

FOSSEE Fellowship Report

SPOKEN TUTORIAL ANALYTICS SYSTEM

Prepared By :

Arish Rehman Khan

Student of MCA (I year) at JNU

Email: arish.rehman.khan@gmail.com

Table of Contents

Table of Contents	1
Introduction	3
Preparing the Github Repository	4
Using Two Databases	5
Generating fake data	7
Some Important Terms	8
Stats Calculation Scripts	8
Daily Statistics	8
Weekly Statistics	9
Monthly Statistics	9
Yearly Statistics	9
Average Statistics	10
Event Statistics	10
Foss Statistics	11
Visitor Activity Statistics	12
Visitor Path Statistics	13
Page View Activity Statistics	14
Came From Activity Statistics	14
Exit Link Activity Info	14
Visitor Spot Statistics	15
Visitor Info Statistics:	15
Location Statistics	17
System Statistics	17
Sources Statistics	18
Came From Statistics	19
Exit Link Statistics	19
Visualizing the statistics	20
Dashboard Page	20
Tiles	20
Trend Chart	20
Datatable	20
Events Page	22

Page Analysis	23
Trend Chart	23
Datatable	23
Foss Page	24
Activities	24
Visitor Activity Page	25
Visitor Path Page	26
Page View Activity	26
Came From Activity	27
Exit Link Activity	28
Visitor Map	29
Visitor Info	30
Reports Page	31
Location Report	31
Foss/Page Report	31
System Report	32
Traffic Report	32
Future Work	34
Conclusion	34

Introduction

This report is about the work that I did in the FOSSEE Summer Fellowship programme - 2020. The task was to develop an Analytics System for the [Spoken Tutorial](#) website and course completion and tutorial progress statistics for users. At this point of time, the spoken tutorial website uses a third party web analytics system called [stat counter](#), and we had to develop something like that. I along with my teammate [Krithik Vaidya](#) developed the system. The work was mainly divided into two major tasks -

1. Storing the log containing information such as ip address, timestamp, page url, browser info, etc. into the mongo database.
2. Using these logs to calculate meaningful insights such as page views, unique visits, returning visits, etc. and their visualization.

First part along with course completion and tutorial progress tasks were done by my teammate. I worked mostly on the second part.

This report contains only the work that I did. Report about my teammate's work can be found [here](#).

The source code of the project is hosted on [github](#).

Preparing the Github Repository

I initialised [github repo](#) with django 2.2.12 and added startbootstrap [sb-admin-2](#) (screenshot attached below) to use it as a base template for creating the dashboard.

The image shows a comprehensive dashboard for 'SB ADMIN 2'. On the left is a blue sidebar with navigation options: Dashboard, Components, Utilities, Pages, Charts, and Tables. The main content area is titled 'Dashboard' and includes a search bar, a user profile for 'Valerie Luna', and a 'Generate Report' button. Key metrics are displayed in four cards: Earnings (Monthly) at \$40,000, Earnings (Annual) at \$215,000, Tasks at 50% completion, and Pending Requests at 18. Below these are two charts: 'Earnings Overview' (a line graph showing monthly trends from Jan to Nov) and 'Revenue Sources' (a donut chart showing Direct, Social, and Referral sources). A 'Projects' section features a table with progress bars for Server Migration (20%), Sales Tracking (40%), Customer Database (60%), Payout Details (80%), and Account Setup (Completed). An 'Illustrations' section promotes 'unDraw' with an illustration of a person at a computer and a link to browse illustrations. A 'Development Approach' section explains the use of Bootstrap 4 utility classes. At the bottom, there are six color-coded utility class boxes: Primary (#4e73df), Success (#1cc88a), Info (#36b9cc), Warning (#ffc23e), Danger (#e74c3b), and Secondary (#858796). The footer contains 'Copyright © Your Website 2019' and an upward arrow icon.

Using Two Databases

The registration data of admin and users is saved in 'spoken' database which is a mysql database and we have to store the logs and statistics in a separate mongo database named 'logs' for efficient querying. To solve this problem we used [django routers](#). Django routers help

to use multiple databases in a single application. `/dashboard/router.py` is used to route the auth related queries to 'spoken' database and the other queries are routed to 'logs' database.

```
dashboard > router.py > AuthRouter
1 class AuthRouter:
2     """
3     A router to control all database operations on models in the
4     auth and contenttypes applications.
5     """
6     route_app_labels = {'auth', 'contenttypes'}
7
8     def db_for_read(self, model, **hints):
9         """
10        Attempts to read auth and contenttypes models go to auth_db.
11        """
12        if model._meta.app_label in self.route_app_labels:
13            return 'spoken'
14        return None
15
16    def db_for_write(self, model, **hints):
17        """
18        Attempts to write auth and contenttypes models go to auth_db.
19        """
20        if model._meta.app_label in self.route_app_labels:
21            return 'spoken'
22        return None
23
24    def allow_relation(self, obj1, obj2, **hints):
25        """
26        Allow relations if a model in the auth or contenttypes apps is
27        involved.
28        """
29        if (
30            obj1._meta.app_label in self.route_app_labels or
31            obj2._meta.app_label in self.route_app_labels
32        ):
33            return True
34        return None
```

Part of `/dashboard/router.py` file

The databases are specified as DATABASES and `/dashboard/router.py` file as DATABASE_ROUTERS in `/analytics_system/settings.py` file.

```

86 DATABASES = {
87     # Default database
88     'default': {
89         'ENGINE': 'django',
90         'NAME': MONGO_DB,
91     },
92     # Database used for authentication
93     'spoken': {
94         'ENGINE': 'django.db.backends.mysql',
95         'NAME': DB,
96         'USER': DB_USER,
97         'PASSWORD': DB_PASS,
98         'HOST': '',
99         'PORT': '',
100     },
101 }
102
103 DATABASE_ROUTERS = [
104     # Router to use 'spoken' database for authentications
105     'dashboard.router.AuthRouter',
106 ]

```

Part of `/analytics_system/settings.py` file

To migrate the each database separately the following command is used.

```
$ python3 manage.py migrate --database=database_name
```

Database used for auth purpose : spoken (mysql)

Database used for storing logs and statistics : logs (mongo)

Generating fake data

The logs of the spoken website were not available as statcounter does not provide logs it only shows the calculated statistics. I wrote a python script `gen_fake_data.py` so that I can generate

fake logs to test the algorithms that calculate the statistics such as page views, unique visits, returning visits, etc. The scripts uses [django-populate](#) package to generate fake logs.

```
218 # Creating populator object
219 populator = Faker.getPopulator()
220
221 # function for Logs Model
222 def randomDataLog():
223     data = {
224         'datetime': lambda x: populator.generator.date_time_between(start_date='-2y', end_date='+10d', tzinfo=india_tz),
225         'path_info': lambda x: "/watch/" + random.choice(foss) + "/" + random.choice(tutorials) + "/" + random.choice(languages)
226         'event_name': lambda x: random.choice(event_names),
227         'page_title': lambda x: random.choice(page_titles),
228         'visited_by': lambda x: populator.generator.user_name() if random.randint(0, 1) == 1 else "anonymous",
229         'ip_address': lambda x: "230.124.0." + str(random.randint(0, 255)),
230         'referrer': lambda x: random.choice(referrer),
231         'browser_family': lambda x: random.choice(browsers),
232         'browser_version': lambda x: random.choice(browser_versions),
233         'os_family': lambda x: random.choice(os),
234         'os_version': lambda x: random.choice(os_versions),
235         'device_family': lambda x: random.choice(["Lenovo K8 Note", "Realme XT", "Realme X2", "Samasung M31S", "Samsung M40"]),
236         'device_type': lambda x: random.choice(["Mobile", "PC", "Tablet"]),
237         'latitude': lambda x: populator.generator.local_latlng(country_code='IN', coords_only=True)[0],
238         'longitude': lambda x: populator.generator.local_latlng(country_code='IN', coords_only=True)[1],
239         'city': lambda x: random.choice(cities),
240         'region': lambda x: random.choice(states_uts),
241         'country': 'India',
242     }
243     return data
244 # Adding data to populator object
245 populator.addEntity(Log, num_rows_log, randomDataLog())
246
247 # Inserting data to database
248 populator.execute()
```

Part of *gen_fake_data.py* file

Steps to generate fake logs:

1. Open the *gen_fake_data.py* file and change the value of `num_rows_log` and `num_rows_exitlink` to specify the number of logs to be generated for each **Log** and **Exit Link Activity** Model. (why we have to generate logs for Exit Link is discussed later).
2. Change the `start_date` and `end_date` of line 224 given in screenshot to specify the range of datetime for which logs are to be generated.
3. Now open the python shell using command
`$ python3 manage.py shell`
4. In the shell run
`>>> exec(open("gen_fake_data.py").read())`

Note:

1. It will take time depending upon the number of logs to be generated.
2. You can specify your own data in the scripts.

Some Important Terms

1. **Page View** : A page view is counted if a user visits any page of a website.
2. **First time visit** : When a user visits any page for the first time.

3. **Returning visit** : When a user revisits the website after a fixed amount of time (30 min in our case).
4. **Unique visit** : When a user visits any page for the first time or revisit any page after a fixed amount of time (30 min in our case).
5. **Visit length** : It is the time duration between the first and last page view in a unique visit.
6. **Path** : Path that the visitor followed when he/she browse the website.

Stats Calculation Scripts

The calculation scripts present in the `/dashboard/calculation_scripts` directory are executed daily at 12:05 AM using [celery](#) package. These scripts calculate and store the statistics necessary for visualization.

Disadvantage: Since these scripts only once a day. We can only calculate and display the data of previous day. The statcounter on the other hand displays the real time data. This functionality can be added in the next version of the system.

These scripts are explained in detail :

Daily Statistics

These refer to the statistics which tell the number of page views, first time visits, returning visits, unique visits and unique visitors of a particular day. These statistics are calculated using `/dashboard/calculation_scripts/dailyStats.py` script.

Algorithm used to calculate daily statistics:

- 1) Get all the logs which were stored on the previous day in increasing order of datetime.
- 2) Total page views will be equal to the number of logs.
- 3) Find all the distinct IP addresses present in the logs.
- 4) Total unique visitors will be equal to the number of IP addresses.
- 5) Initialize the variables `unique_visits`, `first_time_visits` and `returning_visits` with zero.
- 6) For each ip address
 - a) If the IP address occurs for the first time, increment `first_time_visits` and `unique_visits`.
 - b) Else if the same IP occurs after 30 min, increment `returning_visits` and `unique_visits`.
- 7) Save the statistics along with the date.

Weekly Statistics

These refer to the statistics which tell the number of page views, first time visits, returning visits, unique visits and unique visitors of a particular week. These statistics are calculated using

/dashboard/calculation_scripts/weeklyStats.py script.

Algorithm used to calculate weekly statistics:

1. Get all the daily statistics of the current week.
2. Initialize the variables `page_views`, `unique_visits`, `first_time_visits`, `returning_visits` and `unique_visitors` with zero.
3. Loop through all the statistics of current week and add daily statistics to these variables.
4. If the statistics of current week are present, delete them.
5. Save the statistics with the week number of the year and the current year.

Monthly Statistics

These refer to the statistics which tell the number of page views, first time visits, returning visits, unique visits and unique visitors of a particular month. These statistics are calculated using */dashboard/calculation_scripts/monthlyStats.py* script.

Algorithm used to calculate monthly statistics:

1. Get all the daily statistics of the current month.
2. Initialize the variables `page_views`, `unique_visits`, `first_time_visits`, `returning_visits` and `unique_visitors` with zero.
3. Loop through all the statistics of current month and add daily statistics to these variables.
4. If the statistics of current month are present, delete them.
5. Save the statistics with the month number of the year and the current year.

Yearly Statistics

These refer to the statistics which tell the number of page views, first time visits, returning visits, unique visits and unique visitors of a particular year. These statistics are calculated using */dashboard/calculation_scripts/yearlyStats.py* script.

Algorithm used to calculate yearly statistics:

1. Get all the Monthly statistics of the current year.
2. Initialize the variables `page_views`, `unique_visits`, `first_time_visits`, `returning_visits` and `unique_visitors` with zero.
3. Loop through all the statistics of current year and add daily statistics to these variables.
4. If the statistics of current year are present, delete them.
5. Save the statistics with the month number of the year and the current year.

Average Statistics

These refer to the statistics which tell the average daily, weekly, monthly and yearly page views, first time visits, returning visits, unique visits and unique visitors. These statistics are calculated using */dashboard/calculation_scripts/averageStats.py* script.

Algorithm used to calculate average statistics:

1. Get all the Daily/Weekly/Monthly/Yearly statistics.
2. Loop through all the daily statistics and calculate total page views, unique visits, first time visits and returning visits.
3. Divide these values with the total number of days, weeks, months and years for each type.
4. Save the statistics with current datetime (used when the latest record is to be displayed).

Event Statistics

These refer to the info about page views and unique visits that a page gets at a particular day.

The info includes

1. Date
2. Time
3. Page URL (path)
4. Page title
5. Page views
6. Unique Visits

These statistics are calculated using `/dashboard/calculation_scripts/eventStats.py` script.

Note : When logs are stored each page view is associated with an event name, if we use JS implementation of log storing then we can save the title of the page, but it is not possible with middleware implementation. So if we don't have a page title the page title can be found using the list containing the event name and page title present in `/dashboard/events_info.py` file.

Algorithm used to calculate event statistics:

1. Get all the logs which were stored on the previous day in increasing order of datetime.
2. Find all the distinct path_info (paths) present in the logs.
3. For each path in paths
 - a. Get all the logs 'daily_logs' of the previous day having path_info as 'path'.
 - b. Find all the distinct IP addresses (ip_address) present in the 'daily_logs'.
 - c. Set the variables

```
unique_visits = 0
first_time = 0
```
 - d. For ip in ip_address
 - i. For log in daily logs
 1. if ip == log.ip_address:
 - a. if first_time == 0:

```
first_time = 1
unique_visits += 1
```
 - else:

```
if ip occurs after 30 min,unique_visits += 1
```
 - b. prev_datetime = log.datetime

- e. Save the required fields, if the page title is not available, fetch it from *events_info.py* using *event_name*.

Foss Statistics

These refer to the info about page views and unique visits that a foss gets at a particular day.

The info includes

1. Date
2. Time
3. Foss name
4. Page views
5. Unique Visits

These statistics are calculated using */dashboard/calculation_scripts/fossStats.py* script.

Algorithm used to calculate foss statistics:

4. Get all the logs which were stored on the previous day in increasing order of datetime.
5. Find all the distinct foss names (foss) present in the logs.
6. For each *foss_name* in *foss*
 - a. Get all the logs 'daily_logs' of the previous day having *foss_name* in the *path_info*.
 - b. Find all the distinct IP addresses (*ip_address*) present in the 'daily_logs'.
 - c. Set the variables
`unique_visits = 0`
`first_time = 0`
 - d. For *ip* in *ip_address*
 - i. For log in daily logs
 1. if `ip == log.ip_address`:
 - a. if `first_time == 0`:
`first_time = 1`
`unique_visits += 1`
 - else:
if ip occurs after 30 min, set `unique_visits += 1`
 - b. `prev_datetime = log.datetime`
 - e. Save the required fields.

Visitor Activity Statistics

These refer to the info about a particular visit of a visitor like

1. Total **page views** viewed by the visitor
2. Today's **total visits** by the visitor.
3. Latest page view datetime
4. Location

5. Visit length
6. IP address
7. Info about system (browser, os and device)
8. Referrer URL
9. Entry Page
10. Exit Page
11. Visit Page

These info are calculated using */dashboard/calculation_scripts/visitorActivityStats.py* script.

Algorithm used to calculate visitor activity statistics:

7. Get all the logs which were stored on the previous day in increasing order of datetime.
8. Find all the distinct IP addresses present in the logs.
9. For each IP address 'ip'
 - a. Get all the logs 'ip_logs' of the previous day having IP address as 'ip'.
 - b. Set the variables as -


```
prev_datetime = ip_logs.first().datetime
first_datetime = ip_logs.first().datetime
last_datetime = ip_logs.first().datetime
referrer = '(No referring link)'
total_visits = 1
page_views = 1
entry_page = ip_logs.first().path_info
flag = 0
```
 - c. For each 'log' of 'ip_logs'
 - i. If 'ip' occurs after 30 min, save the required fields and set the variables as


```
first_datetime = log.datetime
last_datetime = log.datetime
total_visits += 1
page_views = 1
flag = 1
```
 - ii. Else set


```
page_views += 1
referrer = log.referrer
last_datetime = log.datetime
if flag == 1:
    flag = 0
    entry_page = log.path_info
```
 - iii. Set prev_datetime = log.datetime
 - d. If flag == 1, set flag = 0 and continue else save the required fields.

Visitor Path Statistics

These also refer to the info about a particular visit of visitor but it focus more on the path which the user take when he/she visits the website, the info calculated is

1. Location
2. IP address
3. Visit number
4. Info about system (browser, os and device)
5. Path
 - a. Datetime
 - b. Referrer URL
 - c. Page visited

These statistics are calculated using */dashboard/calculation_scripts/visitorPathStats.py* script.

Algorithm used to calculate visitor path statistics:

1. Get all the logs which were stored on the previous day in increasing order of datetime.
2. Find all the distinct IP addresses present in the logs.
3. For each IP address 'ip'
 - a. Get all the logs 'ip_logs' of the previous day having IP address as 'ip'.
 - b. Set the variables as -

```
prev_datetime = ip_logs.first().datetime
path = []
visit_num = 0
```
 - c. For each 'log' of 'ip_logs'
 - i. If 'ip' occurs after 30 min,
Set path += [{'datetime': log.datetime, 'referrer': log.referrer, 'page_url': log.path_info}] and visit_num += 1
Save the required fields.
Set path = []
 - ii. Else set path += [{'datetime': log.datetime, 'referrer': log.referrer, 'page_url': log.path_info}]
 - iii. Set prev_datetime = log.datetime

Page View Activity Statistics

These refer to the info about a page view, viewed by a visitor and contains

1. Date
2. Time
3. Info about system (browser, os and device)
4. Location
5. IP address

6. Referrer
7. URL of page viewed

These statistics are calculated using */dashboard/calculation_scripts/pageViewActivityStats.py* script.

Algorithm used to calculate page view activity statistics:

1. Get all the logs which were stored on the previous day in increasing order of datetime.
2. Find all the distinct IP addresses present in the logs.
3. For each IP address 'ip'
 - a. Get all the logs 'ip_logs' of the previous day having IP address as 'ip'.
 - b. Save the required fields.

Came From Activity Statistics

These refer to the info about sources from where the visitor comes to the website, it contains info like

1. Date
2. Time
3. Referrer URL
4. URL of page viewed

These statistics are calculated using */dashboard/calculation_scripts/cameFromActivityStats.py* script.

Algorithm used to calculate came from activity statistics:

1. Get all the logs which were stored on the previous day in increasing order of datetime.
2. For each log in logs if referrer is present and is not of the spoken website, save the required fields.

Exit Link Activity Info

These refer to the info about the links to which visitors are going from spoken website, it contains info like

1. Date
2. Time
3. Exit Link Clicked
4. Exit Page

This is stored in the database using JS implementation of logs storing and no info is needed to be calculated for Exit Link Activity.

Visitor Spot Statistics

These refer to the info about location (co-ordinates) from where the visitor belong and contains info like

1. Date
2. Time
3. IP Address
4. Location Co-ordinates

These statistics are calculated using `/dashboard/calculation_scripts/visitorSpotStats.py` script and is used to display the location of visitors on graph.

Algorithm used to calculate visitor spot statistics:

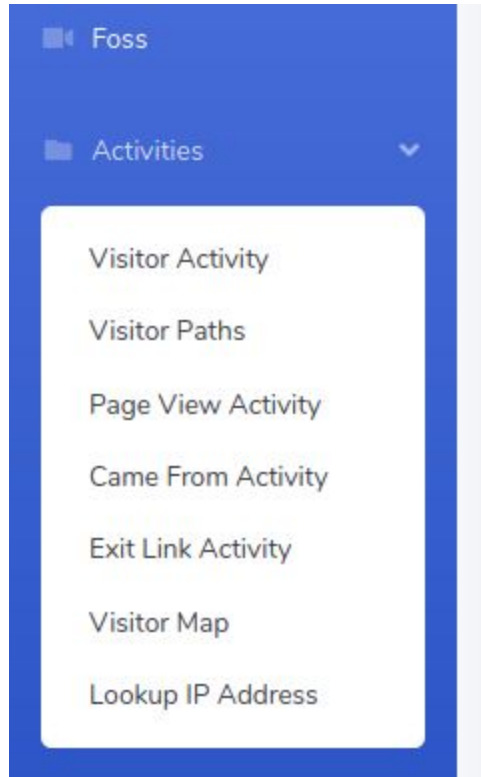
1. Get all the logs which were stored on the previous day in increasing order of datetime.
2. Find all the distinct IP addresses present in the logs.
3. For each IP address 'ip'
 - a. Get the first log of the previous day having IP address as 'ip'.
 - b. Save the required fields.

Visitor Info Statistics:

Visitor info Statistics refer to all the info about the visitor, it contains

1. Referrer URL
2. Location
3. IP address
4. Info about system (browser, os and device)
5. Returning visits
6. Visits length
7. Path
 - a. Datetime
 - b. Referrer URL
 - c. Page visited

These statistics are calculated using `/dashboard/calculation_scripts/visitorInfoStats.py` script and are used to display info about the visitors whenever the admin searches for it using the IP address at Lookup IP address option.



Algorithm used to calculate visitor info statistics:

1. Get all the logs which were stored on the previous day in increasing order of datetime.
2. Find all the distinct IP addresses present in the logs.
3. For each IP address 'ip'

- a. Get all the logs 'ip_logs' of the previous day having IP address as 'ip'.
- b. Set the variables as -

```
prev_datetime = ip_logs.first().datetime
first_datetime = ip_logs.first().datetime
last_datetime = ip_logs.first().datetime
path = []
referrer = '(No referring link)'
returning_visits = 0
```

- c. For each 'log' of 'ip_logs'

- i. If 'ip' occurs after 30 min,
 1. Save the required fields.
 2. Set the variables

```
first_datetime = log.datetime
last_datetime = log.datetime
returning_visits += 1
flag = 1
path = [{'datetime': log.datetime, 'referrer': log.referrer,
'page_url': log.path_info}]
```

- ii. Else set the variables
 - path += [{'datetime': log.datetime, 'referrer': log.referrer, 'page_url': log.path_info}]
 - referrer = log.referrer
 - last_datetime = log.datetime
- iii. Set prev_datetime = log.datetime
- d. If flag == 1, set flag = 0 and continue else save the required fields.

Location Statistics

These statistics refer to the info about the location (city and region) from which users are visiting the website. The info contains:

1. City stats
 - a. Date
 - b. Time
 - c. City name
 - d. Page views
2. Region stats
 - a. Date
 - b. Time
 - c. Region name
 - d. Page views

These statistics are calculated using */dashboard/calculation_scripts/locationStats.py* script.

Algorithm used to calculate location statistics:

1. Get all the logs which were stored on the previous day.
2. Calculate the distinct cities and the page views from them.
3. Save the city statistics.
4. Calculate the distinct region and the page views from them.
5. Save the region statistics.

System Statistics

These statistics refer to the info about the system (browser, os and device) from which users are visiting the website. The info contains:

1. Browser Statistics
 - a. Datetime
 - b. Browser type (pc, mobile, etc)
 - c. Browser name (Chrome, Firefox, etc)
 - d. Page views
2. OS Statistics

- a. Datetime
 - b. OS
 - c. Page views
3. Platform/Device Statistics
- a. Datetime
 - b. Platform
 - c. Page views

These statistics are calculated using `/dashboard/calculation_scripts/systemStats.py` script.

Algorithm used to calculate system statistics:

1. Get all the logs which were stored on the previous day.
2. Calculate the distinct browsers, OS and Platforms and the page views from them.
3. Save the required field.

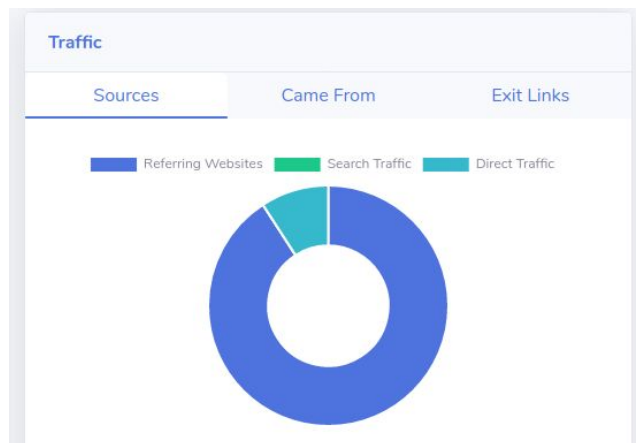
Sources Statistics

These statistics refer to the info about the sources such as referrals, search or direct traffic. This info contain:

1. Date
2. Direct page views
3. Search page views
4. Referrer page views

These statistics are calculated using `/dashboard/calculation_scripts/sourcesStats.py` script.

This info is used to display info on a pie chart at reports. The pie chart displays the percentage of traffics (search, referrals and direct).



Algorithm used to calculate sources statistics:

1. Get the total number of yesterday's page views (a) .
2. Get the total number of referring URLs from spoken website (b).
3. Get the total number of page views with no referring links (c), this is direct traffic.

4. Get the total number of page views having a referring link of search engines (d), this is search traffic.
5. Total referral page views will be $a - b - c - d$.
6. Save these logs.

Came From Statistics

These statistics refer to the info about the websites from where the visitors are coming to the website. This info include

1. Datetime
2. Referrer URL
3. Page views

These statistics are calculated using `/dashboard/calculation_scripts/cameFromStats.py` script.

Algorithm used to calculate came from statistics:

1. Get all the logs which were stored on the previous day.
2. Find distinct referring URLs (referrers).
3. For referrer in referrers
 - a. Find number of page views coming from referrer.
 - b. Save the required fields.

Exit Link Statistics

These statistics refer to the info about the websites to which visitors are going from spoken website. This info include

1. Datetime
2. Exit link
3. Number of time exit link clicked.

These statistics are calculated using `/dashboard/calculation_scripts/exitLinkStats.py` script.

Algorithm used to calculate exit link statistics:

4. Get all the logs which were stored on the previous day.
5. Find distinct exit link URLs (exit_links).
6. For exit_link in exit_links
 - a. Find the number of times the exit link exists in the logs.
 - b. Save the required fields.

Visualizing the statistics

As discussed earlier, to make the dashboard we used startbootstrap [sb-admin-2](#) template. I removed the unnecessary things from the template and used [Chart.js](#) library to display the charts on dashboard, page analytics and reports page.

Detail about each page is discussed in detail below :

Dashboard Page

URL: */dashboard*

This page displays the summary of statistics. It also contains a chart which shows the trends of page views, unique visits and returning visits versus time. It also has a datatable which shows the data which is displayed on chart in tabular form.

Tiles

Tiles are used to display the summary statistics like average daily, weekly, monthly and yearly page views, unique visits, first time visits and returning visits. The granularity (daily, weekly, monthly and yearly) can be changed from the dropdown placed above the chart.

Trend Chart

The data shown on the bar chart can be updated using Navigation buttons or date selects. Ajax requests are used to get data from the server. This chart can be changed to a line chart using chart type select.

Datatable

To display the data table I have used [JQuery datatable](#) plugin. The data of the data table can be exported using the EXPORT button.

Summary Stats

AVERAGE DAILY PAGE VIEWS
270

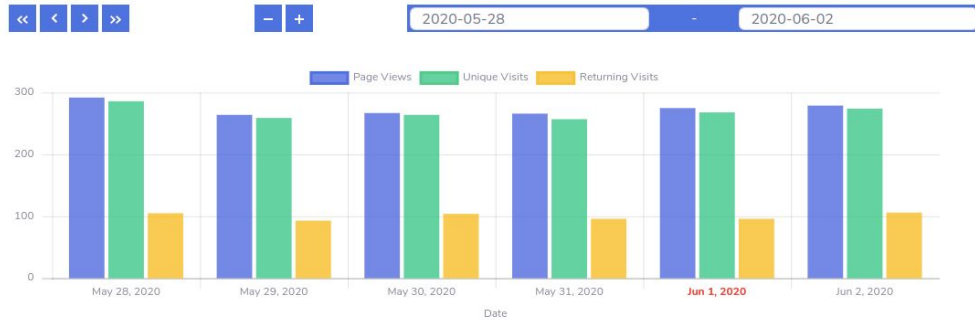
AVERAGE DAILY UNIQUE VISITS
264

AVERAGE DAILY FIRST TIME VISITS
166

AVERAGE DAILY RETURNING VISITS
97

Trend

Daily Bar Chart



Show

10 entries

EXPORT

	Page Views	Unique Visits	First Time Visits	Returning Visits
Tuesday June 2nd 2020	279	274	168	106
Monday June 1st 2020	275	268	172	96
Sunday May 31st 2020	266	257	161	96
Saturday May 30th 2020	267	264	160	104
Friday May 29th 2020	264	259	166	93
Thursday May 28th 2020	292	286	181	105

Showing 1 to 6 of 6 entries

Previous 1 Next

Dashboard page screenshot

Events Page

URL: /dashboard/events

The events page shows a datatable containing the page urls and number of unique visits of the pages between date range specified in date selects. These date selects can be varied to get different data. The datatable also contains Page Analysis buttons for each page which takes us to Page Analysis.

The screenshot displays the Spoken Analytics dashboard. On the left is a blue sidebar with navigation options: Dashboard, Pages, Foss, Activities, and Reports. The main content area is titled 'SPOKEN ANALYTICS' and 'Unique Visits Per Page'. It features a 'Show' dropdown set to '10' entries and date range selectors for '2020-06-02' to '2020-06-02'. Below this is a table with three columns: Page, Unique Visits, and Page Analysis. The table lists ten different pages with their respective unique visit counts and a 'Page Analysis' button for each. At the bottom, there is a pagination control showing 'Showing 1 to 10 of 40 entries' and a page number selector with '1' highlighted.

Page	Unique Visits	
India Map spoken-tutorial.org spoken-tutorial.org/statistics/online-test/	19	Page Analysis
Software Training Select Participants spoken-tutorial.org spoken-tutorial.org/statistics/academic-center/	17	Page Analysis
Academic Centers spoken-tutorial.org spoken-tutorial.org/software-training/resource-center/	16	Page Analysis
Online-Test Statistics spoken-tutorial.org spoken-tutorial.org/statistics/pmmnmtt/fdp/	15	Page Analysis
Academic Centers spoken-tutorial.org spoken-tutorial.org/accounts/register/	15	Page Analysis
India Map spoken-tutorial.org spoken-tutorial.org/software-training/student-batch/	15	Page Analysis
Logout spoken-tutorial.org spoken-tutorial.org/accounts/login/	14	Page Analysis
Logout spoken-tutorial.org spoken-tutorial.org/statistics/training/	14	Page Analysis
CD Content Creation spoken-tutorial.org spoken-tutorial.org/accounts/logout/	13	Page Analysis
Software Training Dashboard spoken-tutorial.org spoken-tutorial.org/cdcontent/	13	Page Analysis

Events page screenshot

Page Analysis

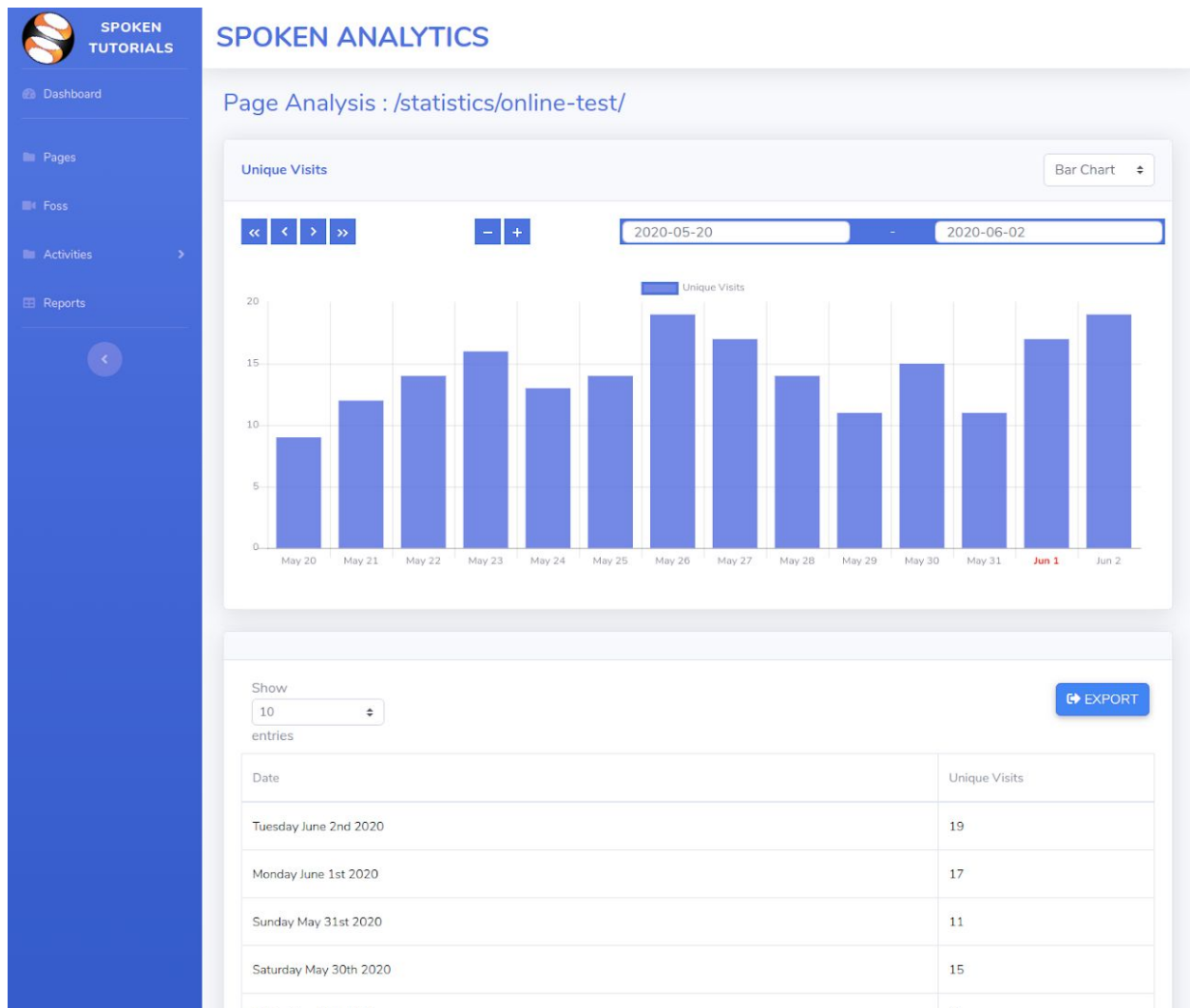
URL: /dashboard/event_analysis/?path=page_url

Trend Chart

Page Analysis has a chart along with navigation buttons and date select. The chart shows trend in the unique visits that the page got per day.

Datatable

It also contains a datatable which shows the data displayed on the chart in tabular form. The data of the datatable can be exported using the EXPORT button.

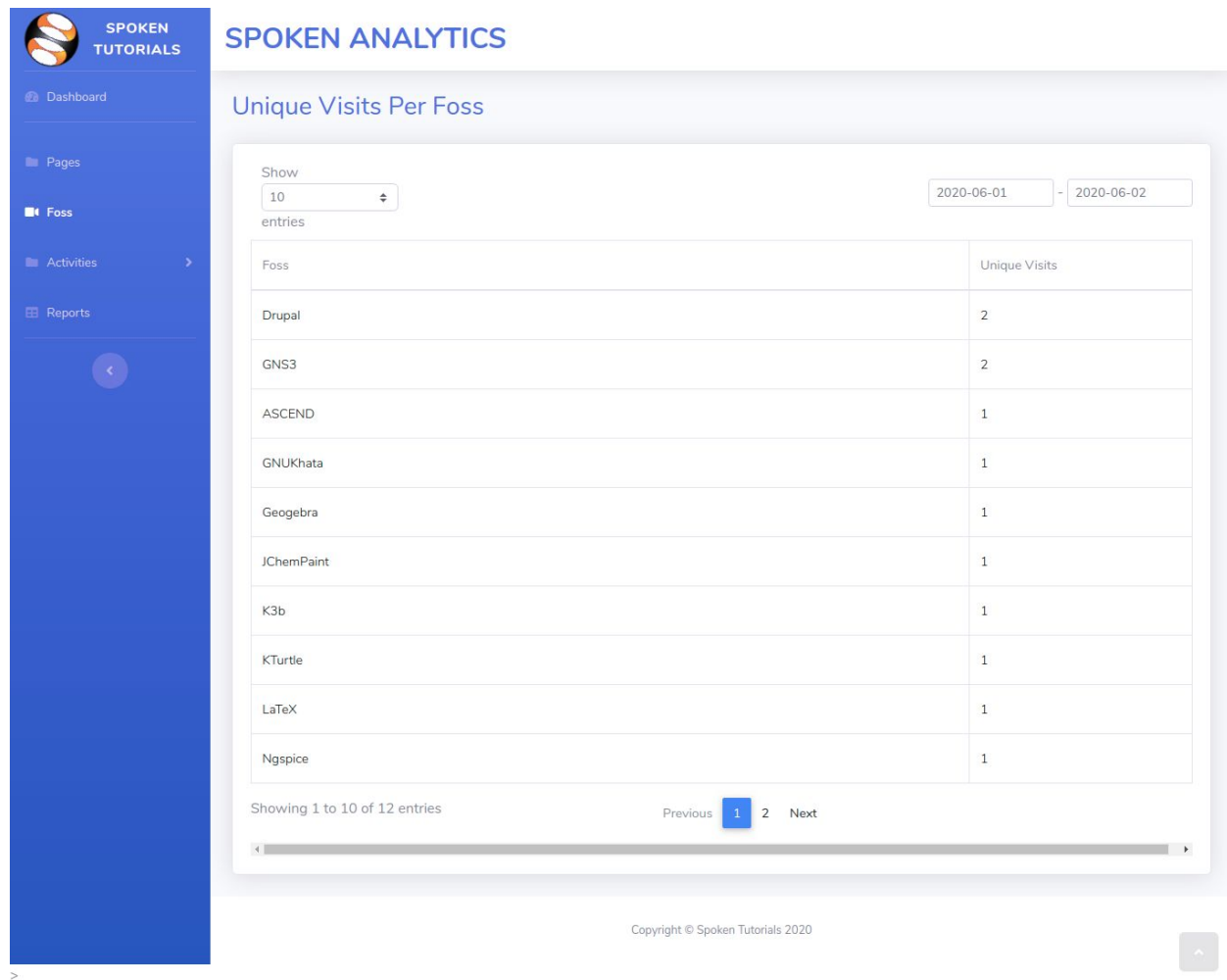


Page Analysis screenshot

Foss Page

URL: `/dashboard/foss`

The foss page shows a datatable containing the foss name and number of unique visits that each foss got between date range specified in date selects. These date selects can be varied to get different data.



The screenshot displays the SPOKEN ANALYTICS dashboard. On the left is a blue sidebar with navigation options: Dashboard, Pages, Foss, Activities, and Reports. The main content area is titled 'Unique Visits Per Foss'. It features a 'Show' dropdown set to '10' entries and two date range selectors for '2020-06-01' and '2020-06-02'. Below this is a table with two columns: 'Foss' and 'Unique Visits'. The table lists 12 entries, with the first 10 visible. At the bottom, there is a pagination control showing 'Showing 1 to 10 of 12 entries' and 'Previous 1 2 Next'.

Foss	Unique Visits
Drupal	2
GNS3	2
ASCEND	1
GNUKhata	1
Geogebra	1
JChemPaint	1
K3b	1
KTurtle	1
LaTeX	1
Ngspice	1

Foss page screenshot

Activities

Similar to statcounter which shows the *Recent Activities* (data of about last 5 minutes), this analytics system has *Activites* but using the date and time selects present in each activity page, we can see all the data present in the database. This is an advantage over statcounter.

The time select slider only allow to display 10 minutes data at a time. This restriction is places so that large amount of data is not fetched from database.

Visitor Activity Page

URL: `/dashboard/visitor_activity`

This page shows the following data about the visitor of the website

1. Page views
2. Total visits
3. Latest page view data and time
4. Location
5. Visit length
6. IP address
7. System Info (Browser, OS and device info)
8. Referrer URL
9. Entry Page
10. Exit Page

SPOKEN ANALYTICS

Visitor Activity

2020-06-02 05:30 - 05:40 VIEW

Show 10 entries

Page Views: 2	Total Visits: 2
Latest Page View: 2020-06-02 08:28	Location: Nashik, Jharkhand, India
Visit Length: 0 min 0 sec	IP Address: 230.124.0.197
System: Opera 81.0	Referring URL: android-app://com.google.android.gm/
Android 8	Entry Page: /statistics/
Lenovo K8 Note Mobile	Latest Page: /statistics/

Page Views: 2	Total Visits: 2
Latest Page View: 2020-06-02 09:09	Location: Aligarh, Tripura, India
Visit Length: 0 min 0 sec	IP Address: 230.124.0.239
System: Firefox 11.2	Referring URL: https://classroom.google.com/c/MTI0OTg2NTkwOTA0lp/MTAzMzY0NDEyNjk2/details
Linux 8.1	Entry Page: /accounts/register/
Realme XT Mobile	Latest Page: /statistics/tutorial-content/

Page Views: 2	Total Visits: 2
Latest Page View: 2020-06-02 17:05	Location: Bhopal, Punjab, India
Visit Length: 0 min 0 sec	IP Address: 230.124.0.251
System: Chrome 76.0	Referring URL: https://classroom.google.com/c/MTI0OTg2NTkwOTA0lp/MTAzMzY0NDEyNjk2/details
Android 8	Entry Page: /statistics/online-test/
Realme XT Tablet	Latest Page: /accounts/register/

Visitor Activity page screenshot

Visitor Path Page

URL: `/dashboard/visitor_path`

This page displayed the paths that the visitors take when the browse through the website. It shows the following data about a visitor:

1. Location
2. IP address
3. Visit number
4. System Info
5. Path

SPOKEN ANALYTICS

Visitor Path

2020-06-02 14:50 - 15:00 [VIEW](#)

Show: 10 entries Search:

Kalyan-Dombivali, Chandigarh, India 230.124.0.65		visit #1	Mac 8, Samsung Internet 81.0, Lenovo KB Note PC
2020-06-01	20:39		android-app://com.google.android.googlequicksearchbox/accounts/register/
2020-06-02	09:23		https://mail.google.com/mail/u/0/watch/Drupal/Tutorial 2/Marathi

Kolkata, Andhra Pradesh, India 230.124.0.25		visit #1	Android 7, Chrome 76.0, Samsung M31S Mobile
2020-06-02	06:52		https://storage.googleapis.com/uniquecourses/online.html/statistics/tutorial-content/
2020-06-02	09:29		https://classroom.google.com/u/0/c/MTI1NTA0MDMzNzk3/software-training/select-participants/

Showing 1 to 2 of 2 entries [Previous](#) **1** [Next](#)

Copyright © Spoken Tutorials 2020

Visitor Path page screenshot

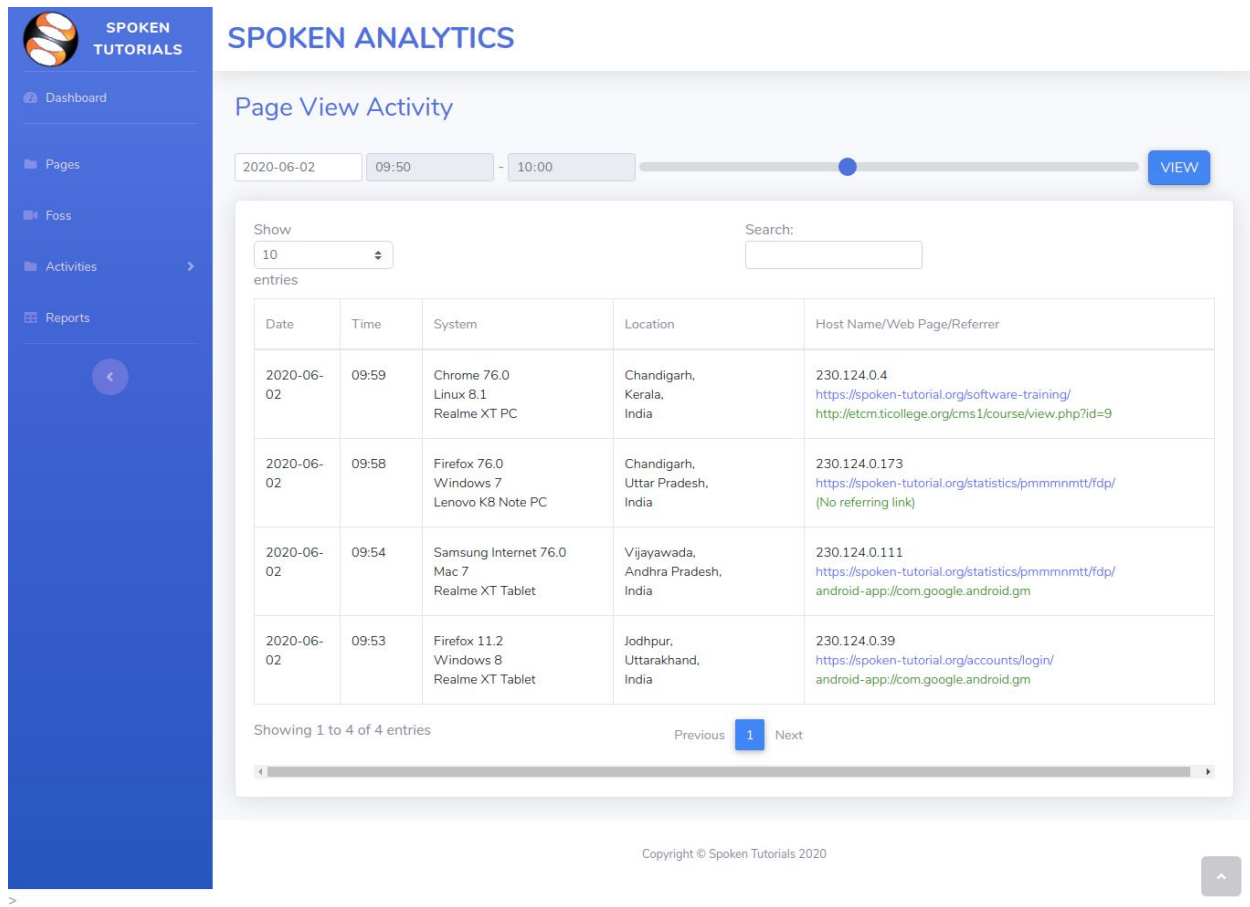
Page View Activity

URL: /dashboard/page_view_activity

This page displays info about pages. The info which is shown is:

1. Date
2. Time
3. System Info
4. Location
5. IP address
6. URL of Page viewed

7. Referrer URL



The screenshot displays the 'Page View Activity' dashboard in Spoken Analytics. The interface includes a sidebar with navigation options like Dashboard, Pages, Foss, Activities, and Reports. The main content area shows a date range filter (2020-06-02 to 10:00) and a 'VIEW' button. Below this is a table with 4 entries, each showing visitor details such as Date, Time, System, Location, and Host Name/Web Page/Referrer. The table data is as follows:

Date	Time	System	Location	Host Name/Web Page/Referrer
2020-06-02	09:59	Chrome 76.0 Linux 8.1 Realme XT PC	Chandigarh, Kerala, India	230.124.0.4 https://spoken-tutorial.org/software-training/ http://etcm.ticollege.org/cms1/course/view.php?id=9
2020-06-02	09:58	Firefox 76.0 Windows 7 Lenovo KB Note PC	Chandigarh, Uttar Pradesh, India	230.124.0.173 https://spoken-tutorial.org/statistics/pmmnmtt/fdp/ (No referring link)
2020-06-02	09:54	Samsung Internet 76.0 Mac 7 Realme XT Tablet	Vijayawada, Andhra Pradesh, India	230.124.0.111 https://spoken-tutorial.org/statistics/pmmnmtt/fdp/ android-app://com.google.android.gm
2020-06-02	09:53	Firefox 11.2 Windows 8 Realme XT Tablet	Jodhpur, Uttarakhand, India	230.124.0.39 https://spoken-tutorial.org/accounts/login/ android-app://com.google.android.gm

Showing 1 to 4 of 4 entries. Navigation: Previous 1 Next.

Copyright © Spoken Tutorials 2020

Page View Activity screenshot

Came From Activity

URL: `/dashboard/came_from_activity`

This page shows the website URLs from which the visitors are coming to the spoken website. It shows info like

1. Date
2. Time
3. Referrer
4. Entry Page

SPOKEN ANALYTICS

Came From Activity

2020-06-02 18:40 - 18:50 [VIEW](#)

Show 10 entries

Date	Time	Referrer	Entry Page
2020-06-02	18:47	https://classroom.google.com/u/1/c/MTAyNzM2MzQ2NjY0	https://spoken-tutorial.org/software-training/student-batch/
2020-06-02	18:44	android-app://com.google.android.gm/	https://spoken-tutorial.org/statistics/academic-center/
2020-06-02	18:42	android-app://com.google.android.gm	https://spoken-tutorial.org/statistics/training/

Showing 1 to 3 of 3 entries [Previous](#) **1** [Next](#)

Copyright © Spoken Tutorials 2020

Came From Activity page screenshot

Exit Link Activity

URL: /dashboard/exit_link_activity

This page shows the links to where the visitors are going from the spoken website. The info displayed on this page include:

1. Date
2. Time
3. Exit Link
4. Exit Page

SPOKEN ANALYTICS

Exit Link Activity

2020-06-02 03:40 - 03:50 VIEW

Show 10 entries

Date	Time	Exit Link Clicked	Exit Page
2020-06-02	03:46	https://www.youtube.com/user/SpokenTutorialITB/	https://spoken-tutorial.orghttps://spoken-tutorial.org/
2020-06-02	03:48	http://application.reimagine-education.com/the-winners-individual/2015/132/2193b0ae3841f24da1464d4b6b70ee0f/Indian Institute of Technology Bombay	https://spoken-tutorial.orghttps://spoken-tutorial.org/stfellowship2020/

Showing 1 to 2 of 2 entries Previous 1 Next

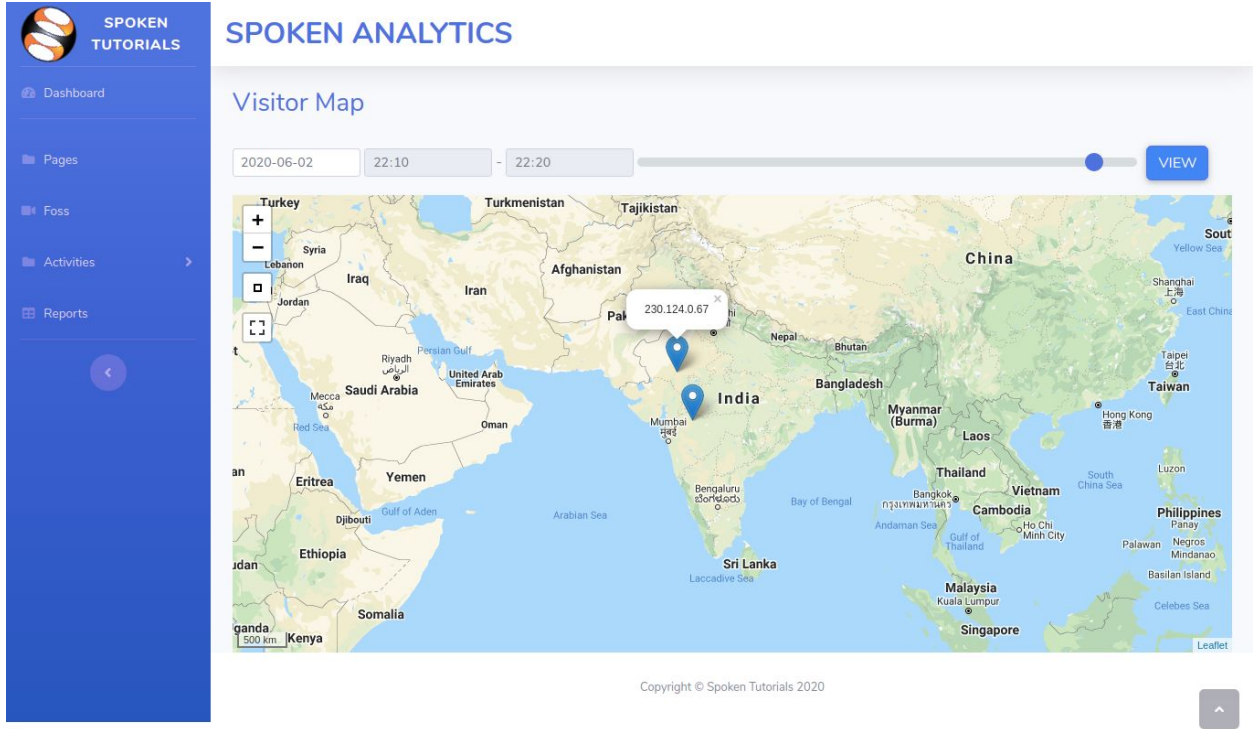
Copyright © Spoken Tutorials 2020

Exit Link Activity page screenshot

Visitor Map

URL: /dashboard/visitor_map

This shows the location of visitors on the map. To show the map [django-leaflet](#) and [django-geojson](#) packages are used.



Visitor Map page screenshot

Visitor Info

URL: </dashboard/magnify/?ip=230.124.0.181>

This page shows info about the visitor of the website. The info include:

1. Referring URL
2. System Info
3. IP address
4. Location
5. Returning visit
6. Visit Length
7. Navigation Path

The statcounter also shows the location of the user on Map, which is not included in this system.

The screenshot shows the Spoken Analytics dashboard. On the left is a blue sidebar with navigation options: Dashboard, Pages, Foss, Activities, and Reports. The main content area is titled 'Magnify IP Address 230.124.0.67'. Below this title is a section 'Visitor Analysis & System Spec' with the following details:

- Referring URL: <https://classroom.google.com/u/0/c/MTI1NTA0MDMzNzk3>
- Browser: Opera 68.0
- IP Address: 230.124.0.67
- Location: Rajkot, Nagaland, India
- Returning Visits: 0
- Visit Length: 0 min sec
- OS/Platform: Windows 7
- Device: Samsung M40 Tablet

Below this is a 'Navigation Path' table:

Date	Time	WebPage
3 Jun	03:15:42	https://classroom.google.com/u/0/c/MTI1NTA0MDMzNzk3/statistics/pmmmmmtt/fdp/

At the bottom of the dashboard, there is a copyright notice: 'Copyright © Spoken Tutorials 2020' and a small upward arrow icon.

Visitor Info page screenshot

Reports Page

URL: */dashboard/reports*

This page shows the reports such as

Location Report

It shows the top ten regions and cities from where the most number page views are requested. The city/region name is shown in a tabular format along with the percentage of page views. The data about all the locations can be seen using the View Report button.

Foss/Page Report

It shows the top ten foss and pages which are getting most number page views. The foss name/page title is shown in a tabular format along with the percentage of page views. The data about all the foss and pages can be seen using the View Report button.

Note: This report is different from Event Page (*/dashboard/events*) and Foss Page (*/dashboard/foss*), because report show percentage of page views rather that unique visits.

System Report

It shows the top ten browsers / OS / Platform(device), from where the most page views are requested. The browser / OS / Platform names are shown in tabular format along with the percentage of page views. All the data can be seen using the View Report button.

Traffic Report

It shows a doughnut chart displaying the percentage of traffic from referring websites, Search Engine traffic and Direct traffic. It also shows the top ten came from links and exit links along with click count. The detailed report can be seen using the View Report button.

SPOKEN TUTORIALS

- Dashboard
- Pages
- Foss
- Activities
- Reports

SPOKEN ANALYTICS

Reports

Traffic from locations

State/Region	City	
Uttarakhand		3.41%
Karnataka		3.26%
Jharkhand		3.26%
Ladakh		3.23%
Chhattisgarh		3.19%
Dadra and Nagar Haveli and Daman & Diu		3.08%
Tamil Nadu		3.05%
Punjab		3.05%
Chandigarh		3.01%
Madhya Pradesh		2.97%

[View Report](#)

Traffic on Foss and Pages

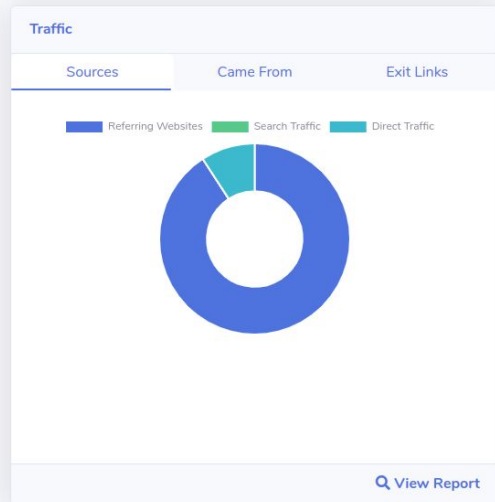
Foss	Pages	
test		11.14%
R		4.18%
C		3.94%
Python		2.78%
ExpEYES		2.55%
Selenium		2.55%
K3b		2.32%
JChemPaint		2.32%
LaTeX		2.32%
GNUKhata		2.32%

[View Report](#)

System

Browser	Platform	OS
Firefox 83.0		5.61%
Chrome 83.0		5.61%
Opera 83.0		5.54%
Chrome for Android 11.2		5.44%
Samsung Internet 11.2		3.79%
Firefox 11.2		3.79%
Firefox 68.0		3.77%
Samsung Internet 81.0		3.76%
Chrome for Android 81.0		3.74%
Firefox 81.0		3.74%

[View Report](#)



Reports Page screenshot

Future Work

Most of the functionalities that the statcounter provides are implemented in this system. Some of the functionalities that are remaining are

1. Keyword Activity
2. Download Activity
3. Location Stats by countries
4. Engagement Report

The HTML templates, views and url paths are included in the respective files of dashboard app and can be used to implement Keyword Activity and Download Activity.

The statistics calculating scripts can be further improved so that they take less time in calculating the stats.

The system is also lacking tests, which can be included in future versions.

Conclusion

This analytics system can be used as the analytics system of the spoken website. There are some limitations like displaying real time statistics and it is missing some functionalities. This system can be improved in the future versions.

At last, I would like to thank FOSSEE and the Spoken Tutorials project for providing me the opportunity to work on this project. I would like to thank my mentor, Sir Abhijit Bonik, for providing guidance and help whenever needed. I would also like to thank Ma'am Nancy Varkey and Ma'am Kirti Ambrey for periodically reviewing my work.