

Indian Institute of Technology Bombay



Project Report

EVENT LOGGING AND CONTENT VERSIONING SYSTEM

Principal Investigator

Prof. D.B. Phatak

Project In-Charge

Mr. Nagesh Karmali

Project Mentor

Mr. Abhijeet Dubey

Project Team

Mr. Akshay Sharma
Ms. Sumedha Birajdar
Mr. Suraj Kothawade

Summer Internship 2017 Project Approval Certificate

Department of Computer Science and Engineering

Indian Institute of Technology, Bombay

The project entitled submitted by Mr. Suraj Kothawade, Mr. Akshay Sharma, Ms. Sumedha Birajdar is approved for Summer Internship 2017 programme from 10th May 2017 to 5th July 2017, at Department of Computer Science and Engineering, IIT Bombay.

Prof. Deepak B. Phatak
Dept of CSE, IITB
Principal Investigator

Mr. Nagesh Karmali
Dept of CSE, IITB
Project In-charge

Place: IIT Bombay, Mumbai
Date: July 5 , 2017

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Mr. Akshay Sharma

MNNIT, Allahabad

Ms. Sumedha Birajdar

NIT, Calicut

Date:

Mr. Suraj Kothawade

SGGSIE&T, Nanded

TABLE OF CONTENTS

ACKNOWLEDGEMENT	5
ABSTRACT	6
CHAPTER 1	7-8
1.1 Introduction	7
1.2 Purpose	8
1.3 Scope	8
CHAPTER 2	9-10
2.1 Motivation	9
2.2 Objective	9-10
2.3 Technologies Used	10
CHAPTER 3	11-28
3.1 Features & Implementation	11-18
1. Ability to log to a file instead of a database:	11-12
Implementation	11
Issues	11
Solutions to issues	11-12
Link to git issue	12
Screenshots	12
2. Clear log messages button:	13-14
Implementation	13
Issues	13
Solutions to issues	13
Link to git issue	13
Screenshots	14
3. Backup event_log database content:	14-16
Implementation	14
Issues	14-15
Solutions to issues	15
Link to git issue	16
Screenshots	16
4. Notify Illegal access to authorized user:	16-17
Description	16
Implementation	16-17
Issues	17

	4
Solutions to issues	17
Link to git issue	17
5. Log data to Cloud (Experimental):	17-18
Description	17
Implementation	18
3.2 Visualizations & Implementation	19-28
1. Log Data Visualization Analysis	19-21
Description	19
Implementation	19-20
Screenshots	20-21
2. Log Data Visualization User	21-24
Description	21
Implementation	22
Screenshots	23-24
3. Log Data Visualization Article	24-26
Description	24
Implementation	24-25
Screenshots	25-26
4. Log Data Visualization Collab	26-28
Description	26
Implementation	26-27
Screenshots	27-28
CHAPTER 4	29-30
4.1 Challenges & Solutions	29-30
CHAPTER 5	31-32
5.1 Future Work	31-32
5.2 Results	32
CHAPTER 6	33
6.1 References	33

ACKNOWLEDGEMENT

We, the summer interns of this group, are overwhelmed in all humbleness and gratefulness to acknowledge our deep gratitude to all those who have helped us put our ideas to perfection and have assigned tasks well above the level of simplicity and into something concrete and unique.

We, wholeheartedly thank **Prof. D.B. Phatak** for having faith in us, selecting us to be a part of his valuable project and for constantly motivating us to do better.

We thank **Mr. Nagesh Karmali** for providing us the opportunity to work on this project. We are also very thankful to our mentor **Mr. Abhijeet Dubey** for his valuable suggestions. He was and is always there to show us the right track when needed help. With help of his brilliant guidance and encouragement, we all were able to complete our tasks properly and were up to the mark in all the tasks assigned. During the process, we got a chance to see the stronger side of our technical and nontechnical aspects and also strengthen our concepts.

Last but not the least, we wholeheartedly thank all our other colleagues working in different projects under **Prof. D.B Phatak** for helping us evolve better with their critical advice.

ABSTRACT

Event Logging and Content Versioning System is a Fundamental Research Group project aimed to improve and add functionalities to the existing event log module in Drupal and to provide an interface for the collaborating communities group to fetch the desired data. Moreover, a new module, namely, event log visualization has been added which provides the advantage of analysis of logs with the help of graphical visualizations over the raw data analysis. This project makes use of the existing event log module along with d3.js module. The d3.js module caters various libraries and predefined functions for the development of various SVG data visualization techniques with minimal overhead. The event log module also provides the feature of taking a backup of the event log table. The modules that have been implemented by us is developed mainly using PHP and MySQL database. These module also facilitate the collaborating communities project in implementing the recommendation system.

Both the modules developed under this project can run on ‘Drupal 7’ with php7 installed.

CHAPTER 1

1.1 Introduction

Event Logging and Content Versioning System is a project undertaken by Fundamental Research Group at IIT, Bombay which focuses on advancing the event log module in drupal which was developed earlier by Cipix Internet. This module logs specific events. The events are saved in the database and can be viewed on the page admin/reports/events. Furthermore, a views integration is provided in which you can relate for instance a node to its events. You could use this to display the total number of views, or the number of times that the node has been modified and by which users.

Currently, the following events are supported:

- User authentication (login/logout/request password)
- Node operations (CRUD)
- User operations (CRUD)
- Menu operations (custom menu's and menu items CUD operations)
- Taxonomy operations (vocabulary and term CUD operations)

This project empowers the users to more advanced features based on the core event log module. Apart from the above features, four new features have been added: Namely,

- Clear log messages button
- Ability to log to a file instead of a database.
- Backup event_log database content.
- Notify Illegal access to authorized user.

Since data visualization is imperative and indispensable to business and data analytics, this project comes with three pellucid visualizations. Namely,

- Log Data Visualization(2)
- Log data Visualization User(1)

In order to facilitate the collaborating communities project, interface for the following features/data has also been provided:

- A tabular representation of the top 10 viewed articles
- A tabular representation of the top 10 viewed articles and the corresponding tags for a particular user
- A tabular representation of the top N active users where N is input from the user

1.2 Purpose

This project provides a more efficient and well-equipped version of the existing event log module. It is quite user friendly. It is made with the purpose of enhancing the functionalities and providing a graphical analysis of the performed operations without any manual effort. The data used for analysis is stored in MySQL database. It is quite helpful for those who need operation analysis. Results are shown in a graphical format which is visually appealing too. The purpose of this project is also to support the recommendation system of the collaborating communities project by providing them the required tags, users and articles.

1.3 Scope

The target audience of our project is not limited to any particular group or system rather the module(project) is compatible with every Drupal 7 system having php7 installed on it. This module can be used for analysing views of articles, activities of user, building recommendation systems, etc.

CHAPTER 2

2.1 Motivation

- The Event Log module developed by the Cipix Internet performs only the basic function of logging the operations in the database. There are no additional features to facilitate the user's working experience.
- The user might want to log data to a file instead of database to perform his confidential work but the original event log module does not support any such feature. Ability to log data to a file instead of database is an essential feature for event log module.
- The size of database may increase to a large extent and the user may want to clear it. This is also great during releases to start from a fresh log. The event log module should have the ability to wipe all existing event logs. Possibly behind a permission so that only privileged users are able to clear them. Typically all logs are verified which is why the module should have the ability to just get rid of the ones that are verified to remove clutter.
- Currently, database records are deleted after a particular row limit set by the administrator, but no backup is created for these deleted logs. It would be useful if a backup in a file is created for these logs for future reference. Also, if these backups are mailed to the administrator when required or when cron runs, would be a cherry on the cake.
- There should be a feature that will notify authorized user about the 3 consecutive illegal attempts to access his/her account.
- There can be no useful information that can be inferred from the raw data. So, there should be some graphical visualization of log data.
- The event log module does not provide any information or ranking about the most used articles or active users. For building recommendation system, such information is critical. Moreover, the information about the tags related to the articles is also required.

2.2 Objective

The main goal behind this project is to provide a module that removes the flaws and enhances the functionality of the event logging and content versioning in Drupal 7. The functionalities include:

- Allowing the user to log data to a file instead of database.
- Allowing the user to clear the event log.
- Allowing the user to create a backup of the current event log and send it as an e-mail.

- Adding the security feature that automatically sends an e-mail to the registered user on 3 consecutive login failures.
- Providing graphical visualization of the event log data to prevent user from the complicated raw data analysis.
- Providing a mechanism that lists the most active users.
- Providing a mechanism that lists the most viewed articles.
- Providing a mechanism that visualizes the operations of a particular user.
- Providing a mechanism that lists the most viewed articles by a particular user along with their corresponding tags. This can also serve as an interface for returning required data to the collaborating communities group.

2.3 Technologies Used:

- **PHP (Hypertext Preprocessor):** PHP is a server scripting language, and a powerful tool for making dynamic and interactive Web pages. PHP is a widely-used, free, and efficient. All modules in Drupal are implemented using PHP.
- **Drupal 7:** Drupal is free, open source software that can be used by individuals or groups of users to easily create and manage many types of Web sites. The application includes a content management platform and a development framework. Drupal 7 has been out since early 2011 and is the most widely used version of Drupal.
- **MySQL:** MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web-based software applications. Drupal uses MySQL for database management.
- **D3.js:** D3.js is a JavaScript library for manipulating documents based on data. D3 helps in bringing data to life using HTML, SVG, and CSS. D3's emphasis on web standards gives the full capabilities of modern browsers without tying to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation. Visualizations have been created using d3.js module.
- **Python:** Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy which emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax which allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. Experimental feature of storing log entries on cloud has been implemented using Python.

CHAPTER 3

3.1 Features & Implementation

As mentioned above, a detailed description of each feature is elaborated below:

1. Ability to log to a file instead of a database:

Description

- This feature enables the administrator and privileged users to log the events to file instead of a database.
- This can be done by enabling the File Log feature in the modules section under statistics package.
- Once you enabled this feature, go to the configuration option corresponding to this feature and enable “Save logs to file” checkbox.
- This file can be found under sites/default/files/ in the drupal installation folder and is named as eventlog.txt

Implementation

A new sub-module named event_log_file is added in the event_log folder. The .module file implements the desired action using a couple of hook api functions namely, hook_menu and hook_permission.

Details about this functions can be found at :

- hook_menu : To register paths to configuration page and associate a callback form function with it.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_menu/7.x
- hook_permission : To enable administrator to provide privileges to different users.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_permission/7.x

For logging events to a file, the event_log_insert function is modified in the event_log.module file. PHP file i/o functions are used for this purpose.

Issues

1. Added Sub-Module does not appear in the same section as the event log

2. No configuration option corresponding to the added sub-module
3. Path to eventlog.txt in the event_log_file module is not working.
<https://github.com/fresearchgroup/drupal-logger/issues/7>

Solutions to issues

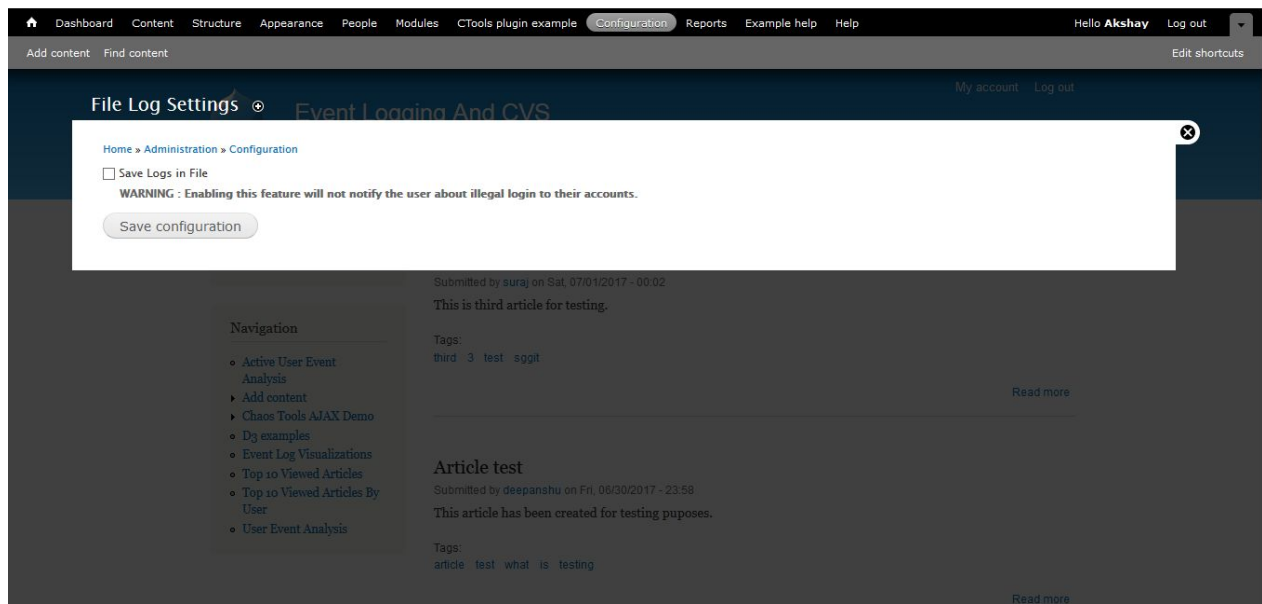
1. Add a package name and dependencies to your .info file so that your sub-module appears under the right section, which is statistics in the event_log module case.
E.g : package = Statistics
dependencies[] = event_log
2. Add configure = admin/config/event_log_file
3. Change the path to public://eventlog.txt instead of specifying the complete path. By doing so, a file is automatically created at sites/default/files/ and no permission issues arise.

Link to git issue

<https://github.com/fresearchgroup/drupal-logger/issues/4>

Screenshot

1. Configuration Page:



2. Clear log messages button:

Description

This feature enables the administrator and privileged users to clear event logs just on the click of button. This button is located in the configuration option in the modules section corresponding to the Delete Log module.

Implementation

A new sub-module named event_log_delete is added in the event_log folder. The .module file implements the desired action using a couple of hook api functions namely, hook_menu and hook_permission.

Details about this functions can be found at :

- hook_menu : To register paths to configuration page and associate a callback form function with it.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_menu/7.x
- hook_permission : To enable administrator to provide privileges to different users.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_permission/7.x

Issues

1. Added Sub-Module does not appear in the same section as the event log
2. No configuration option corresponding to the added sub-module

Solutions to issues

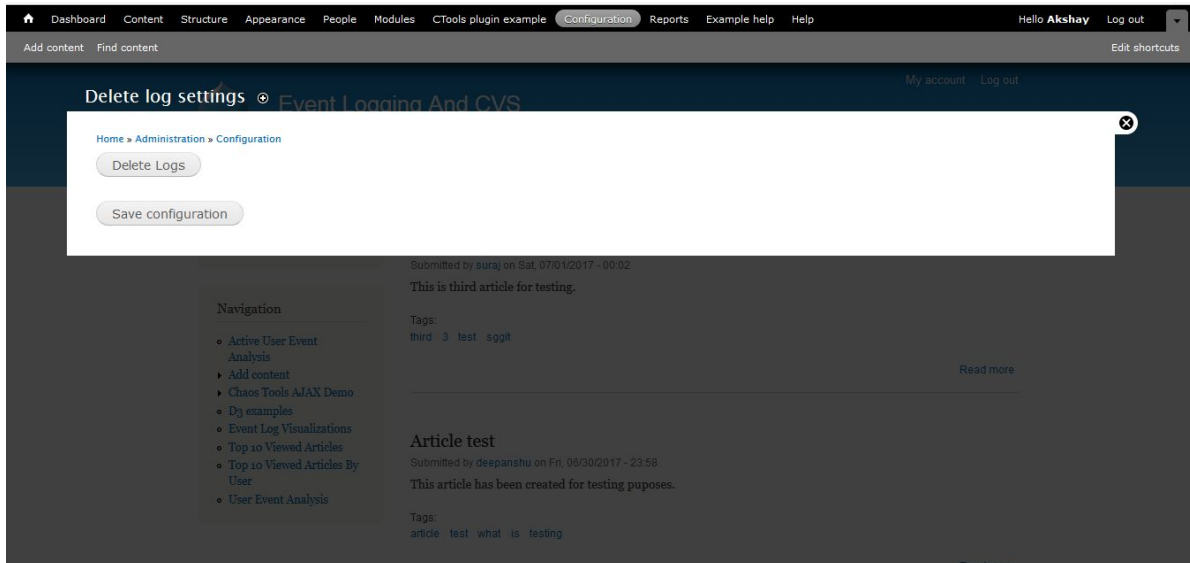
1. Add a package name and dependencies to your .info file so that your sub-module appears under the right section, which is statistics in the event_log module case.
E.g : package = Statistics
dependencies[] = event_log
2. Add configure = admin/config/event_log_delete

Link to git issue

<https://github.com/fresearchgroup/drupal-logger/issues/6>

Screenshot

1. Configuration Page:



3. Backup event_log database content:

Description

- This feature enables the administrator and privileged users to create a backup file that consists of current database records.
- This can be done by enabling the Event Log Backup sub-module in the modules section. Once the module is enabled, backup can be created and viewed by pressing the “View/Update and Save Backup Content” button.
- The backup file is created at /sites/default/files folder. Every time the “View/Update and Save Backup Content” button is clicked, a new file named backup_#.txt is created where # is a suffixed incrementing integer starting from 0.
- Moreover, the content of backup_#.txt file is mailed to the administrator.
- This also helps in keeping various versions of the database content.
- Also, by checking the “Enable Backup Content in Cron Run” checkbox, the same set of actions are performed.

Implementation

A new sub-module named event_log_backup is added in the event_log folder. The .module file implements the desired action using hook api functions namely, hook_menu, hook_cron, hook_mail and hook_permission.

Details about this functions can be found at :

- hook_menu : To register paths to configuration page and associate a callback form function with it.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_menu/7.x
- hook_permission : To enable administrator to provide privileges to different users.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_permission/7.x
- hook_cron : Defines additional instructions to be executed when cron runs.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_cron/7.x
- hook_mail : Prepare a message based on parameters; called from drupal_mail().
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_mail/7.x

Besides the above hook api functions, some user defined functions in the sub-module are:

- event_log_backup_read : Queries the database to fetch the events and saves to backup_#.txt
- event_log_backup_send : Mails the content of backup_#.txt using drupal_mail().

Issues

1. Added Sub-Module does not appear in the same section as the event log.
2. No configuration option corresponding to the added sub-module.
3. Path to backup_#.txt in the event_log_backup module is not working.

Solutions to issues

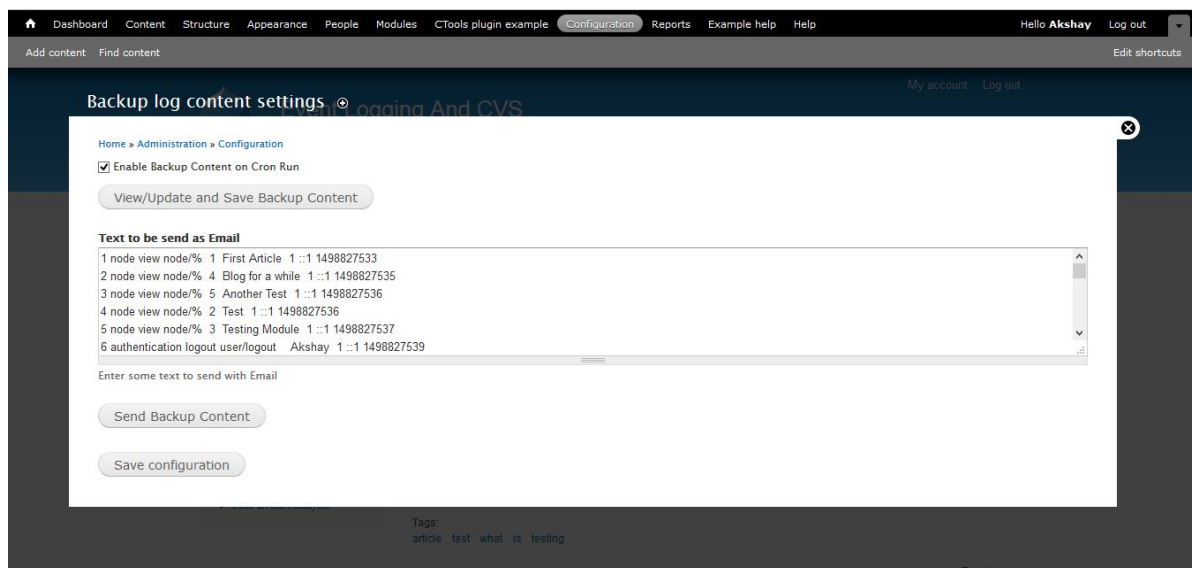
1. Add a package name and dependencies to your .info file so that your sub-module appears under the right section, which is statistics in the event_log module case.
E.g : package = Statistics
dependencies[] = event_log
2. Add configure = admin/config/event_log_backup
3. Change the path to public://backup_#.txt instead of specifying the complete path. By doing so, a file is automatically created at sites/default/files/ and no permission issues arise.

Link to git issue

<https://github.com/fresearchgroup/drupal-logger/issues/5>

Screenshot

1. Configuration Page:



4. Notify Illegal access to authorized user:

Description

This feature will notify authorized user about the 3 consecutive illegal access to his/her account. This will send an email to user's email provided at the time of account creation, to notify when 3 password fails occur. This will also send the IP address from which fails has occurred.

Upstream link : <https://www.drupal.org/node/2883760>

This features notifies the authorized user about fraudulent attempts to access their account by sending a mail.

This feature does not require to be engaged by the administrator. It is enabled by default with the event_log module.

Implementation

One hook api function called hook_mail is used in event_log.module.

- hook_mail : Prepare a message based on parameters; called from drupal_mail().

https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_mail/7.x

In the event_log_insert() function, when the current log operation is “fail”, another user defined function called event_log_login_fail() is called.

- event_log_login_fail() : It checks whether the user is authorized user and if it is, it queries the event_log database to fetch the previous fail login operations to check if the timestamps are in a vicinity of two minutes.

Issues

1. The feature that logs data to file instead of database contradicts the dependency that events are logged to the database. For this feature previous fail entries of logs are accessed. But, accessing these entries from the file will be catastrophic because files are accessed sequentially, leading to significant overheads.

Solutions to issues

1. This issue can be resolved by simply disabling this feature when admin enables log to file feature. This is done by using check condition on value present in checkbox event_log_file_enable in module event_log_file.

Link to git issue

<https://github.com/fresearchgroup/drupal-logger/issues/8>

5. Log data to Cloud (Experimental):

Description

- This feature creates a backup of the event log database by instantaneously writing to a csv file every new record that is being logged to the database. All that is required is a google account for cloud storage (Drive) and a few implementation steps.
- In totality, the following python libraries were required for implementing this feature :
 - import time
 - import json
 - import gspread (Connects to a particular spreadsheet and a worksheet)
 - from oauth2client.client import SignedJwtAssertionCredentials (For authentication via service key)

Implementation

- Prerequisites for implementing this functionality include a cloud storage, in this project we have used Google Drive which provides 15GB of free storage to begin with. Apparently, more storage can be purchased by upgrading Drive
- Since security is a major concern for logging data which might include important credentials, Drive supports authentication using Google APIs.
- Google APIs use the OAuth 2.0 protocol for authentication and authorization. Google supports common OAuth 2.0 scenarios such as those for web server, installed, and client-side applications. The basic steps to be followed for using OAuth 2.0 are :
 - i. Obtain OAuth 2.0 credentials from the Google API Console : Visit the Google API Console to obtain OAuth 2.0 credentials such as a client ID and client secret that are known to both Google and your application. **Note:** The spreadsheet to which the backup is been taken has to be shared with the Client ID specified in the service key obtained from the credentials section in the Google Developers console. This key is available in json & P12 format.
 - ii. Obtain an access token from the Google Authorization Server. Before event log can access private data using a Google API, it must obtain an access token that grants access to that API. A single access token can grant varying degrees of access to multiple APIs. A variable parameter called scope controls the set of resources and operations that an access token permits. During the access-token request, event log sends one or more values in the scope parameter. **Note:** Scope for event log is <https://spreadsheets.google.com/feeds>
 - iii. Send the access token to an API: Access tokens are valid only for the set of operations and resources described in the scope of the token request.
 - iv. Refresh the access token, if necessary: Access tokens have limited lifetimes. If event log needs access to a Google API beyond the lifetime of a single access token, it can obtain a refresh token. A refresh token allows event log to obtain new access tokens.
- The data to be logged is collected from the MySQL drupal database (event_log) by 'MySQLdb' library.
- This data is logged to the Google Spreadsheet by the 'gspread' library.
- This python script can be executed by using the exec function in PHP through sub-module.
- Feel free to find the code for the above implementation on our github page.

Note: This feature was implemented in spite of MongoDB which could scale better for larger data when used with RabbitMQ.

3.2 Visualizations & Implementation

A new module named `log_data_visualization` is added containing two sub-modules:

1. Log Data Visualization Analysis:

Description

- This sub-module visualizes the event log data with two visualization charts:
 - Bar Chart: This chart represents the total count of each operation in the database. X axis represents the triggering count for each operation and Y axis represents the name of each operation.
 - Line Graph: This graph represents the monthly trigger count of each operation. X axis represents each month, along with the year, starting from date corresponding to the timestamp of the first entry in the database. Y axis represents the triggering count for each operation for that corresponding month.
- This module makes use of the `d3.js` module of Drupal. This caters various libraries and predefined functions for the development of various SVG data visualization techniques with minimal overhead.
- There are two dependencies for the `log_data_visualization_analysis` module:
 - Event Log Module: (Github link to this module: <https://github.com/fresearchgroup/drupal-logger>)
 - D3 Module: (Drupal link to this module: <https://www.drupal.org/project/d3>)

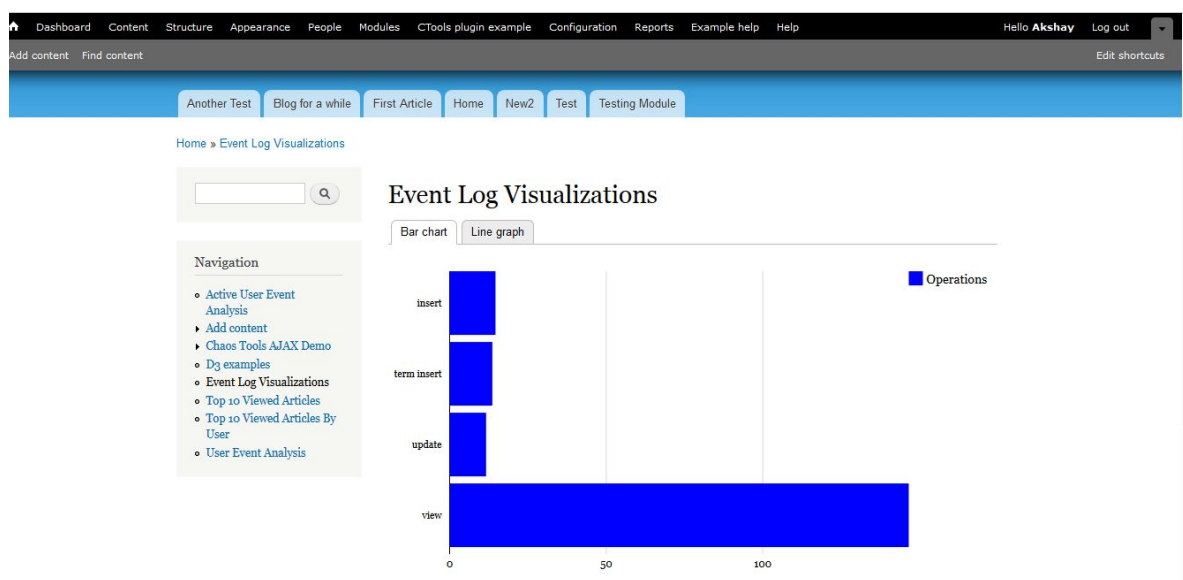
Implementation:

- A new sub-module named `log_data_visualization_analysis` is added. The `.module` file implements the desired action using hook api functions namely, `hook_menu` and `hook_permission`. Details about this functions can be found at :
 - `hook_menu` : To register paths to configuration page and the page used for representation of visualizations. It also associates callback functions to the corresponding registered pages.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_menu/7.x
 - `hook_permission` : To enable administrator to provide privileges to different users.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_permission/7.x

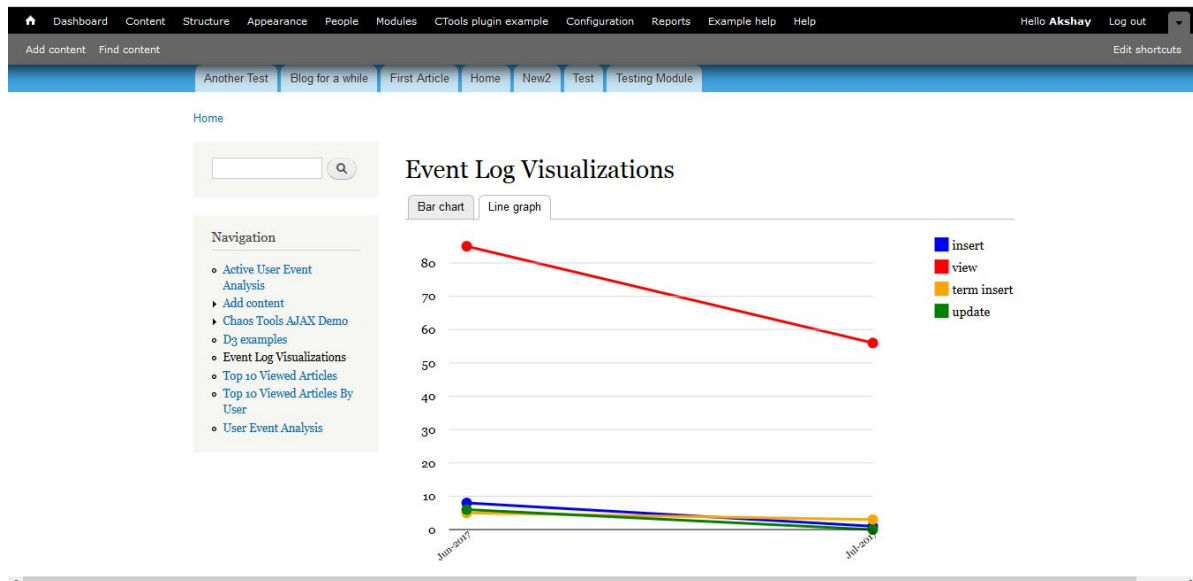
- Besides the above hook api functions, some user defined functions in the sub-module are:
 - `log_data_visualization_analysis_admin_form`: This function creates a ‘Go to Visualizations’ button for redirecting to the visualizations page.
 - `log_data_visualization_analysis_select`: This function is called when the ‘Go to Visualizations’ button is clicked and it redirects the user to the visualizations page.
 - `log_data_visualization_analysis_linegraph`: This function creates a line graph for representing the monthly trigger count of each operation. Firstly, it retrieves the distinct operations from the database, then calculates their count for a particular month and finally converts that data to the desired format for calling the `d3_draw()` function from d3 module.
 - `log_data_visualization_analysis_bar`: This function creates a bar chart for representing the total count of each operation in the database. Firstly, it retrieves the distinct operations from the database, then calculates their total count in the database and finally converts that data to the desired format for calling the `d3_draw()` function from d3 module.

Screenshots

1. Bar Chart:



2. Line Graph:



2. Log Data Visualization User:

Description:

- This sub-module visualizes the event log data for finding a user along with performed operations using a pie chart. The other feature is to find a list of N users with top usage for which N (input) is provided using a textfield.
- This allows the administrator to keep a track of the operations performed by various users.
- This module makes use of the d3.js module of Drupal. This caters various libraries and predefined functions for the development of various SVG data visualization techniques with minimal overhead.
- There are two dependencies for the log_data_visualization_analysis module:
 - Event Log Module: (Github link to this module: <https://github.com/fresearchgroup/drupal-logger>)
 - D3 Module: (Drupal link to this module: <https://www.drupal.org/project/d3>)

Implementation:

- A new sub-module named `log_data_visualization_user` is added. The `.module` file implements the desired action using hook api functions namely, `hook_menu` and `hook_permission`.
- Details about this functions can be found at :
 - `hook_menu` : To register paths to configuration page and the page used for representation of visualizations. It also associates callback functions to the corresponding registered pages.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_menu/7.x
 - `hook_permission` : To enable administrator to provide privileges to different users.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_permission/7.x
- Besides the above hook api functions, some user defined functions in the sub-module are:
 - `log_data_visualization_user_admin_form`: This function uses text field form api to create the required text field and submit button api to create a button.
 - `log_data_visualization_user_select`: This function is called when the ‘Go to Visualization’ button is clicked and it redirects the user to the visualizations page using `drupal_goto()`. Further, it also sends the selected user’s uid to the visualization page.
 - `log_data_visualization_user_pie`: This function is the callback function(this function is executed on redirection) for the visualization page which retrieves the operations and maintains their count for a particular uid (received as argument). Finally, it converts that data to the desired format for calling the `d3_draw()` function from d3 module.
 - `log_data_visualization_user_active`: After entering the number of users required to view, who are extensively using operations and then pressing the submit button, `drupal_goto()` will be called to redirect the user to a page wherein data will be rendered by executing this function.

Screenshots:

1. Configuration Page:

Configuration page for log analysis

Home > Administration > Configuration

Enter number of Users

This text area where admin enter number of top active user

Submit Number

Enter User

This text area where admin enter name of user

Submit User

- Chaos Tools AJAX Demo
- D3 examples
- Event Log Visualizations
- Top 10 Viewed Articles
- Top 10 Viewed Articles By User
- User Event Analysis

Article Title	Article Content
First Article	first article check
New2	new article testing product
Article test	article test

2. Top N Active Users (here N=5):

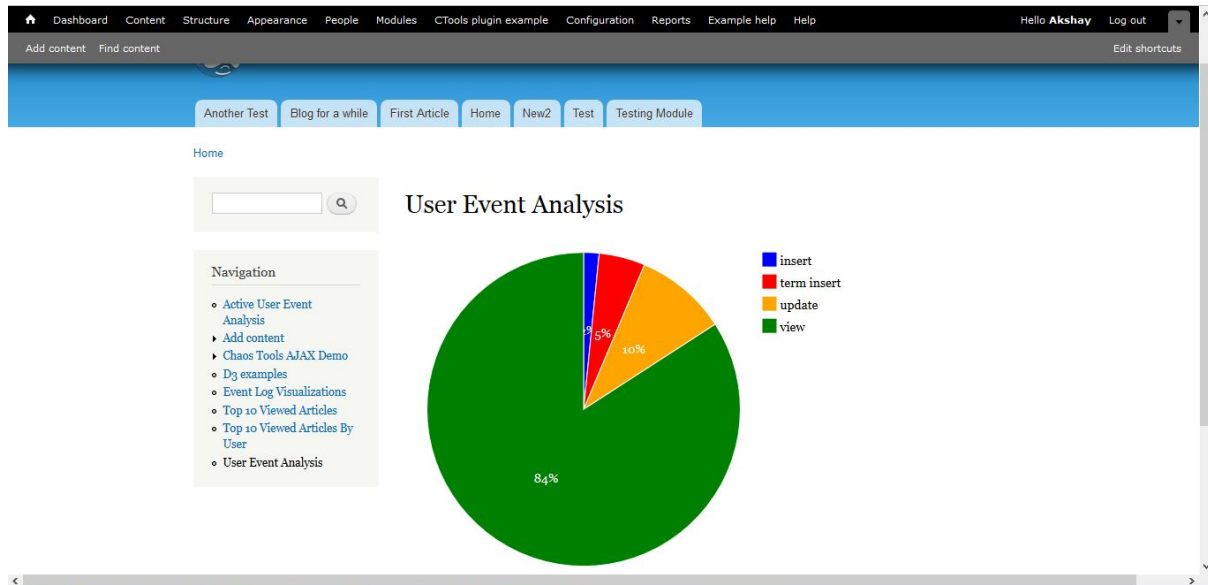
Event Logging And CVS

Home > Administration > Configuration

Active User Event Analysis

USER	% Usage
Akshay	39
Anonymous	16
suraj	11
deepanshu	9
abhishek	7

3. Operation Pie Chart for a particular user (here user=Akshay):



3. Log Data Visualization Article:

Description

- This sub module facilitates the administrator to view a list of top 10 viewed articles.
- This feature can give insights based on which article is rated the most based on which many imperative decisions with regards to business can be taken. For instance, relevant advertisements.
- There is one dependency for the log_data_visualization_article module:
 - Event Log Module: (Github link to this module: <https://github.com/fresearchgroup/drupal-logger>)

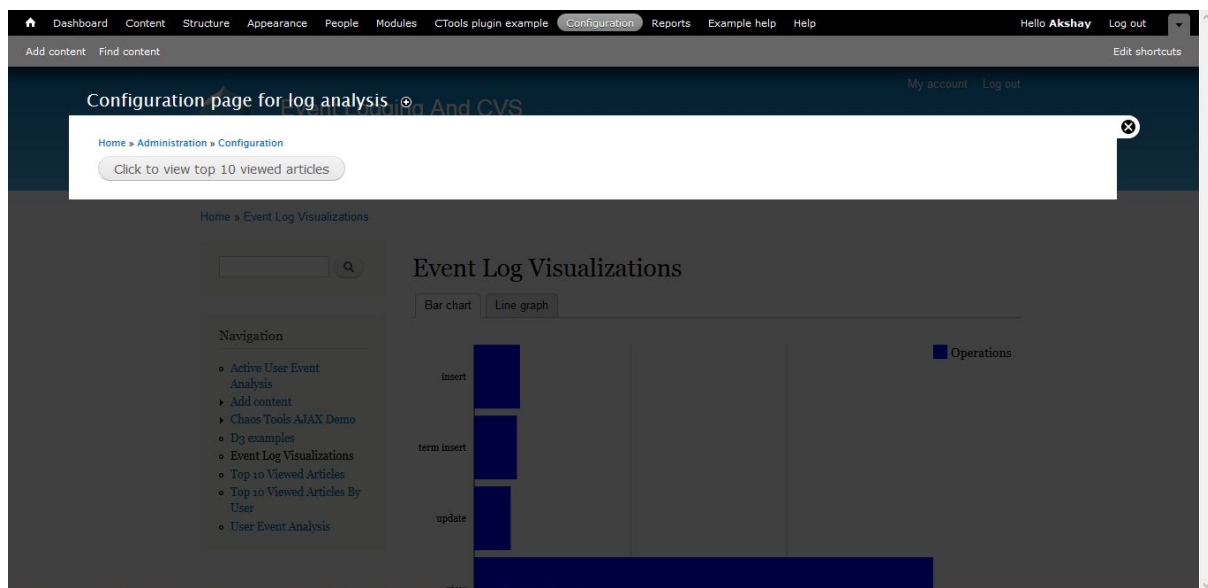
Implementation

- A new sub-module named log_data_visualization_user is added. The .module file implements the desired action using hook api functions namely, hook_menu and hook_permission.
- Details about this functions can be found at :
 - hook_menu : To register paths to configuration page and the page used for representation of visualizations. It also associates callback functions to the corresponding registered pages.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_menu/7.x

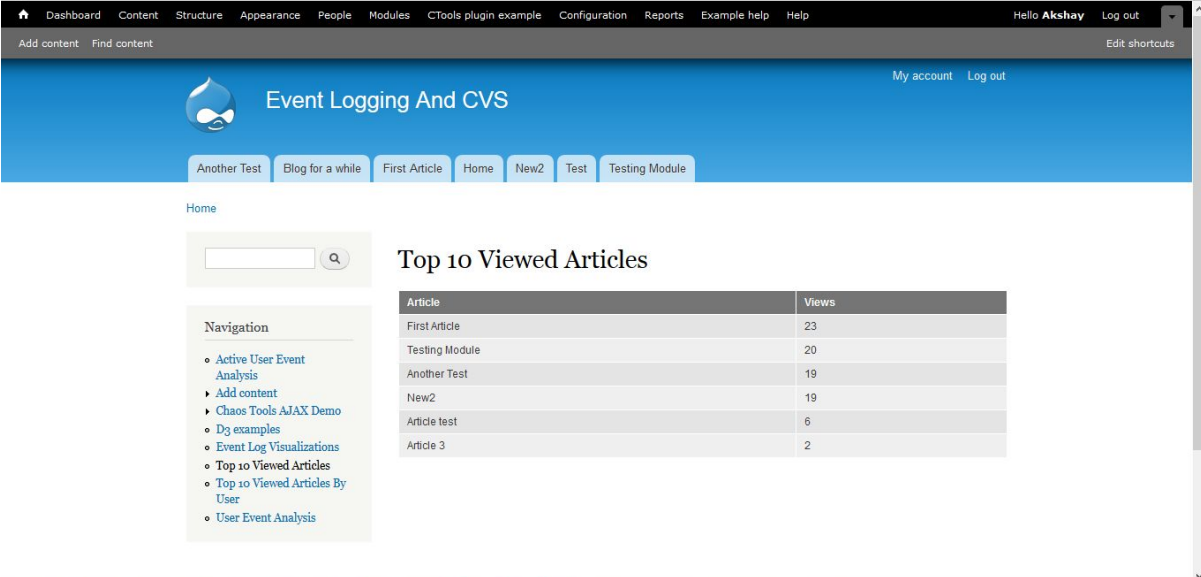
- hook_permission : To enable administrator to provide privileges to different users.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_permission/7.x
- Besides the above hook api functions, some user defined functions in the sub-module are:
 - log_data_visualization_article_admin_form: This function creates a button which on clicked executes log_data_visualization_article_select.
 - log_data_visualization_article_select: This function redirects the user to the desired page using drupal_goto() and the data for this page is rendered by executing log_data_visualization_article_view .
 - log_data_visualization_article_view: By executing an SQL query, appropriate data is fetched from the database and shown to the user in a tabular format.

Screenshots:

1. Configuration Page:



2. Top 10 viewed articles:



The screenshot shows a Drupal website interface. The header includes a navigation menu with links like Dashboard, Content, Structure, Appearance, People, Modules, CTools plugin example, Configuration, Reports, Example help, and Help. The site title is 'Event Logging And CVS'. Below the header, there's a blue banner with a search bar and a 'Navigation' menu. The main content area displays a table titled 'Top 10 Viewed Articles'.

Article	Views
First Article	23
Testing Module	20
Another Test	19
New2	19
Article test	6
Article 3	2

4. Log Data Visualization Collab:

Description

- This sub-module facilitates the administrator to view a list of top 10 viewed articles along with their corresponding tags for a particular user in a tabular format.
- This feature has been implemented to facilitate the development of recommendation system by the collaborating communities project.
- This sub-module provides a function that acts as an interface for the collaborating communities project to access the desired information.
- There is one dependency for the `log_data_visualization_article` module:
 - Event Log Module: (Github link to this module: <https://github.com/fresearchgroup/drupal-logger>)

Implementation

- A new sub-module named `log_data_visualization_collab` is added. The `.module` file implements the desired action using hook api functions namely, `hook_menu` and `hook_permission`.
- Details about this functions can be found at :
 - `hook_menu` : To register paths to configuration page and the page used for representation of visualizations. It also associates callback functions to the corresponding registered pages. https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_menu/7.x

- hook_permission : To enable administrator to provide privileges to different users.
https://api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_permission/7.x
- Besides the above hook api functions, some user defined functions in the sub-module are:
 - log_data_visualization_collab_admin_form: This function creates a text field and a button(using form api) which on clicked executes log_data_visualization_collab_select.
 - log_data_visualization_collab_select: This function redirects the user to the desired page using drupal_goto() and the data(username) for this page is passed as an argument.
 - log_data_visualization_collab_view: By executing an SQL query, appropriate data is fetched from the database and shown to the user in a tabular format.

Screenshots

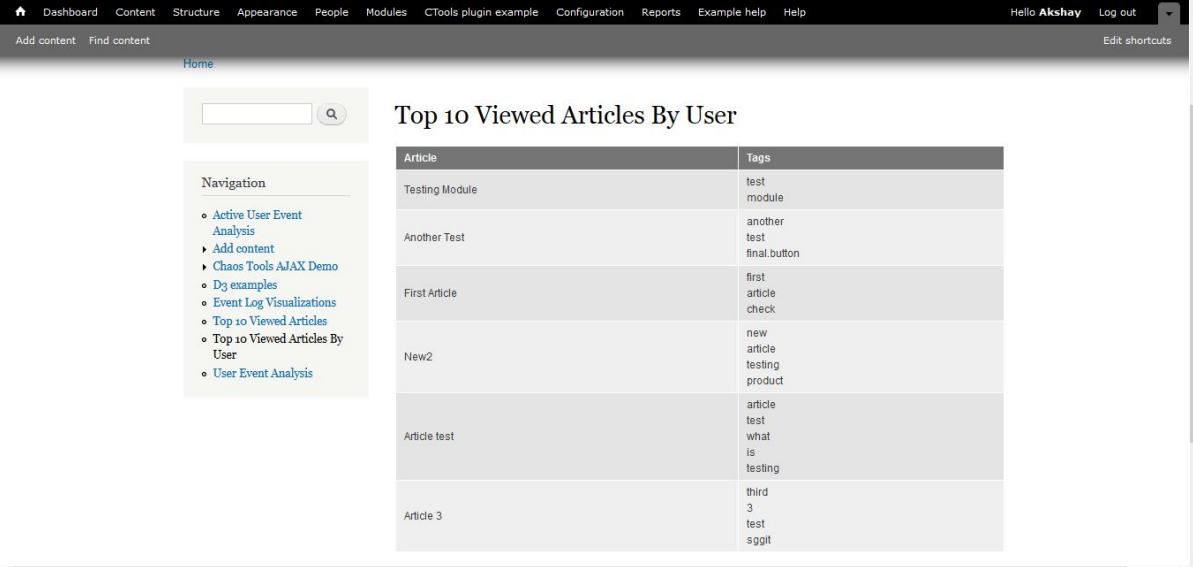
1. Configuration Page:

The screenshot shows the 'Configuration page for log analysis' in a web application. The page has a dark blue header with navigation links: Dashboard, Content, Structure, Appearance, People, Modules, CTools plugin example, Configuration (active), Reports, Example help, and Help. The user is logged in as 'Hello Akshay' and can click 'Log out'. Below the header, there's a search bar with 'Add content' and 'Find content' buttons. The main content area is titled 'Configuration page for log analysis' and includes a breadcrumb trail: Home » Administration » Configuration. A form labeled 'Enter the user name' is present, with a text input field and a button 'Click to view top 10 viewed articles by user'. Below the form, a table displays the top 10 viewed articles by the selected user.

Article	Views
First Article	23
Testing Module	20
Another Test	19
New2	19
Article test	6
Article 3	2

On the left side of the page, there is a 'Navigation' menu with the following items: Active User Event Analysis, Add content, Chaos Tools AJAX Demo, D3 examples, Event Log Visualizations, Top 10 Viewed Articles, Top 10 Viewed Articles By User, and User Event Analysis.

2. Articles and corresponding Tags for a particular user(here user= Akshay):



The screenshot shows a web application dashboard for user Akshay. The top navigation bar includes links for Dashboard, Content, Structure, Appearance, People, Modules, CTools plugin example, Configuration, Reports, Example help, and Help. The user's name 'Akshay' and a 'Log out' button are visible in the top right. Below the navigation bar, there is a search bar and a 'Home' link. On the left side, there is a 'Navigation' menu with links to Active User Event Analysis, Add content, Chaos Tools AJAX Demo, D3 examples, Event Log Visualizations, Top 10 Viewed Articles, Top 10 Viewed Articles By User, and User Event Analysis. The main content area displays the 'Top 10 Viewed Articles By User' for Akshay, which is a table with two columns: Article and Tags.

Article	Tags
Testing Module	test module
Another Test	another test final.button
First Article	first article check
New2	new article testing product
Article test	article test what is testing
Article 3	third 3 test sgit

CHAPTER 4

4.1 Challenges & Solutions

While working on the project, following challenges and their corresponding solutions were identified as:

1.
 - **Challenge:** Added Sub-Module does not appear in the same section as the event log.
 - **Solution:** Add package name and dependencies to your .info file so that the sub-module appears under the right section, which is statistics and visualizations in the event_log and log_data_visualization modules respectively.
E.g : package = Statistics
dependencies[] = event_log
2.
 - **Challenge:** No configuration option is available when a new sub-module is added.
 - **Solution:** Add configure = admin/config/event_log_file in the corresponding .info file of the module
3.
 - **Challenge:** Path to backup_#.txt in the event_log_backup module is not working.
 - **Solution:** Change the path to public://backup_#.txt instead of specifying the complete path. By doing so, a file is automatically created at sites/default/files/ and no permission issues arise.
4.
 - **Challenge:** The feature that logs data to file instead of database contradicts the dependency that events are logged to the database. For “Notify illegal access to authorized user” feature previous fail entries of logs are accessed. But, accessing these entries from the file will be catastrophic because files are accessed sequentially, leading to significant overheads.
 - **Solution:** This issue can be resolved by simply disabling this feature when admin enables log to file feature. This is done by using check condition on value present in checkbox event_log_file_enable in module event_log_file.

5.

- **Challenge:** On enabling the “Ability to log to file instead of database” sub-module the size of file may become too large and hence may become difficult to manage. Also, there should be some centralised storage of files.
- **Solution:** Above problem can be solved by using some type of database partitioning that separates very large databases into smaller, faster, more easily managed parts. This partitioning is known as Sharding. MongoDB provides an auto-sharding functionality which is not present in MySQL. However, following reasons make it unsuitable to opt for MongoDB against MySQL for event_log:
 - In drupal there are certain(around 74 tables) that are hard-wired with core drupal, they are stored in memcache so they can't be completely transferred to mongodb and we require some of those tables for facilitating the “Collaborating communities” project group.
 - The performance benefits which MongoDB provides are outweighed by the complexity of trying to extend Drupal features backed by MongoDB. Moreover most of the drupal modules use hard-coded MySQL queries which are incompatible with MongoDB.
 - A document storage database like MongoDB is much better suited at server with lots of "reads" very quickly and allows for scaling to multiple servers very easily. So, if you have a large website that servers an enormous amount of content to be read (and not updated) by users, it might be advantageous to use a solution like MongoDB.
 - But, if you have a lot of interactive content with editing and updating, so writes to the database, then MongoDB may not offer any improvements and actually may cause problems with duplication if not properly managed. Event Logging is based mainly on updating the database very often, hence MongoDB may not be useful for it.
 - The MongoDB module clearly states what it can store in mongodb: Cache, Field storage, Session, Watchdog, Lock, Block and Queue. However, the required event log table is not included for conversion in the MongoDB module.

The solution to the above challenge is under development and has been implemented on an experimental basis. In this, we are storing the new log entries to file on the cloud by executing a python script and as soon as the threshold for storing the logs in the file is reached, a new file is created and further log entries are done in that file.

CHAPTER 5

5.1 Future Work

1. Sharding in Backup event_log database content using python:

- Presently, the event_log_backup sub-module enables the administrator and privileged users to create a backup file that consists of current database records. However, this feature can be extended to incorporate the auto sharding functionality to enable partitioning and preventing the creation of large-sized backup files, hence making their management easier.
- As of now, records are being stored in a csv file on the cloud using the implementation of Feature 5 as mentioned above. However, if the administrator requires a restriction of x records on a page, then that can be easily done by changing the worksheet on required number of records.

2. Using the Pagerank algorithm to visualize the pages with most hits:

- The Pagerank algorithm designed by Larry Page can be implemented to display the pages with maximum hits with the help of a Force Directed graph.
- The force directed graph takes into consideration the page ranks, based on which the diameter of each node of the graph is determined.
- Along with the diameter of the node, the links to different nodes in the graph tantamounts to the pages that are accessible from the respective page under consideration.
- Firstly, a start url is used as the first node (generally the homepage). The handle for this url is fetched using the urllib library and the source code for this page is then extracted and stored in the database. Then, subsequent url's are extracted from the source code via href tags and are stacked in the Pages table. (Note: The database "urldb" has three tables, namely, "Pages", "Webs", "Links") This functionality is coded in the spider.py file.
- Secondly, extracted data from url's are then dumped into the database using spdump.py which is later used to create a json file required for visualisation. This functionality is implemented in the spdump.py file.
- Thirdly, retrieved pages are ranked using the pagerank algorithm and new rank is calculated and replaced. (The old rank 1.0 is replaced) This functionality is implemented in the sprank.py file.
- Ranked pages are then visualized by creating a json file of pages and their corresponding ranks. Visualization is done by using d3.v2.min.js and code of force directed graph which is readily available on d3js.org. This functionality is implemented in the spjson.py file. Visualization can be accessed either by opening the force.html file.

3. Extension of experimental feature of logging data on cloud:

- Currently, this feature allows the user to log the data into the Spreadsheet present on cloud.
- However, for accessing this feature the python script has to be executed explicitly. This script can't be executed through Drupal sub-module due to import issues.
- In the future work, a solution has to be found to execute the python script through Drupal.

5.2 Results:

In the Event Logging & Content Versioning System project, we have successfully extended the functioning of Event Log module and implemented a new Log Data Visualization module.

The Event Log Module is now able to provide the following features apart from the existing features created by [Cipix](#) :

- Clear log messages button
- Ability to log to a file instead of a database.
- Backup event_log database content.
- Notify Illegal access to authorized user.
- Log data to Cloud (Experimental)

The Log Data Visualization Module also successfully provides users a graphical analysis of event log data as well as an interface which caters to the requirements of the collaborating communities group. Major visualizations of the Log Data Visualization module include:

- User
- Analysis
- Article
- Collab

Here's a link to the git repository :

<https://github.com/fresearchgroup/drupal-logger/wiki>

CHAPTER 6

6.1 References

- **Drupal:** https://www.drupal.org/project/event_log
- **Drupal D3:** <https://www.drupal.org/project/d3>
- **W3Schools:** <https://www.w3schools.com/php/>
- **Tutorialspoint:** <https://www.tutorialspoint.com/mysql/>
- **D3.js:** <https://d3js.org/>
- **Code Karate Videos:**
<https://www.youtube.com/playlist?list=PL-Ve2ZZ1kZNRJVY5cpaLaJoJdB8AiLA96>
- **Drupal 7 Core Tables:** <https://www.drupal.org/node/2360815>
- **Stackexchange:**
<https://drupal.stackexchange.com/questions/15493/drupal-7-with-mongodb>
- **MongoDB & Drupal(@DrupalCon):**
<https://derickrethans.nl/drupal-and-mongodb.html>
- **Monarch Digital:**
<https://www.monarchdigital.com/blog/2017-02-03/using-mongodb-drupal>
- **Wikipedia:** [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))