A Web Application to Analyze GitHub Data

Ashish Lakamale (18210227)
Nivedha Nagarajan(18210700)

Introduction

The GitHub is a most widely used cloud-based publishing tool and a hosting platform. We have created an application which analyses the data of GitHub using Google's BigQuery. The cloud technology we have used is Apache Spark where we have done certain implementations on the data and have visualized them using plotly. A dynamic user interface is been created using Dash and the interactive visualizations are been displayed.

Web Application

The web application can be found at http://gharchive35.herokuapp.com/

Source Code

The source code for our implementation is uploaded to GitHub. It can be found using this link https://github.com/harshalchaudhari35/Github_Archive.

Hosting

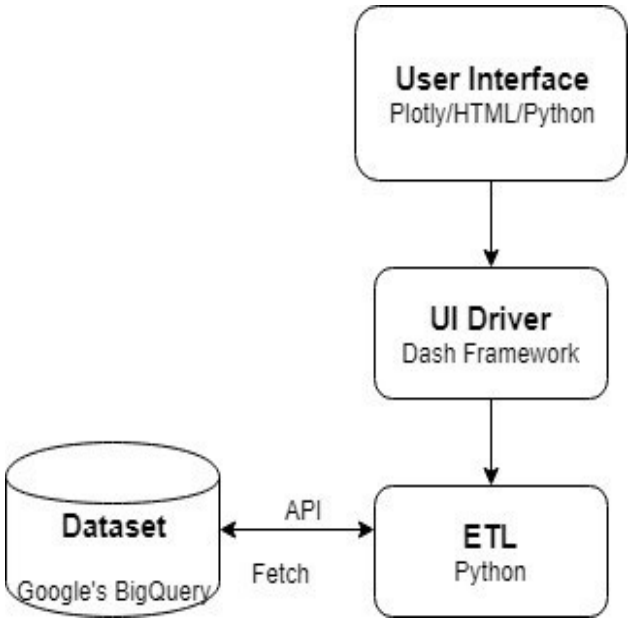
The **main.py** file consists of the dash framework where the necessary installations are done, and the data which was extracted and queries on queries.py file is fetched. This file (main.py), when executed gives a path/ link which can be hosted on the local machine. Also, we have deployed our web application on Heroku cloud where it provided as an application in order to upload all our implementation files.

Queries

- **Trending languages over time:** The line graph is made interactive where the data changes can be seen over time.
- **Sizes of Github Repos on Head Branch in MBs:** The graph shows the repositories along with their sizes.
- **Trending Github Repositories:** The horizontal bar chart shows the top trending reposon GitHub.
- **Language Popularity Score:** The vertical bar graph shows the popular languages based on the scores.

- **Top Java Repositories by their commits count:** The bar graph shows the top repositories in Java which has the greatest number of commits.

Flowchart



User Interface

The Plotly library in python was used to visualize the data which were queried using BigQuery. ***Main.py*** is the file which has the source code for visualizations of Github analytics. The source code of UI is available in the link for the Github repo. The queries mentioned above are queried and extracted from respective tables using bq_helper and are stored separately which can be seen in ***queries.py*** file. This data is called in main.py and the corresponding visualizations are done using plotly library in python.

UI Driver

The **Dash framework** along with Flask was used to create a user interface. The visualizations are embedded in the Dash framework. **main.py** also has the source code ofhow the dash framework was created and how the visualization obtained by querying using bq_helper is embedded in the framework.

Extract, Transform & Load

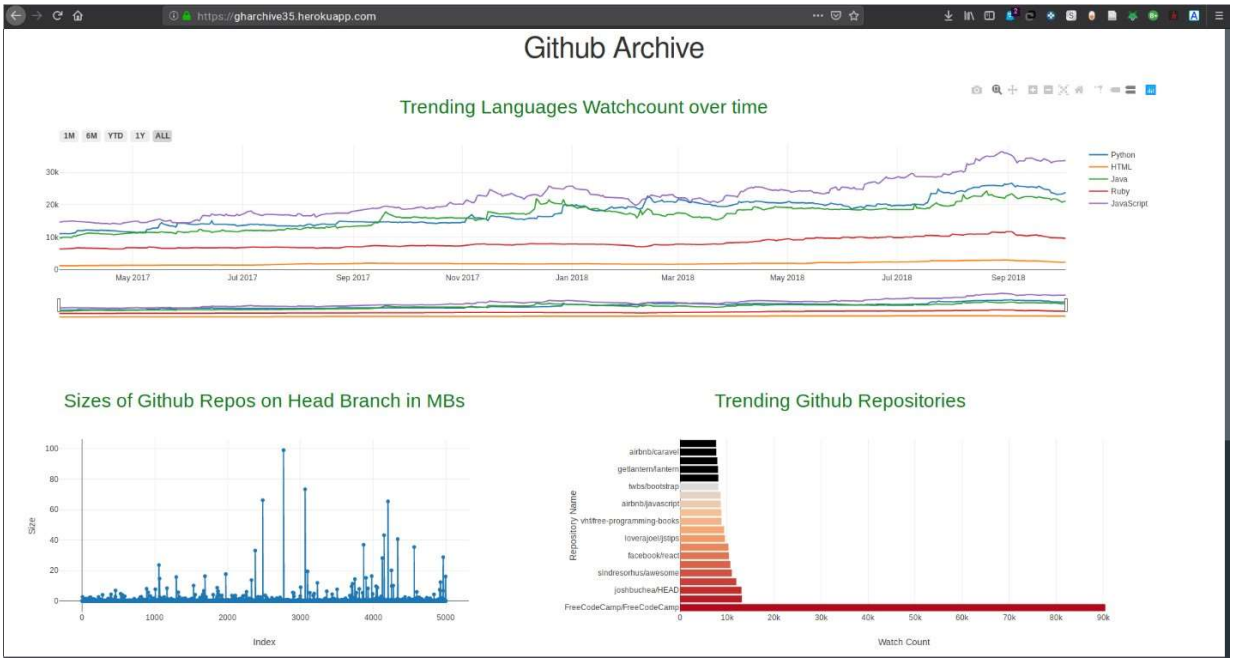
The dataset was extracted with the help of BigQuery using python. The necessary installations and imports were done for data extraction. The required data is pulled and extracted using an API key. There were in total of 5 queries which were used to obtain the required data. We have obtained the information which are mentioned above on the queries section such as trending languages, repos etc. and have visualized it interactively.

Dataset

The dataset which we use is GH Archive which can be obtained from https://www.gharchive.org/. It is a project to record the public GitHub timeline, archive it and make it easily available for further analysis. There are more than 20 event types, commits and fork events which also includes commenting and adding members to a project. **Google's BigQuery** is used to obtain the dataset using an API key.

Web Application

We have created a dynamic web application which has the GitHub data visualizations. This application is deployed in Heroku cloud. The image below shows the screenshot of the application.



Screencast video link

The link for our screencast video: https://youtu.be/rHhVBR1dNfE

Peer Feedback Response

Apache Spark was mentioned to be one of our key technologies which we had an idea of using. But we had the best, cleaner and an easiest way of fetching the data using BigQuery which we found to be time saving as well. Also, the suggestion to use Apache Pig was not needed asthe data which we obtained through BigQuery was clean and no further implementations to clean the data was required. We have considered the problems and the challenges which our peer group felt that we might face and have sorted it accordingly. The linking of application with data was done using Dash and python and the data is queried to get the necessary information for our visualizations.

Challenges and lessons learned

Steps were taken to make future predictions using pyspark. But the dataset we obtained cannot be used as we encountered few challenges on which features to make the predictions. In addition to that, fetching the dataset initially from GitHub was a challenge. Later we came to know that BigQuery is the easiest and fastest way to obtain the dataset.

Responsibility Statement.

All our team members shared the work equally and have made equal contributions to the project. Each of the task right from querying the dataset to deploying the application was a teamwork and the percentage of contribution can be seen below.

Individual Marking Group L

Student	**Mark(%)**
Ashish Lakamale	50
Nivedha Nagarajan	50