**PROJECT OPEN SOURCE**

**Objective:** Analyze the number patches submitted (committed done) by a specific user/development team to Gerrit, Android, GitLab. Which can be used by high level managers to analyze the performance of employees and teams.

**Big Idea:**

Opensource contribution helps to reduce the cost and the maintenance by leveraging learnings in the community. Periodically generating such reports helps to understand the development teams’ contributions and align for future corrections. Expanding it to larger organizations and automated dashboards help to reduce manual efforts.

**My Audience:**

**Primary**

Leadership, customers and opensource communities

**Secondary**

Senior management

**Tools Required:**

1. Excel
2. Jupyter Notebook
3. Power BI

**What does my Audience care about:**

Team contribution to open-source community and tracking of work done. multiple employees are working in a team contributing across multiple opensource repositories by submitting patches(code commit/merge) in repositories. We need to track the employee performance by analyzing the number of patches submitted and size of code submitted, Number of commits done per month, number of patches approved/Abandoned/merged by reviewer.

**Important questions are:**

1. How many commits are done by a specific user/team in each branch?

2.How many changes were approved and merged to the main branch?

3.Most recent commit?

4.How many patches were Abandoned?

5. Trend of commits by month/year?

6. How many patches were Merged and how many are open?

**Approach Followed**

1. Writing the names/email id of the person/team in an Excel spreadsheet.

2. Writing the python code to read the names from excel spreadsheet and get the data from corresponding websites.

3. Data from Gerrit, Gitlab and Android websites were separately collected from corresponding Web Api URL/End points.

4. Used Python code to hit the URL and get the data for each status (open, merged, Abandoned)

5. The web response was converted to a Data frame and only required columns were selected and stored in a data frame.

6. Finally the data frame was exported to Excel file in python by giving the file path you want to be stored.

7. The file path from Excel as a source is connected to the Power BI.

9. Necessary cleaning of the file is done in Power Query in Power BI.

10. Created visualizations which help to get insights required from the management.

Excel file which has information about the team

Python Code to read from Excel.

Visualizing in Power BI by connecting to Excel

Saving the data in an Excel file.

Collecting data from open source by hitting API.

**Sample Dashboard**

A close-up of a graph

Description automatically generated