are CSV, TSV, Data Table (JSON), Data Table (JSON string). This makes it extraordinarily extensible. While creating a new query in superProxy it allows you to set the automatic refresh interval, start-date and end-date. After the query is created, for each query it provides an interface to manage them by manually clearing errors, manually refreshing and access the data table response in different formats.

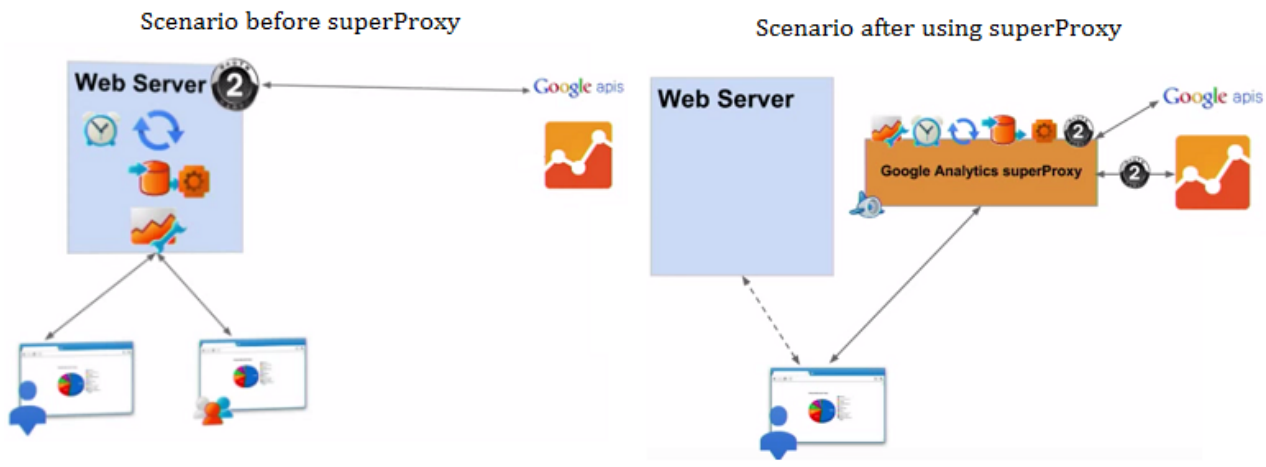


Fig. 2. Handling of complexities by Google Analytics superProxy as compared to the scenario where there is no superProxy
Source: https://developers.google.com/analytics/solutions/google-analyticssuper-proxy

E. Hosting the Web Server Application with App Engine

Google App Engine[5] provides a local deployment environment for the application. It facilitates the application with a platform to host the application. The application is now up and runs on the infrastructure of Google. It creates an ease of access for the admin on the Google cloud by constructing a URL for the deployed application "yourapp.appspot.com/admin". Google's App Engine supports four runtime environments JAVA, Python, PHP and Go. In order to deploy superProxy, the best runtime environment is Python 2.7.

The vantage-point for hosting the application on the app engine is that one can now easily scale the various data resources[6] through the app engine interface. It also monitors different aspects of the hosted web application for the utilization of datastore resources, frontend instance hours, logs data storage, outgoing bandwidth etc. Apart from this, it also serves the admin by proving request counts for each query, quota information etc. It has a very comprehensive

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

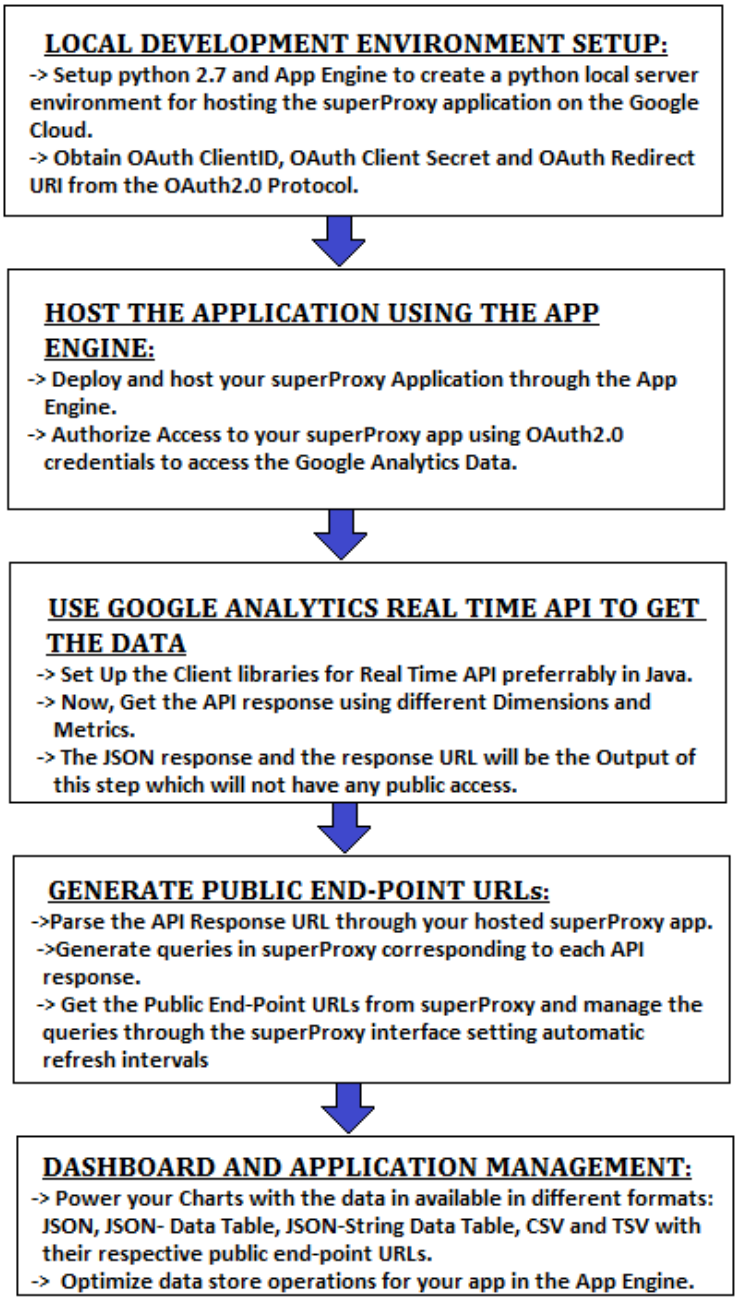
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

Dashboard that manifests the entire application status through a summary graph of counts/sec, data resources statuses, Instances, Current Load, Server Errors and Client Errors.

App Engine reports play a very crucial role in the optimization of the data resources for the hosted web application. Index file in superProxy is configurable to handle the datastore read and datastore write operations. This gives the admin a tremendous opportunity to optimize the datastore read and write operations so as to maximize the use quotas and end up saving free quotas avoiding inordinate and disproportionate wastage.

III. PROPOSED ALGORITHM



ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

IV. RESULTS

Figure No.	Response Data Format	Dimension	Metrics	Automatic Refresh Interval
Fig.3	JSON (Java Service Object Notation)	rt:browser	rt:activeUsers	90s
Fig.4	Data Table (JSON Response)	rt:browser	rt:activeUsers	90s
Fig.5	Data Table (JSON-String)	rt:city	rt:activeUsers	90s

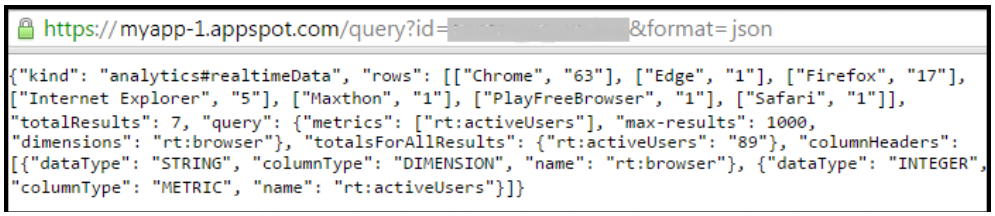


Fig.3: JSON(Java Service Object Notation) for real time Dimension:"rt:browser" and metric:"rt:activeUsers"

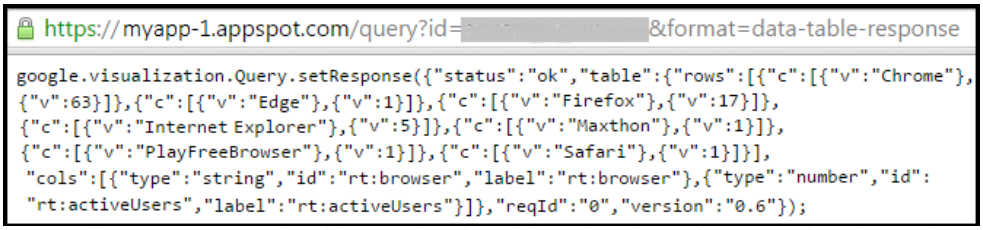


Fig. 4. Data-Table (JSON) (Java Service Object Notation) for real time Dimension:"rt:browser" and metric:"rt:activeUsers"

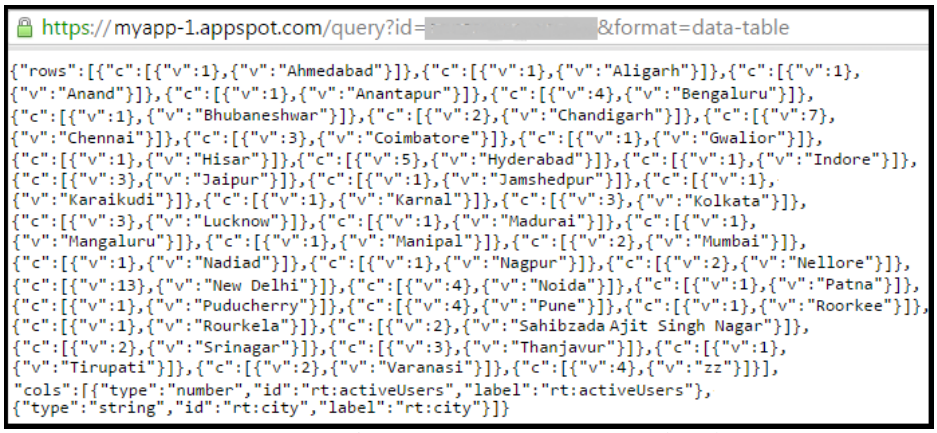


Fig. 5. Data-Table (JSON-String) for real time Dimension:"rt:city" and metric:"rt:activeUsers"

V. IMPLEMENTATION

The Fig.6 elucidates a Dashboard prototype of Real Time Reporting system for serbonline.in (Science and Engineering Research Board portal maintained by E-Governance Department of Centre for Development of Advanced Computing - Noida) developed by the author. Dimensions and Metric used for each chart is as follows:

Dashboard Chart Name	Dimension	Metric	Refresh Interval
Active User Count		rt:activeUsers	90s
Active Browsers	rt:browser	rt:activeUsers	90s
Active Devices	rt:deviceCategory	rt:activeUsers	90s
Active Locations	rt:city	rt:activeUsers	90s

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

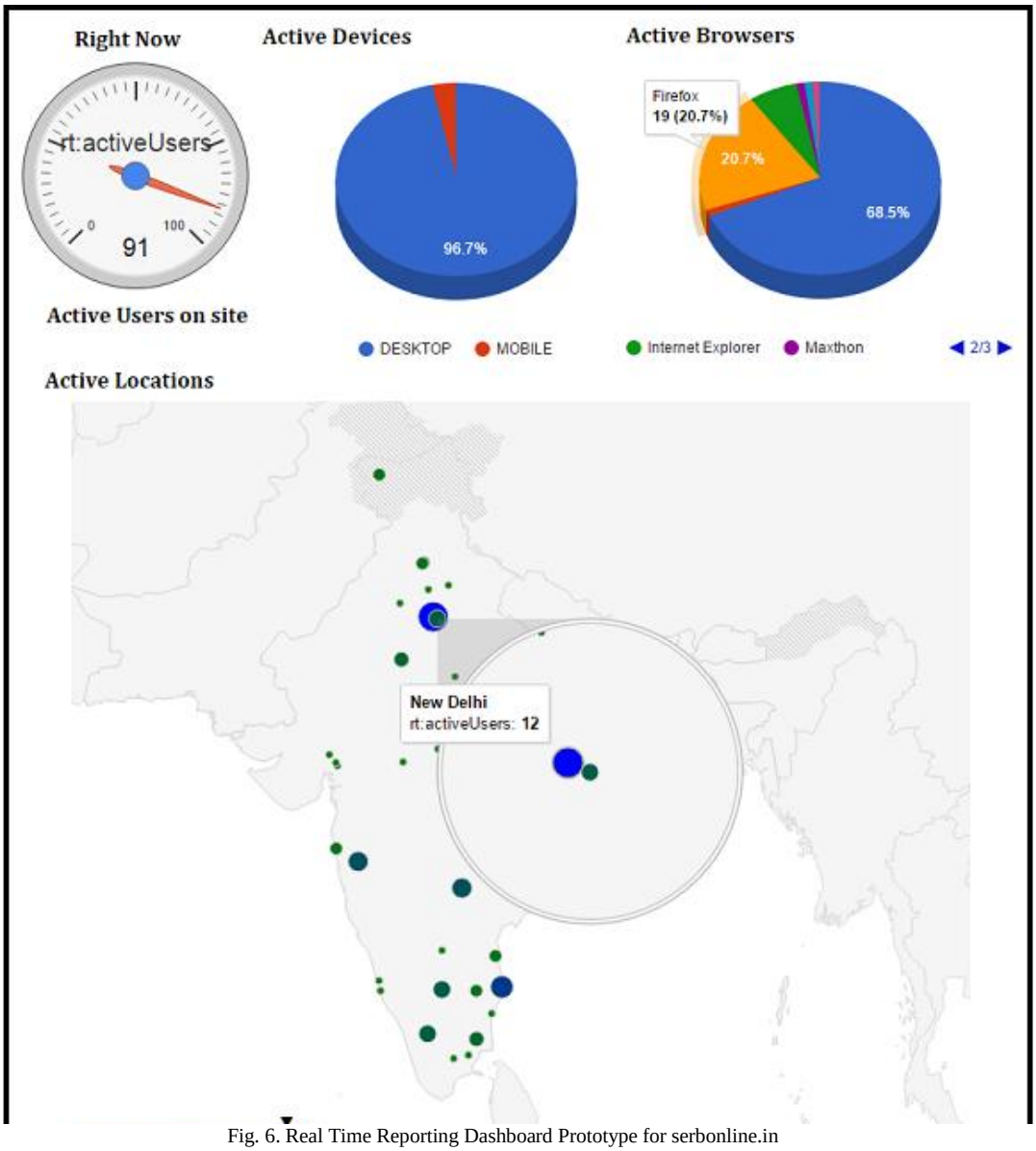


Fig. 6. Real Time Reporting Dashboard Prototype for serbonline.in

APP Engine graphs and reports are shown in Fig.7:

Summary Count/sec Graph	Resource Usage Table	Current Load	Server Errors

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

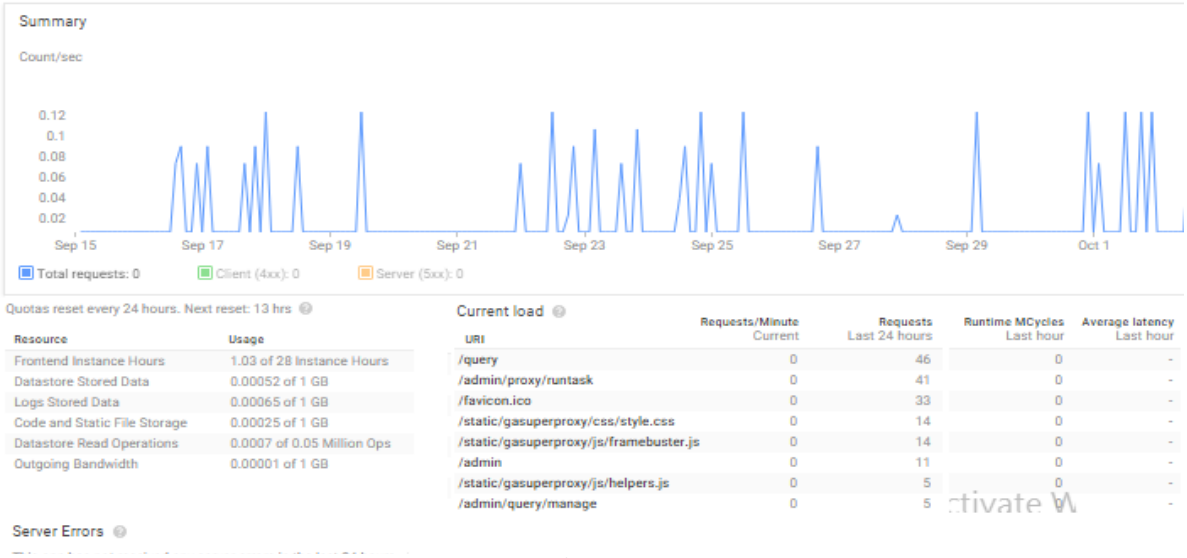


Fig. 7. App Engine reports for the webserver application

VI. CONCLUSION

The Upshot of the results of real time reporting project is that with the profound understanding of the concepts pertaining to Google analytics real-time API and other related technologies can help you to construct an efficient real-time dashboard or widget for any of your web property. The responses in different formats obtained from superProxy can be used to power the dashboard charts. The end point URL queries can be created and managed by the admin through the app-spot superProxy interface. Moreover, the entire server application can be monitored for data-store resources at the backend for handling the API query requests, API responses through the app-engine.

VII. FUTURE SCOPE

Real Time Applications have an unfathomable gamut, and so does Real Time Analytics have. After the scalability of the Real Time API, for the inclusion of Custom Dimensions, Just like Core-Reporting System, we will be able to get the UserID custom dimension from the real-time API. This will increase the flexibility of the API to a great extent. Using the UserID, from the real-time API, we will be able to fetch the corresponding information for that particular UserID like Username etc. from our web application database and subsume it with the analytics data from Google in real time. This would enable us to track a particular user of our web application for different dimensions in real time API.

ACKNOWLEDGMENT

The author would like to thank Mr Anshu Jain, Technical Officer of E-Governance Division, CDAC -Noida for giving a prodigious opportunity as Project Intern, during Summer Internship in CDAC -Noida, to fathom out the enigmatic concepts of building a Real Time Reporting Dashboard for serbonline.in (Science and Engineering Research Board portal maintained by E-Governance Division of CDAC -Noida). His guidance was an impetus for the author to explore beyond the theoretical underpinnings and also come up with a great future scope of UserID tracking in real time.

REFERENCES

1. Analytics Core Reporting API: https://developers.google.com/analytics/devguides/reporting/core/v3/
2. Analytics Real Time Reporting API: https://developers.google.com/analytics/devguides/reporting/realtime/v3/
3. Analytics Feature Specific Resources: https://developers.google.com/analytics/solutions/google-analytics-super-proxy
4. Google Identity Platform: https://developers.google.com/identity/protocols/OAuth2
5. Google Cloud Platform: https://cloud.google.com/appengine/docs
6. Google Cloud Platform Quotas: https://cloud.google.com/appengine/docs/quotas?hl=en

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

7. About Real-Time: https://support.google.com/analytics/answer/638635?hl=en
8. http://www.sjsu.edu/people/rakesh.ranjan/courses/cmpe272/s2/Team-Matrix-Google-Analytics.pdf

BIOGRAPHY

Pallavi Tanejais a 4th year B.Tech Student in the Information Technology Department, Indira Gandhi Delhi Technical University for Women. She has been awarded an Appreciation Letter from Mr P.N Barwal, Joint Director, for building the proposed real time analytics system for serbonline.in, in CDAC –Noida. Her research interests are Web & Semantics Technologies, Big Data, Algorithm Analysis and Machine Learning.