PROJECT REPORT

on

TWITTER SENTIMENT ANALYSIS

SUBMITTED FOR FULFILLMENT OF AWARD OF

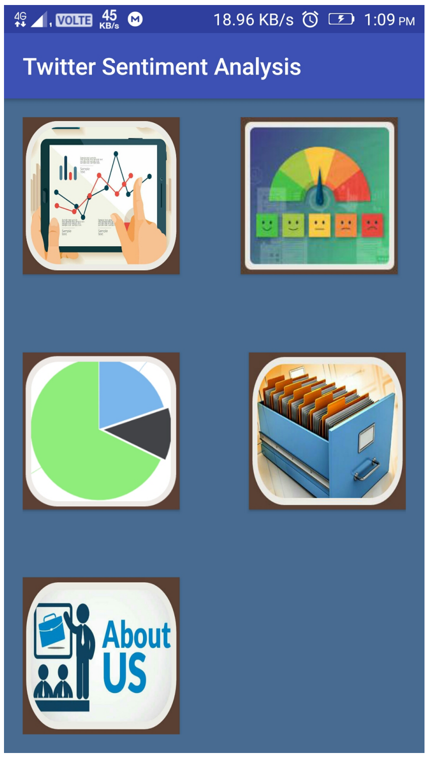
BACHELOR OF TECHNOLOGY
(Computer Science And Engineering)

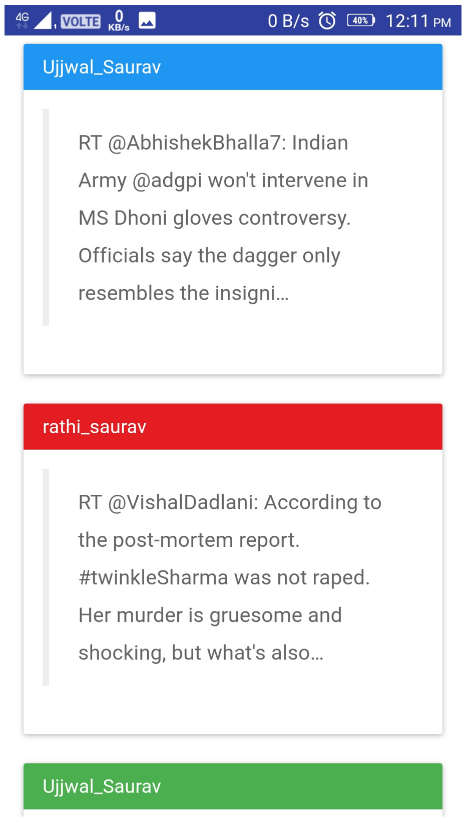
ANDROID APP

Menu

The interface for menu is the Activity_main navigation is optimized for features that is app provides The interface is based on a custom Button view that allows to choose displaying feautes on the main screen and allows switching between different kinds of features.

PREVIEW





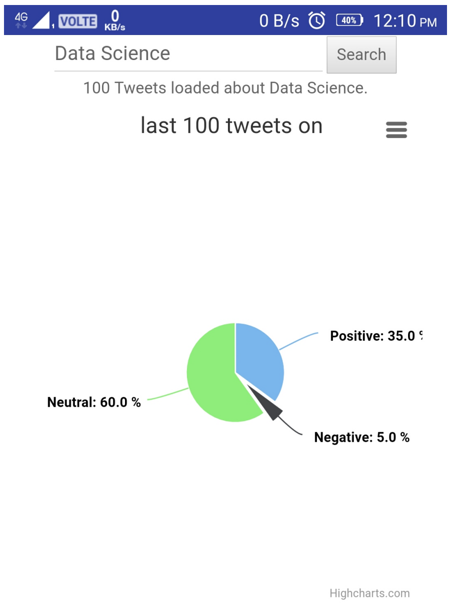
Ujjwal_Saurav

RT @AbhishekBhalla7: Indian Army @adgpi won't intervene in MS Dhoni gloves controversy. Officials say the dagger only resembles the insigni…

rathi_saurav

RT @VishalDadlani: According to the post-mortem report. #twinkleSharma was not raped. Her murder is gruesome and shocking, but what's also…

Ujjwal_Saurav



Data Science Search

100 Tweets loaded about Data Science.

last 100 tweets on

Features

Splash Screen

A splash screen is a graphical control element consisting of a window containing an image, a logo, and the current version of the software. A splash screenusually appears while a game or program is launching.

Big analysis

Big analysis is the big data mining tool which mines data upto 10000 tweets. It gives the precise result of over all tweets.

Sentiment Tweets Analysis

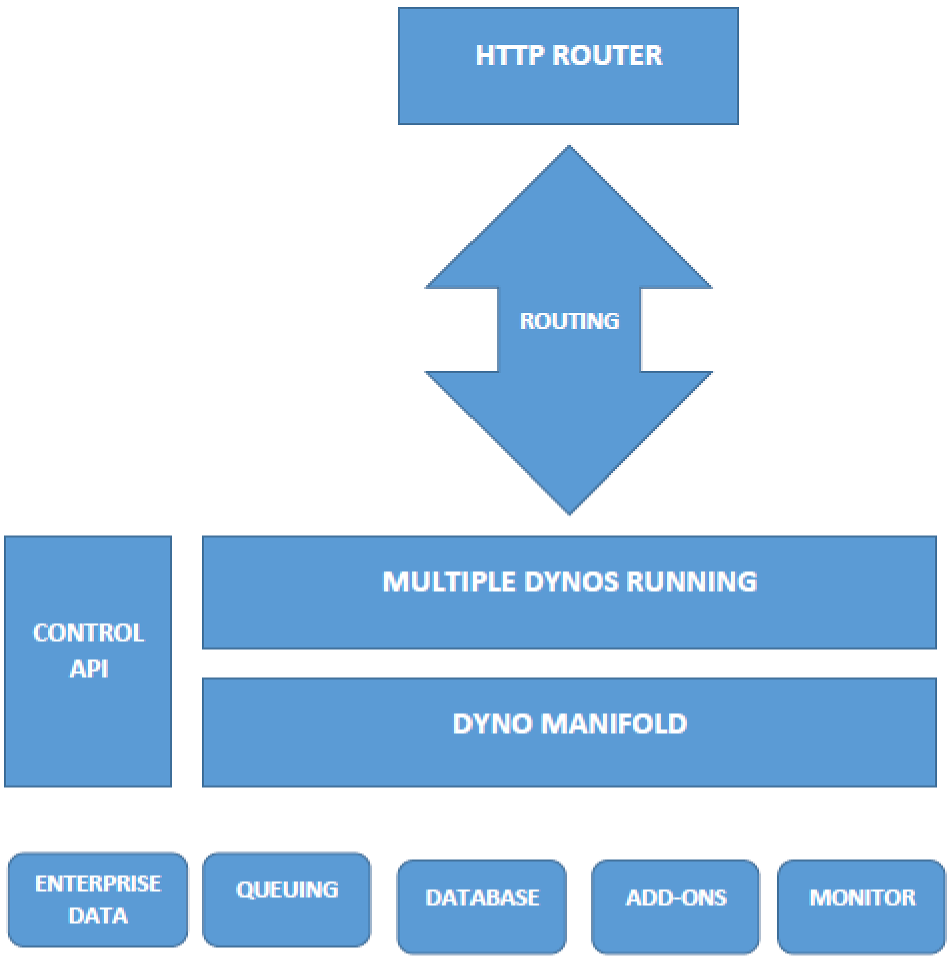
Sentiment Analysis is the tool which shows the Sentiment of specific tweet.

Chart view

Chart view is the Optimizes the view of chart around 100 tweets.

External Server Details

Heroku is a cloudplatform as a service(PaaS) supporting several programming languages. Heroku, one of the firstcloud platforms, has been in development since June 2007, when it supported only theRubyprogramming language, but now supportsava,Node.js,Scala,Clojure,Python,PHP, andGo.



Applications that are run on Heroku typically have a unique domain (typically "*applicationname.herokuapp.com*") used to route HTTP requests to the correct dyno. Each of the application containers, or *dynos*, are spread across a "dyno grid" which consists of several servers. Heroku's Git server handles application repository pushes from permitted users.

Deploy

•The main content of the development are the source code, related dependencies if they exist, and a Procfile for the command.
•The application is sent to Heroku using either of the following: Git, GitHub, Dropbox, or via an API.
•There are packets which take the application along with all the dependencies, and the language runtime, and produce slugs. These are known as build-packs and are the means for the slug compilation process.
•A slug is a combination/bundle of the source code, built dependencies, the runtime, and compiled/generated output of the build system which is ready for execution.
•Next is the Config vars which contain the customizable configuration data that can be changed independently of the source code.
•Add-ons are third party, specialized, value-added cloud services that can be easily attached to an application, extending its functionality.
•A release is a combination of a slug (the application), config vars and add-ons.
•Heroku maintains a log known as the append-only ledger of releases the developer makes.

Runtime

•The main unit which provides the run environment are the Dynos which are isolated, virtualized unix containers.
•The application's dyno formation is the total number of currently-executing dynos, divided between the various process types the developer has scaled.
•The dyno manager is responsible for managing dynos across all applications running on Heroku.
•Applications that use the free dyno type will sleep after 30 minutes of inactivity. Scaling to multiple web dynos, or a different dyno type, will avoid this.
•One-off Dynos are temporary dynos that run with their input/output attached to the local terminal. They're loaded with the latest release.

•Each dyno gets its own ephemeral filesystem with a fresh copy of the most recent release. It can be used as temporary scratchpad, but changes to the filesystem are not reflected to other dynos.
•Logplex automatically collates log entries from all the running dynos of the app, as well as other components such as the routers, providing a single source of activity.
•Scaling an application involves varying the number of dynos of each process type.
A detailed description of the architecture involves:

Define the application

The definition of the application i.e. the source code and the description is built on the framework provided by Heroku which converts it into an application. The dependency mechanisms vary across languages: for Ruby the developer uses a Gemfile, in Python a requirements.txt, in Node.js a package.json, in Java a pom.xml, and so on.

Knowing what to execute

Developers don't need to make many changes to an application in order to run it on Heroku. One requirement is informing the platform as to which parts of the application are runnable. This is done in a Procfile, a text file that accompanies the source code.[23] Each line of the Procfile declares a process type — a named command that can be executed against the built application.

Deploying applications

Application development on Heroku is primarily done through git. The application gets a new git remote typically named as Heroku along with its local git repository where the application was made. Hence to deploy heroku application is similar to using the git push command.

There are many other ways of deploying applications too. For example, developers can enable GitHub integration so that each new pull request is associated with its own new application, which enables all sorts of continuous integration scenarios. Dropbox Sync lets developers deploy the contents of **Dropbox** folders to Heroku, or the Heroku API can be used to build and release apps.

Deployment then, is about moving the application from a local system to Heroku.

Building applications

The mechanism for the build is usually different for different languages, but follows the consistent pattern of retrieving the specified dependencies, and creating any necessary assets (whether as simple as processing style sheets or as complex as compiling code). The source code for the application, together with the fetched dependencies and output of the build phase such as generated assets or compiled code, as well as the language and framework, are assembled into a slug.

Running applications on dynos

Applications in Heroku are run using a command specified in the Procfile, on a dyno that's been preloaded with a prepared slug (in fact, with the release, which extends the slug, configuration variables and add-ons).

It's like running dyno[22] as a lightweight, secure, virtualized Unix container that contains the application slug in its file system. Heroku will boot a dyno, load it with the slug, and execute the command associated with the web process type in the Procfile. Deploying a new version of an application kills all the currently running dynos and starts new ones (with the new release) to replace them, preserving the existing dyno formation.

Configurations

A customization of the existing configuration is possible as the configuration is done not within the code but in a different place outside the source code. This configuration is independent of the code currently being run. The configuration for an application is stored in config vars.

At runtime, all of the config vars are exposed as environment variables so they can be easily extracted programatically. A Ruby application deployed with the above config var can access it by calling ENV["ENCRYPTION_KEY"]. All dynos in an application will have access to exactly the same set of config vars at run-time.

Releases

The combination of slug and configuration is called a release. Every time a new version of an application is deployed, a new slug is created and release is generated.

As Heroku contains a store of the previous releases of the application, it's designed to make it easier to roll back and deploy a previous release. A release, then, is the mechanism behind how Heroku lets the developer modify the configuration of the application (the config vars) independently of the application source (stored in the slug) — the release binds them together. Whenever the developer changes a set of config vars associated with the application, a new release will be generated.

Dyno manager

Dyno manager help maintain and operate the dynos created. Because Heroku manages and runs applications, there's no need to manage operating systems or other internal system configuration. One-off dynos can be run with their input/output attached to the local terminal. These can also be used to carry out admin tasks that modify the state of shared resources, for example database configuration, perhaps periodically through a scheduler.

Add-ons

Dynos do not share file state, and so add-ons that provide some kind of storage are typically used as a means of communication between dynos in an application. For example, Redis or Postgres could be used as the backing mechanism in a queue; then dynos of the web process type can push job requests onto the queue, and dynos of the queue process type can pull jobs requests from the queue. Add-ons are associated with an application, much like config vars, and so the earlier definition of a release needs to be refined. A *release*of the applications is not just the slug and config vars; it's the slug, config vars as well as the set of provisioned add-ons.

Logging and monitoring

Heroku treats logs as streams of time-stamped events, and collates the stream of logs produced from all of the processes running in all dynos, and the Heroku platform components, into the Logplex- a high-performance, real-time system for log delivery. Logplex keeps a limited buffer of log entries solely for performance reasons.

HTTP routing

Heroku's HTTP routers distribute incoming requests for the application across the running web dynos. A random selection algorithm is used for HTTP/HTTPS request load balancing across web dynos. It also supports multiple simultaneous connections, as well as timeout handling.