Indian Railways Twitter Complaints Prioritisation

By

Srijan Prakash (1602713110)
Rishitosh Kumar Singh (1602710114)
Nikki Rastogi (1602710082)
Silki Gupta (1602713107)

Submitted to the Department of Computer Science & Engineering in partial fulfillment of the requirements for the degree of

Bachelor of Technology
in
Computer Science & Engineering

Ajay Kumar Garg Engineering College,Ghaziabad
Dr. APJ Abdul Kalam Technical University, Lucknow

YEAR: 2019-2020

TABLE OF CONTENTS

List of Figures

Declaration

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature :
Name :Srijan Prakash
Roll No : 1602713110
Date :

Signature :
Name : Rishitosh Kumar Singh
Roll No : 1602710114
Date :

Signature :
Name :Nikki Rastogi
Roll No : 1602710082
Date :

Signature :
Name :SIlki Gupta
Roll No : 1602713107
Date :

Certificate

This is to certify that Project Report entitled **"Indian Railways Twitter Complaints Prioritisation"** which is submitted by **Srijan Prakash (1602713110), Silki Gupta (1602713107), Rishitosh Kumar Singh (1602710114), Nikki Rastogi (1602710082)** in partial fulfillment of the requirement for the award of degree B. Tech. in Department of Computer Science and Engineering of Dr. APJ Abdul Kalam Technical University,is a record of the candidate own work carried out by him under our supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

Supervisor

Mrs. Lipika Goel
(Assitant Professor)
CSE Department

Date:

Acknowledgement

We would like to express our sincerest gratitude to all the people who have contributed towards the successful completion of our project. We would like to extend our heartfelt thanks to the Head of Computer Science and Engineering Department Dr. Sunita Yadav, for nurturing a congenial yet competitive environment in the department, which motivates all the students to pursue higher goals. We want to express our special gratitude to our guide "Mrs. Lipika Goel ", Department of Computer Science and Engineering, Ajay Kumar Garg Engineering College, Ghaziabad for her constant support, guidance, encouragement and much needed motivation. Her sincerity, thoroughness and perseverance has been a constant source of inspiration for us. Last but not the least, we would like to extend our thanks to all the teaching and non teaching staff members of our department, and to all our colleagues who helped us in completion of the project.

Signature :
Name :Srijan Prakash
Roll No : 1602713110
Date :

Signature :
Name : Rishitosh Kumar Singh
Roll No : 1602710114
Date :

Signature :
Name :Nikki Rastogi
Roll No : 1602710082
Date :

Signature :
Name :SIlki Gupta
Roll No : 1602713107
Date :

1. Introduction

The project reduces the work complexity of scanning through thousands of useless data to find particular information but from here we can directly find the relevant tweets that needs attention. To solve the problem, we will be using a Machine Learning (Naive Bayes) model. In machine learning, Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. They are among the simplest Bayesian network models. Naïve Bayes has been studied extensively since the 1960s. It was introduced (though not under that name) into the text retrieval community in the early 1960s, and remains a popular (baseline) method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines. It also finds application in automatic medical diagnosis. Also, the machine learning model will be flexible and can be applied on various other local authorities like Nagar Nigam, State electricity board and various other authorities. To do that we just have to filter the tweets on the basis of that particular authority. It will reduce the operational time.

1.1 Background

The Indian Railways currently receives close to 5,000 Tweets a day. The Tweets are monitored 24×7 and responded to by their level of seriousness or urgency. By this logic a complaint of sexual harassment or threat to personal safety feature on top as do medical emergencies and dealt with very quickly. Second come complaints against the Railway management, uncooperative personnel, meals, hygiene or matters that need immediate attention within trains or in stations. Last come complaints and requests that are not of immediate importance and can be solved later.

1.2 Purpose

- We try to implement and improve the current working model to a better machine learning model in the future development and also try to work with new cutting edge technology stack.
- We have tried to achieve increased fault-tolerance. We also want to increase the accuracy and reliability of the current Indian Railway Twitter Complaint Registration System.
- By the use of machine learning, we have tried to increase the operational efficiency as the tweets that were not attended due to lack of manual labour, will now be addressed as our model will not skip any tweet, and will classify the tweet into feedback and emergency, which will lead to better decision making by the Indian Railway.

- We have also increased Data Security as manual labour is not involved so no unwanted user can read all such information.
- We tried to automate all this process so that the human resource of the government can be saved and same resource can be utilized in a better way on something else.
- This software package can be readily used by non-programming personal avoiding human handled chance of error.

1.3 Scope of the project

This project has a large scope as it has the following features which help in making it easy to use, understand and modify it:

- Automation of Feedback and complaint addressing. No Need to do Paper Work.
- To save the environment by using paper free work.
- To increase the accuracy and efficiency of the complaint addressing procedure. Management of passenger data.
- This software package can be readily used by non-programming personal avoiding human handled chance of error.

1.4 Significance

- Simplified Management of complaint addressing.
- Job Posting.
- Feedback and Emergency Results.
- Real-time Information Publishing through system alerts.

2. Requirement Analysis and Feasibility Study

2.1 Requirement Gathering

In this phase, we try to find out what are the requirements that need to be fulfilled by the proposed system, what all aspects we need to cover so as to make this system worth of usage.

2.1.1 Information Gathering

To collect all the information related to the project we have to specify that what is required and what are the specifications of the project. What we want to done with our project means what are the functionalities we want to provide. Information gathering is done in order to understand the problem statement and collect the useful information. Now for our project, we collect all the necessary information that is tweets. We also collect basic information about our user, PNR number of the train, tweet id.

2.1.2 Functional Requirements

User: The user should have a twitter account.If the user does not have a twitter account he/she can signup by filling the some particular details like name,mail id,etc.The user can tweet with @RailMinIndia if he/she have can complain or feedback regarding Indain Railway.

Dataset: A data set (or dataset, although this spelling is not present in many contemporary . dictionaries) is a collection of data. Most commonly a data set corresponds to the contents of a single database table,or a single statistical data matrix, where every column of the table represents a particular variable, and each rowcorresponds to a given member of the data set in question. The dataset lists values for each of the variables, such as height and weight of an object, for each member of the data set. Each value is known as a datum. The data set may comprise data for one or more members, corresponding to the number of rows.

App module: This module contain admin module,login module,reply module.
Admin module- The person who reply for the tweets.
Login module- The admin can only login into the app with particular login id and password.
Reply Module- To reply the tweets in real time.

Web module: This module contain admin module,login module,reply module.
Admin module- The person who reply for the tweets.
Login module- The admin can only login into the app with particular login id and password.
Reply Module- To reply the tweets in real time.

2.1.3 Non Functional Requirements

2.1.3.1 Essential Requirement

- The system should have 97 percentage of reliability.
- The system should have a backup schedule every day.
- The data in the system should provide security using encryption method.
- The system should update the scheme availabilities real time.

2.1.3.2 Optional Requirement

- The system should be able to process each request within 2 seconds.
- The system can retrieve the data since created.

2.1.3.3 Software Quality Attributes

Scalability

When designing for scalability the primary goal is to ensure efficient resource management. Designing for scalability is not limited to any particular tier or component of an application. Application architects must consider scalability at all levels, from the user interface to the data store. Following are the factors which need to be considered when designing for Scalability:

1. Prioritization of services- In initial roll out the main services or the services which are immediately required are introduced with the system rollout and remaining services will be plugged in to system in phased manner as and when required. The Architectural framework should be designed to cater this aspect.
2. Classes of service - As stated above the Architecture should also accommodate to increase or decrease the number of classes, in order to achieve more services.
3. Dynamic change support- The system architecture should be capable to support the dynamic change into the system.
4. Transparent resource addition - The system architecture should be capable to support the transparent resource addition with the system.

Reliability

Reliability of a system is a measure of the ability of a system to keep operating overtime. It is typically measured as its mean time between failures (MTBF), expected type of system. Delivery of data to intended recipient (notification to sender by email or message). Reliability service – notify user – if delivery fail. Together TCP IP provide reliable service. Vaccinate has the ability to behave consistently in a user acceptable manner when operating within the environment for which the system was intended.

Availability

It is apt that resources that should be available to authorized user actually are available (Safety

& dependability). A faulty associate with availability is denial of service attack degree to which system is in specified operable and committable state at the start. It will operate satisfactorily at given point in time. Ideal support environment includes corrective maintenance downtime.

Maintainability

Following are the factors which need to be considered while require time to add a Component /Service / Module to an Existing Application.

1. Architecture and Component Framework - The Architecture is based on open standards take less time to add the Component / Service / Module to an Existing Application.

2. Classes and services- It is very important to understand how the classes and services has been developed and configured in the application framework. These should be created in such a way that any future requirement or addition of services etc helps to introduce new services easily.

3. Access and Roles- This service must be compatible enough to take care of the whole hierarchy of the departments which are going to be included. In case the flexibility in design has not been taken in cognizance it may create the problem at later stage and take longer time to add the service into the architecture.

2.2 Feasibility Study

Here, we have studied all the feasibility aspects of the project under consideration to check out if the project is feasible with the decided requirements and availability of information, technologies, and budget.

2.2.1 Operational Feasibility

Operational feasibility refers to the measure of solving problems with the help of a new proposed system. It helps in taking advantage of the opportunities and fulfills the requirements as identified during the development of the project. It takes care that the management and the users support the project. The proposed Real time Indian Railways twitter Complaint Administration System is capable identifying the emergency tweets more quickly and is more flexible and efficient than the existing systems. The machine learning and bid data has made the system more congenial to end user.

2.2.2 Technical Feasibility

The system as mentioned is developed using the latest technologies which include Kafka ,Spark ,Zookeeper ,Xampp ,Cloud, Flutter, Big data. It uses a Machine Learning model trained with the dataset extracted from various sources. Furthermore, the Navie Bayes Algorithm is used for the classification of data into emergency and feedback .

2.2.3 Economic Feasibility

Cost-benefit analysis is very important in deciding whether the project is economically feasible or not. It is alone sufficient to save time and money. Through cost-benefit analysis, it is quite evident that the benefits of this system outweigh costs and thus the project is economically feasible.

2.2.4 Behavioral Feasibility

Behavioral feasibility determines how much effort will go into educating, selling and training the user staff on a candidate system. This project is evaluated to be behaviorally feasible as it is very user-friendly and hardly needs any extra efforts to educate user for its facility and functioning. This is a one-time setup project and once initially installed, requires no intervention of the user.

3. System Analysis and Design

3.1 System Analysis

Systems analysis is a problem solving technique that decomposes a system into its component pieces for the purpose of the studying how well those component parts work and interact to accomplish their purpose". According to the Merriam-Webster dictionary, systems analysis is "the process of studying a procedure or business in order to identify its goals and purposes and create systems and procedures that will achieve them in an efficient way". Analysis and synthesis, as scientific methods, always go hand in hand; they complement one another. Every synthesis is built upon the results of a preceding analysis, and every analysis requires a subsequent synthesis in order to verify and correct its results.

3.1.1 Existing System Description

The Indian Railways currently receives close to 5,000 Tweets a day. The Tweets are monitored 24×7 and responded to by their level of seriousness or urgency. By this logic a complaint of sexual harassment or threat to personal safety feature on top as do medical emergencies and dealt with very quickly. Second come complaints against the Railway management, uncooperative personnel, meals, hygiene or matters that need immediate attention within trains or in stations. Last come complaints and requests that are not of immediate importance and can be solved later.

Whenever a passenger (Twitter user) tweets with the required tags, The application at Indian Railways receives a request. This request is automated by twitter and sent to Indian Railways. Once the application (A software) receives such request, it generates a 'Ticket' in the support team. There is a whole big team sitting in front of their Computers to help you resolve your complaints. So this Ticket gets assigned to one of the support team members (Let us call him 'A') and it pops up on his computer.As soon as the person 'A' receives the ticket, he has to act on it within a particular amount of time.

3.1.2 Proposed System

The project reduces the work complexity of scanning through thousands of useless data to find particular information but from here we can directly find the relevant tweets that needs attention. To solve the problem, we will be using a Machine Learning (Naive Bayes) model. In machine learning, Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features.

The proposed system will divide the tweets in two catogery that is emergency and feedback.The manual work done by the employes will be reduced .The tweets will be replied in the reat time.

This system will improve the –

Accuracy and reliability
Reduced operational time
Fault-tolerance

3.2 System Design

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. System Analysis is the process that decomposes a system into its component pieces for the purpose of defining how well those components interact to accomplish the set requirements. The purpose of the System Design process is to provide sufficient detailed data and information about the system and its system elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.

3.2.1 ER –Diagram

An entity–relationship model (or ER model) describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between entities (instances of those entity types). It is usually drawn in a graphical form as boxes (entities) that are connected by lines (relationships) which express the associations and dependencies between entities. An entity-relationship diagram is a data modelling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure. While useful for organizing data that can be represented by a relational structure, an entity-relationship diagram can't sufficiently represent semi-structured or unstructured data, and an ERD is unlikely to be helpful on its own in integrating data into pre-existing information system. Three main components of an ERD are the entities, which are objects or concepts that can have data stored about them, the relationship between those entities, and the cardinality, which defines that relationship in terms of numbers.

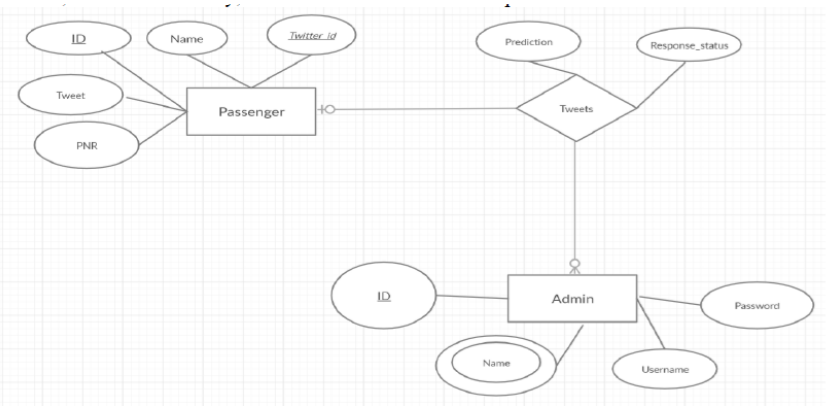

Figure 3.1: E-R Diagram

3.2.2 Data Flow Diagram

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

3.2.2.1 Level 0 DFD:

Its shows the abstract view of the system. Context Diagrams and DFD Layers and Levels. A context diagram is a top level (also known as "Level 0") data flow diagram. It only contains one process node ("Process 0") that generalizes the function of the entire system in relationship to external entities.

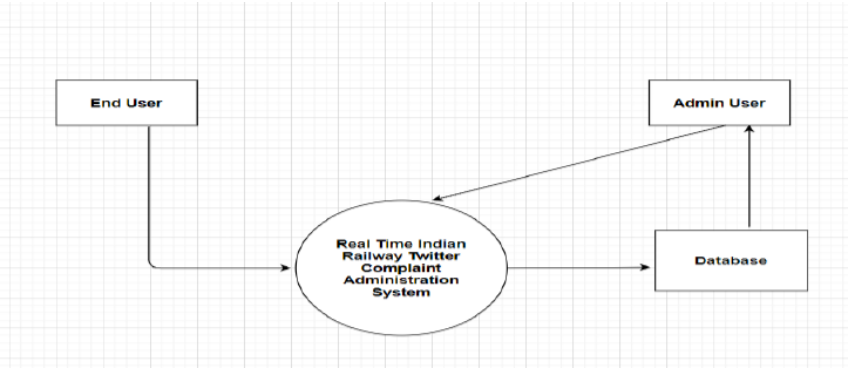

Figure 3.2: Level 0 DFD

3.2.2.2 Level 1 DFD:

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, we try to describe the system using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper. It explains the system in detailed view. When drawing Context Level DFD's, we must first identify the process, all the external entities and all the data flows. We must also state any assumptions we make about the system. It is advised that we draw the process in the middle of the page. We then draw our external entities in the corners and finally connect our entities to our process with the data flows.

3.2.2.3 Level 2 DFD

2-level DFD goes one step deeper into parts of 1-level DFD.It can be used to plan or record the specific/necessary detail about the system's functioning.

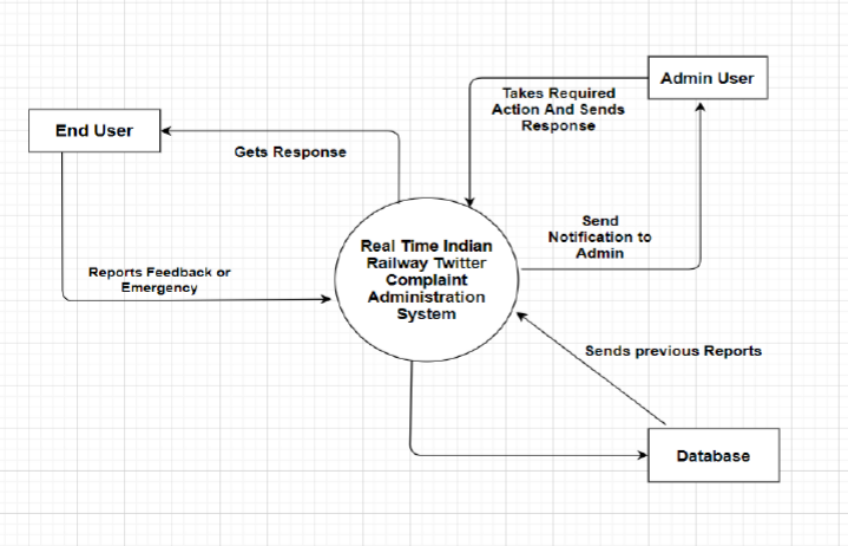

Figure 3.3: Level 1 DFD

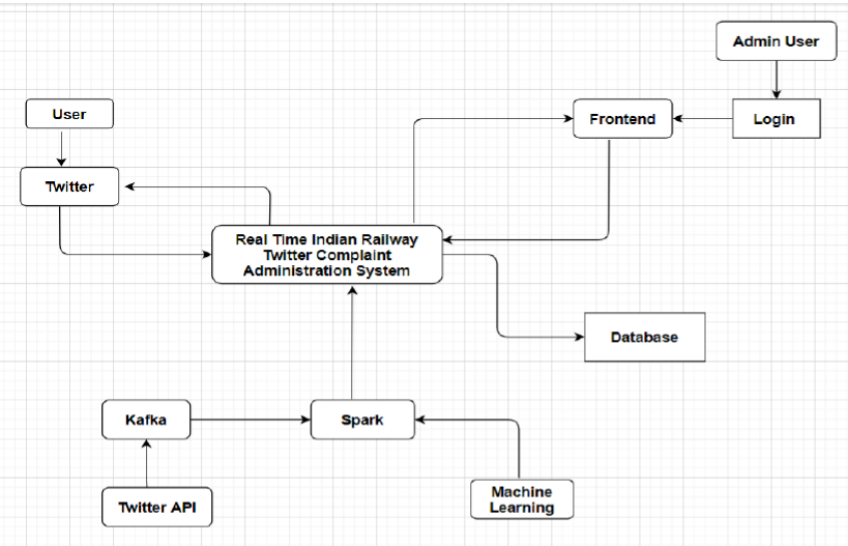

Figure 3.4: Level 2 DFD

3.2.3 Use case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. Simplest representation of a user's interaction that shows the relationship between the user and the different use cases in which user involved.

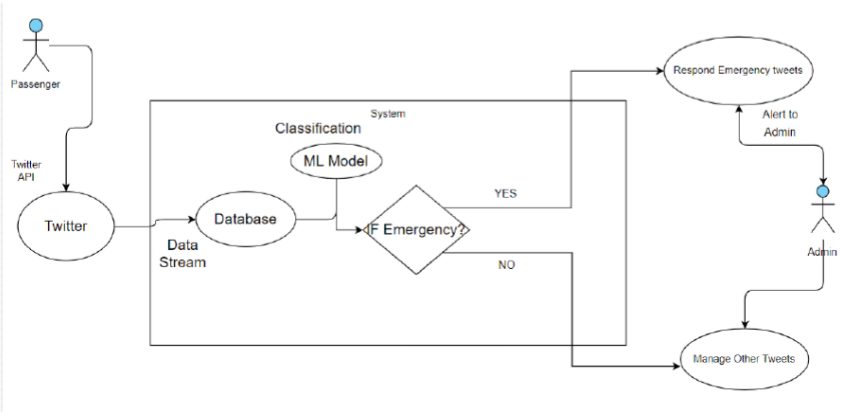

Figure 3.5: Use Case Diagram

3.2.4 Sequence Diagram

Sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

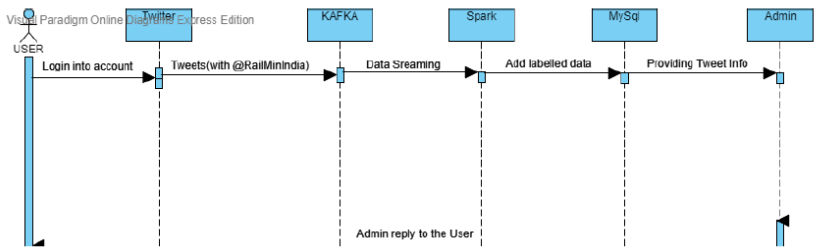

Figure 3.6: Sequence Diagram

3.3 Snapshots

3.3.1 Data and machine learning pipeline

Figure 3.7: Data and machine learning pipeline snapshots

Figure 3.8: Data and machine learning pipeline snapshots

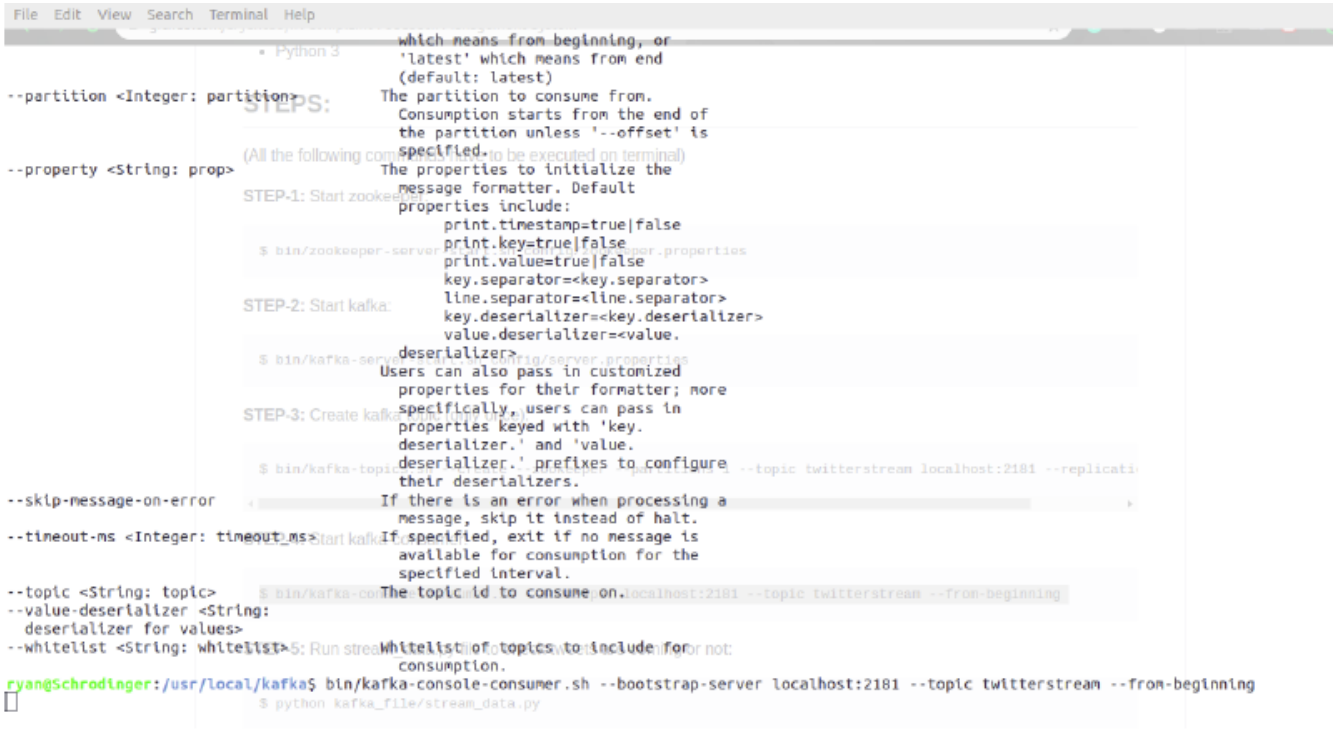

Figure 3.9: Data and machine learning pipeline snapshots

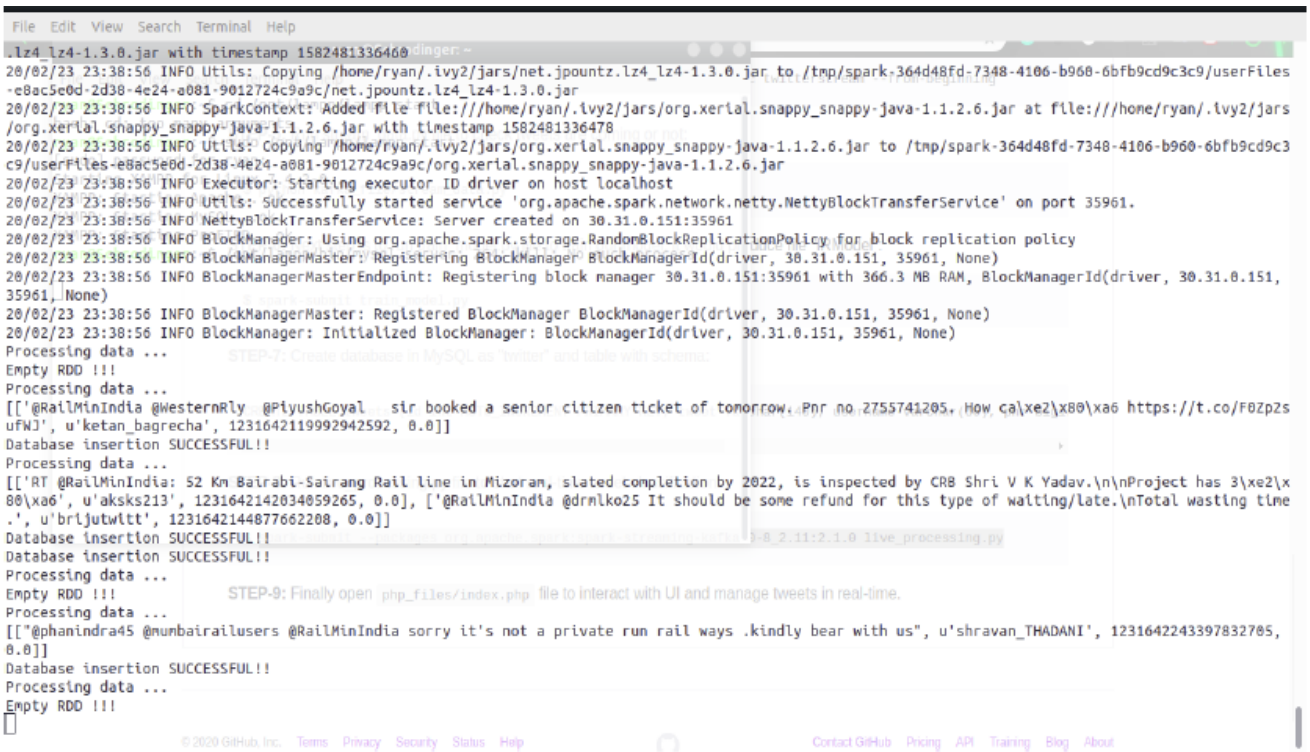

Figure 3.10: Data and machine learning pipeline snapshots

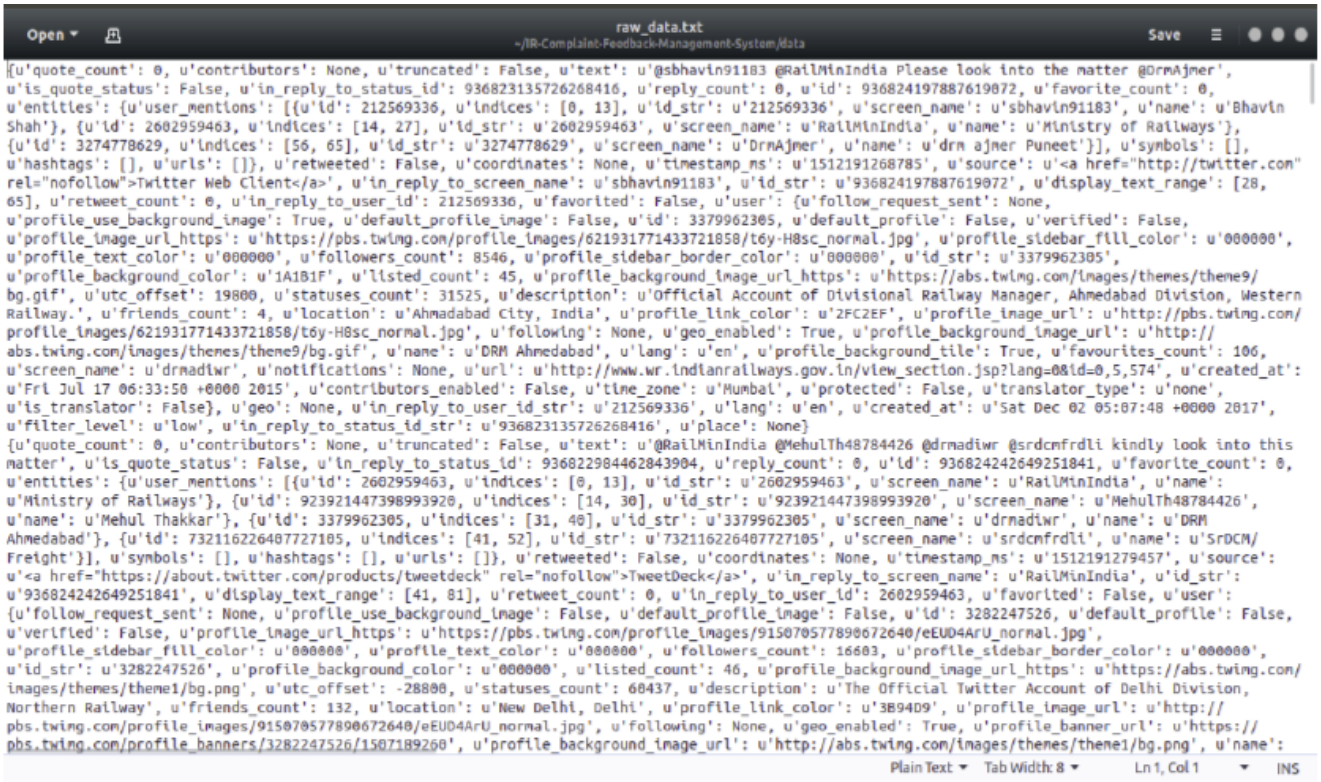

Figure 3.11: Data and machine learning pipeline snapshots

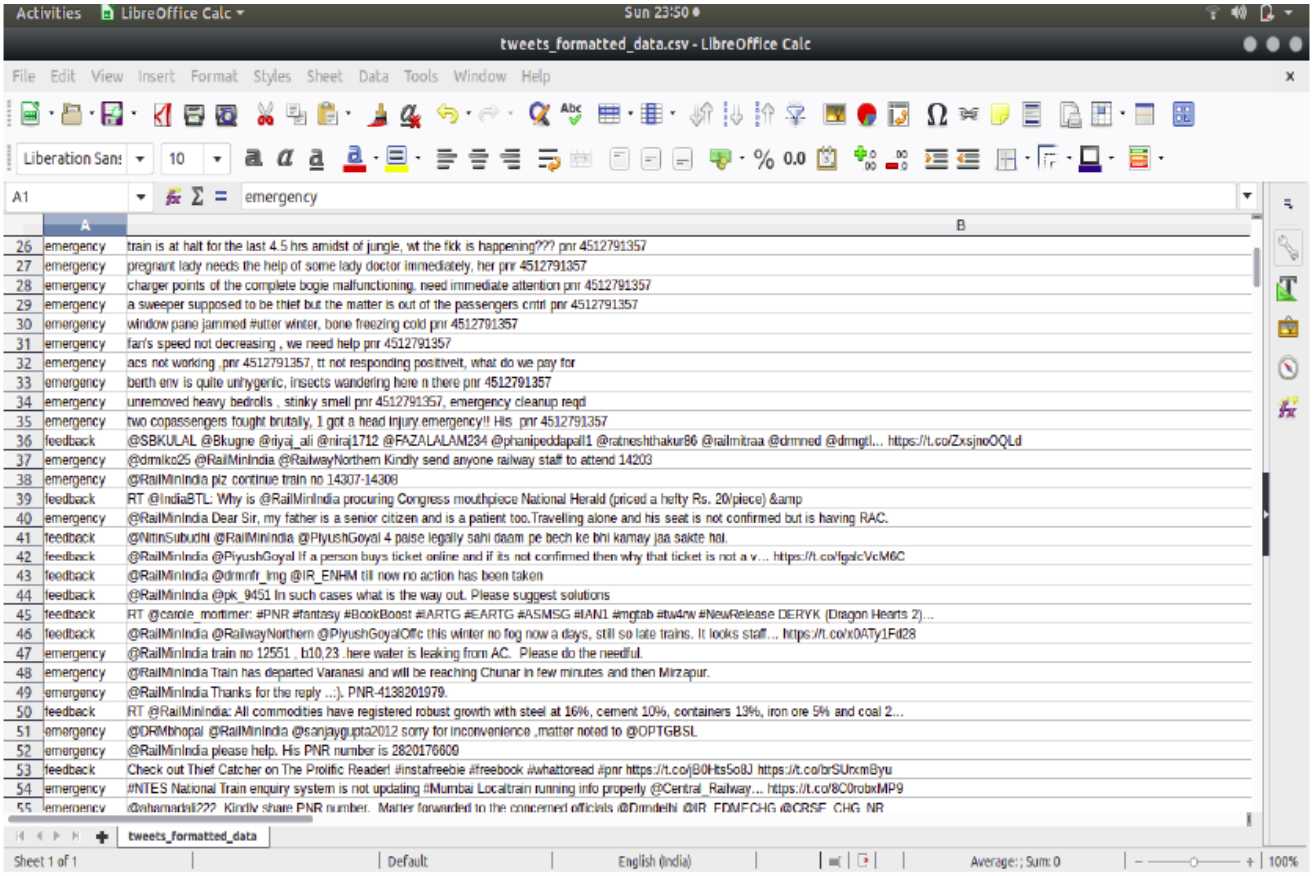

Figure 3.12: Data and machine learning pipeline snapshots

Figure 3.13: Data and machine learning pipeline snapshots

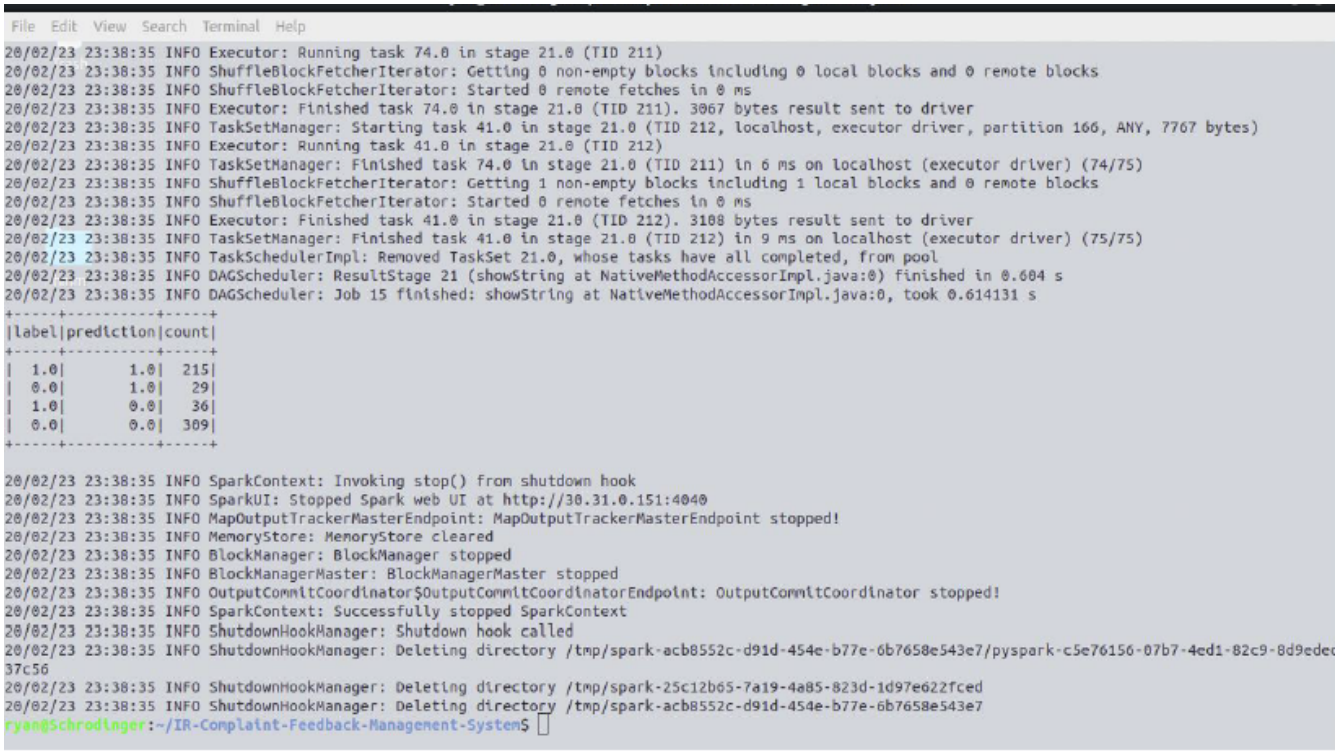

Figure 3.14: Data and machine learning pipeline snapshots

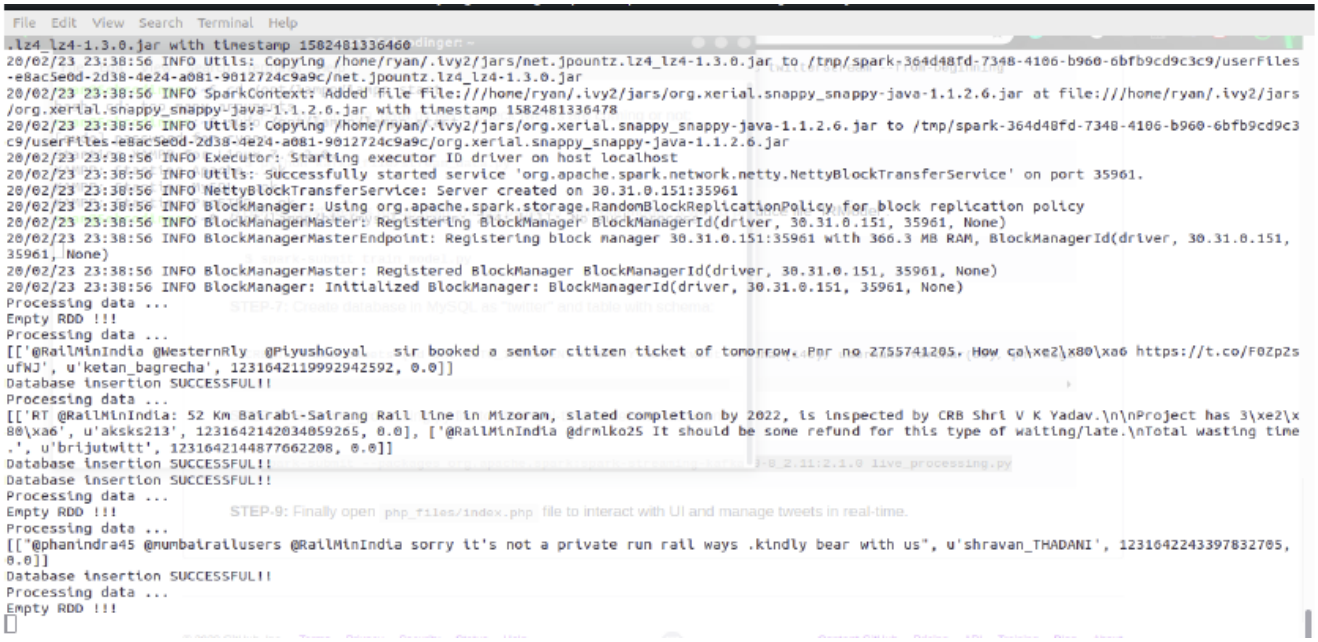

Figure 3.15: Data and machine learning pipeline snapshots

3.3.2 Database

MongoDB is an open-source document database and leading NoSQL database. MongoDB is written in C++. MongoDB compass creates and deploys a highly scalable and performance-oriented database.MongoDB is a cross-platform, document oriented database that provides, high performance, high availability, and easy scalability. MongoDB works on concept of collection and document.Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database.A document is a set of key-value pairs. Documents have dynamic schema.

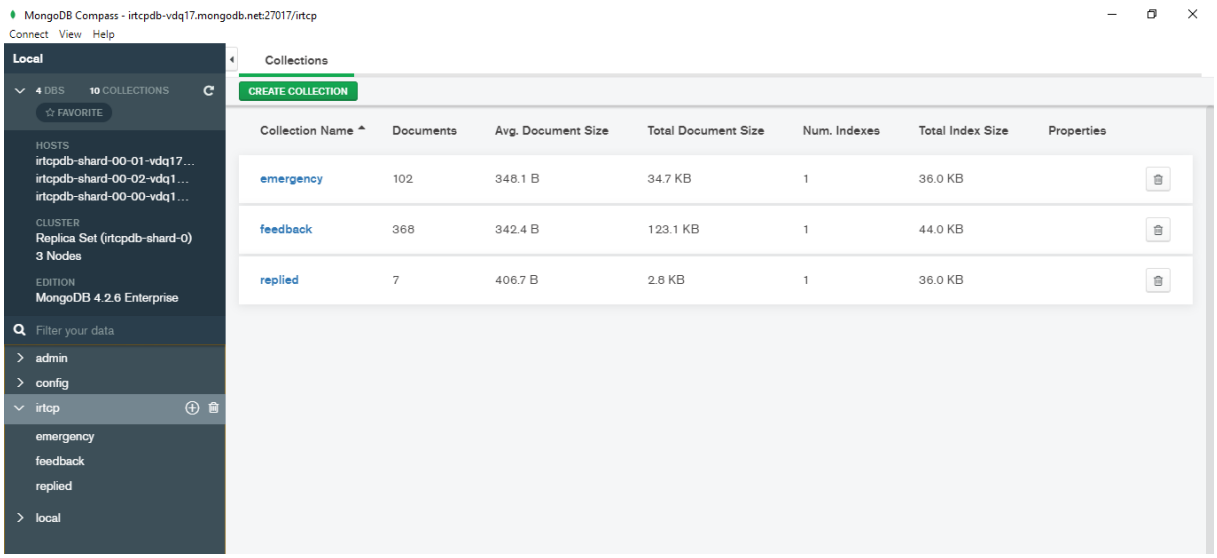

Figure 3.16: Database Snapshot

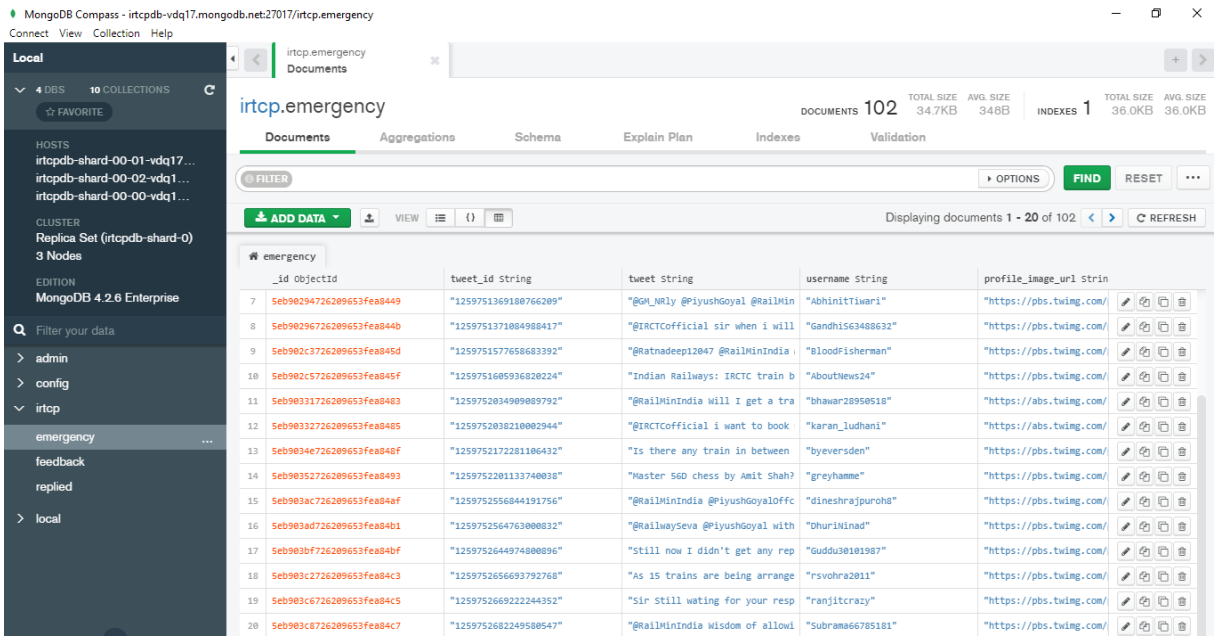

Figure 3.17: Database snapshots

3.3.3 App Module

There are Mainly three sub-modules present:

1. Emergency section
2. Feedback section
3. Replied section

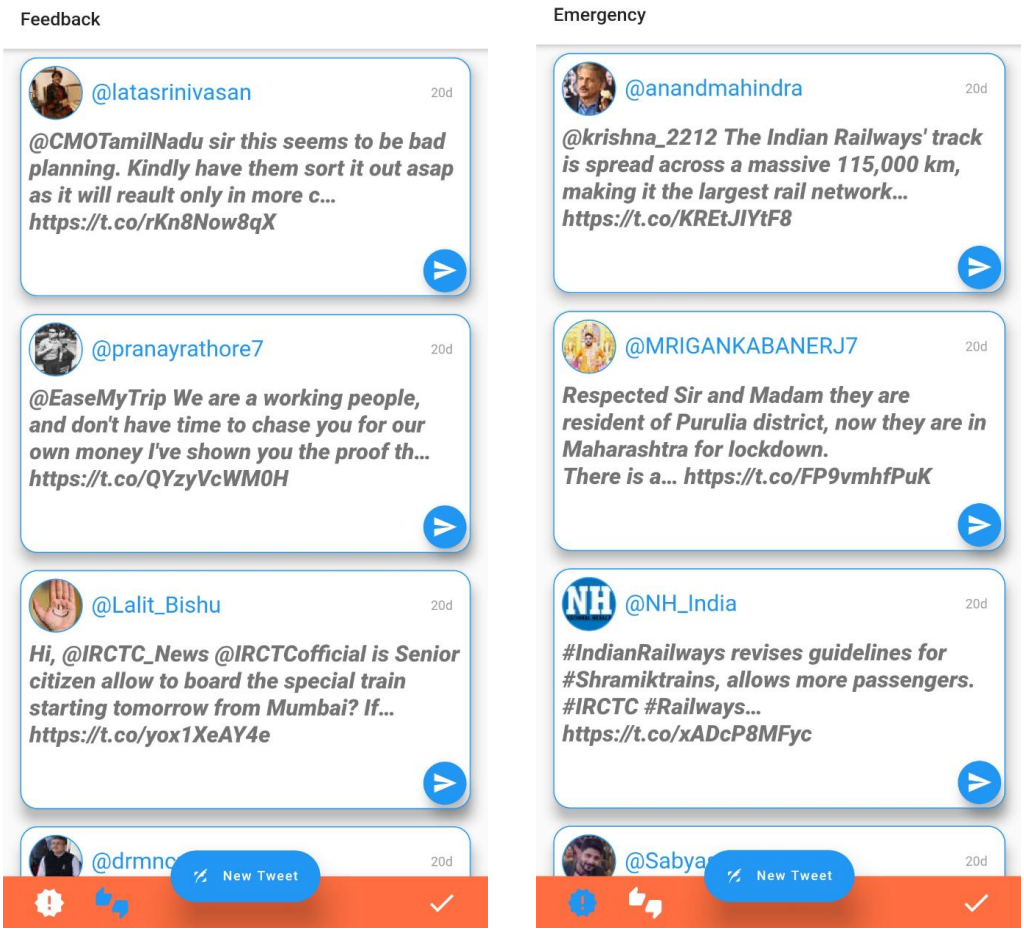

Figure 3.18: App snapshots

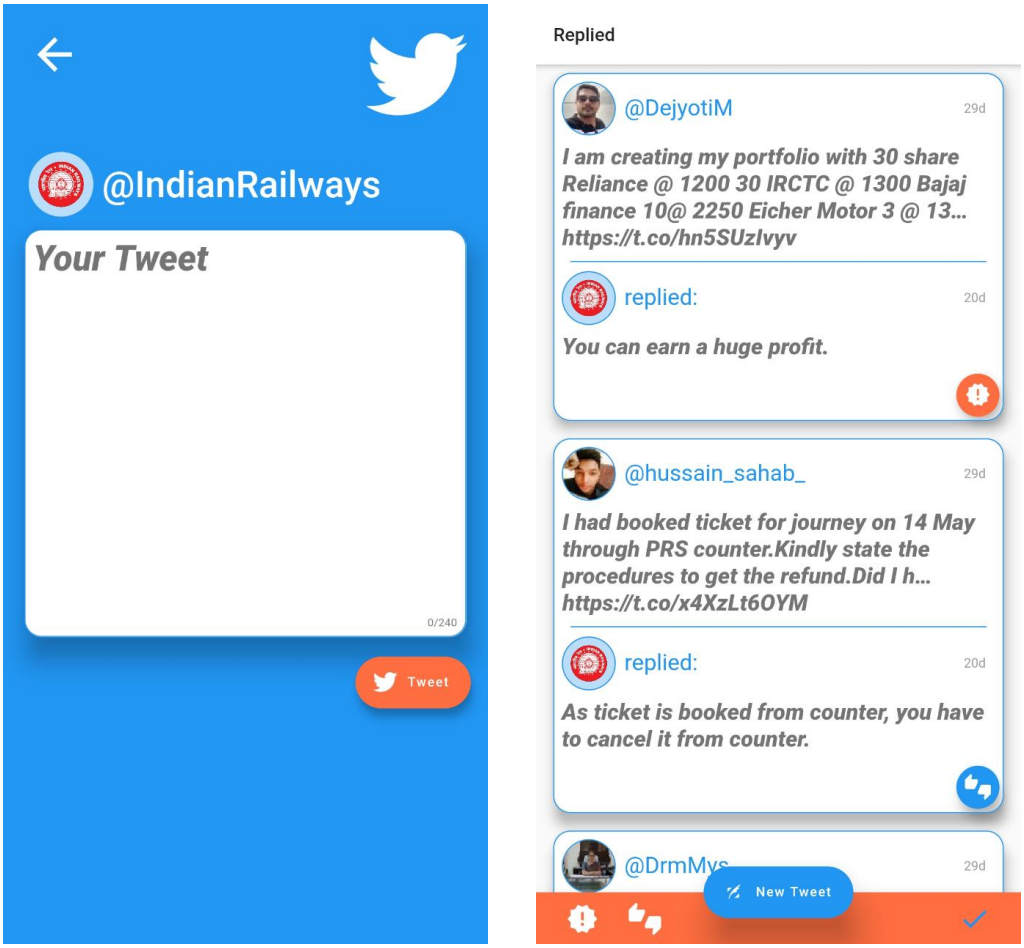

Figure 3.19: App snapshots

3.3.4 Web Module

There are Mainly three sub-modules present:

1. Emergency section
2. Feedback section
3. Replied section

3.3.5 Deployment on Cloud

- Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.
- Amazon EC2 presents a true virtual computing environment, allowing you to use web service interfaces to launch instances with a variety of operating systems, load them with your custom application environment, manage your network's access permissions, and run your image using as many or few systems as you desire.

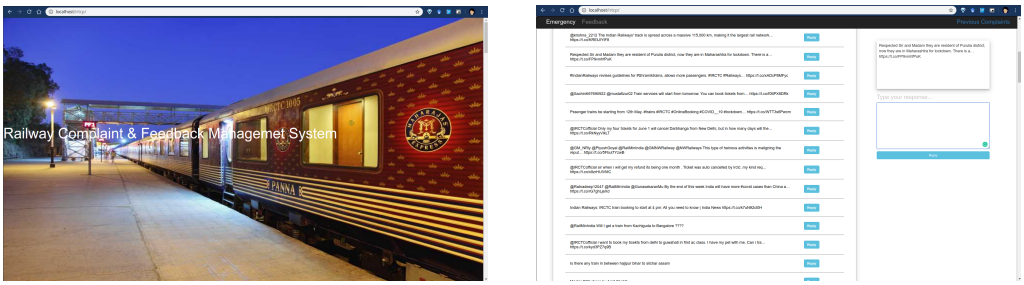

Figure 3.20: Website snapshots

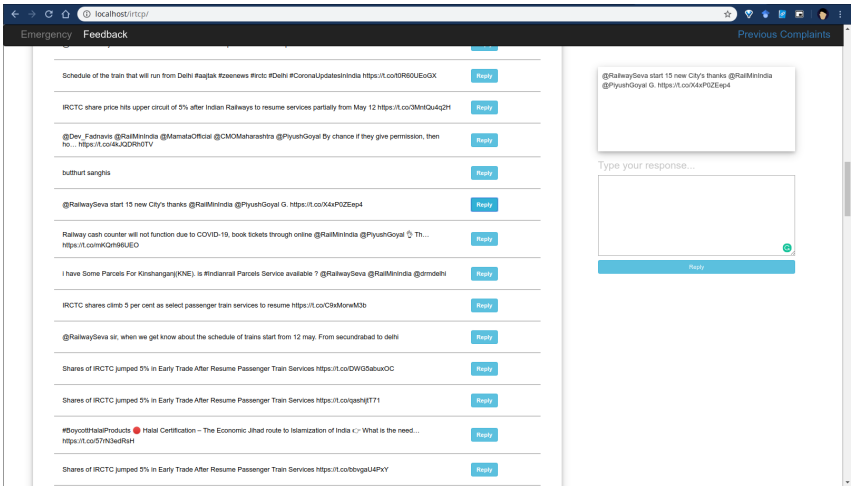

Figure 3.21: Website snapshots

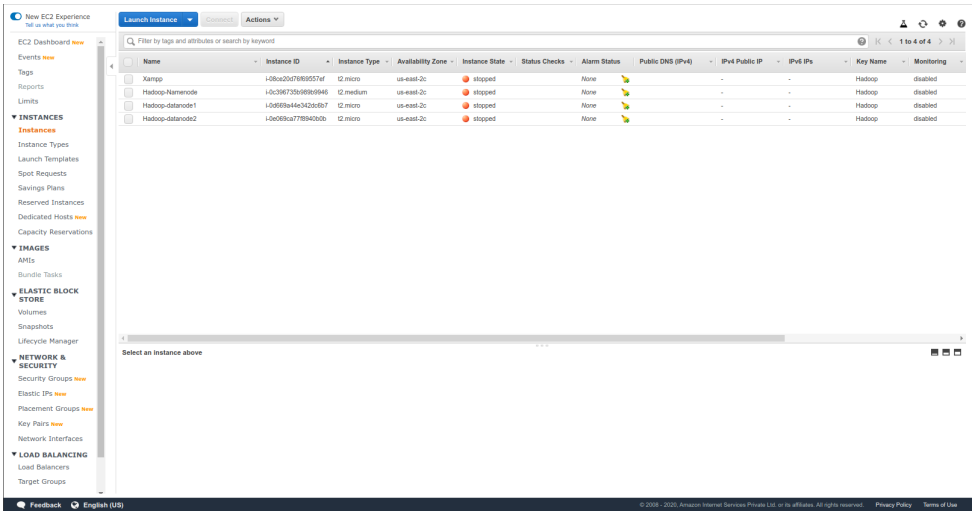

Figure 3.22: EC2 Instances

4. Coding and Testing

4.1 Coding

About the technology used:

4.1.1 Kafka

Apache Kafka is an open-source stream-processing software platform developed by LinkedIn and donated to the Apache Software Foundation, written in Scala and Java. The project aims to provide a unified, high-throughput, low-latency platform for handling real-time data feeds. Kafka can connect to external systems (for data import/export) via Kafka Connect and provides Kafka Streams, a Java stream processing library. Kafka uses a binary TCP-based protocol that is optimized for efficiency and relies on a "message set" abstraction that naturally groups messages together to reduce the overhead of the network roundtrip. This "leads to larger network packets, larger sequential disk operations, contiguous memory blocks [...] which allows Kafka to turn a bursty stream of random message writes into linear writes.

Machine Learning and the Apache Kafka Ecosystem are an excellent combination for training and deploying scalable analytical models. Here, Kafka becomes the central nervous system in the ML architecture, feeding analytical models with data, training them, using them for forecasting and monitoring them. This yields enormous advantages:

- Data pipelines are simplified.
- The implementation of analytical models is separated from their maintenance.
- Real time or batch can be used as needed.
- Analytical models can be applied in a high-performance, scalable and business-critical environment.

4.1.2 Spark

Spark is a general-purpose distributed data processing engine that is suitable for use in a wide range of circumstances. On top of the Spark core data processing engine, there are libraries for SQL, machine learning, graph computation, and stream processing, which can be used together in an application. Programming languages supported by Spark include: Java, Python, Scala, and R. Application developers and data scientists incorporate Spark into their applications to rapidly query, analyze, and transform data at scale. Tasks most frequently associated with Spark include ETL and SQL batch jobs across large data sets, processing of streaming data from sensors, IoT, or financial systems, and machine learning tasks. Apache Spark has following features:

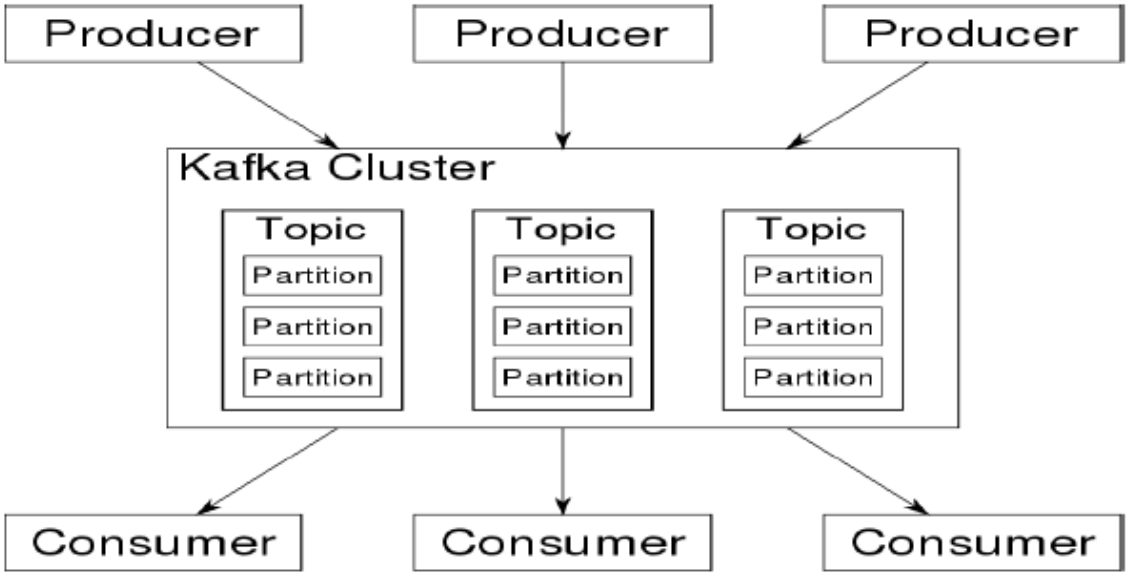

Figure 4.1: Kafka

- Speed Spark helps to run an application in Hadoop cluster, up to 100 times faster in memory, and 10 times faster when running on disk. This is possible by reducing number of read/write operations to disk. It stores the intermediate processing data in memory.
- Supports multiple languages Spark provides built-in APIs in Java, Scala, or Python. Therefore, you can write applications in different languages. Spark comes up with 80 high-level operators for interactive querying.
- Advanced Analytics Spark not only supports 'Map' and 'reduce'. It also supports SQL queries, Streaming data, Machine learning (ML), and Graph algorithms.

4.1.2.1 TF-IDF

Term frequency-inverse document frequency (TF-IDF) is a feature vectorization method widely used in text mining to reflect the importance of a term to a document in the corpus. Denote a term by tt, a document by dd, and the corpus by DD. Term frequency TF(t,d)TF(t,d) is the number of times that term tt appears in document dd, while document frequency DF(t,D)DF(t,D) is the number of documents that contains term tt. If we only use term frequency to measure the importance, it is very easy to over-emphasize terms that appear very often but carry little information about the document, e.g., "a", "the", and "of". If a term appears very often across the corpus, it means it doesn't carry special information about a particular document. Inverse document frequency is a numerical measure of how much information a term provides:

$$IDF(t, D) = log|D| + 1DF(t, D) + 1, IDF(t, D) = log|D| + 1DF(t, D) + 1,$$

where —D——D— is the total number of documents in the corpus. Since logarithm is used, if a term appears in all documents, its IDF value becomes 0. Note that a smoothing term is applied to avoid dividing by zero for terms outside the corpus. The TF-IDF measure is simply the product of TF and IDF:

$$TFIDF(t, d, D) = TF(t, d)IDF(t, D).TFIDF(t, d, D) = TF(t, d)IDF(t, D).$$

There are several variants on the definition of term frequency and document frequency. In spark.mllib, we separate TF and IDF to make them flexible.

Our implementation of term frequency utilizes the hashing trick. A raw feature is mapped into an index (term) by applying a hash function. Then term frequencies are calculated based on the mapped indices. This approach avoids the need to compute a global term-to-index map, which can be expensive for a large corpus, but it suffers from potential hash collisions, where different raw features may become the same term after hashing. To reduce the chance of collision, we can increase the target feature dimension, i.e., the number of buckets of the hash table.

4.1.2.2 Machine learning

In machine learning, naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. They are among the simplest Bayesian network models.[1] But they could be coupled with Kernel density estimation and achieve higher accuracy levels.

Naive Bayes is a simple multiclass classification algorithm with the assumption of independence between every pair of features. Naive Bayes can be trained very efficiently. Within a single pass to the training data, it computes the conditional probability distribution of each feature given label, and then it applies Bayes' theorem to compute the conditional probability distribution of label given an observation and use it for prediction.

spark.mllib supports multinomial naive Bayes and Bernoulli naive Bayes. These models are typically used for document classification. Within that context, each observation is a document and each feature represents a term whose value is the frequency of the term (in multinomial naive Bayes) or a zero or one indicating whether the term was found in the document (in Bernoulli naive Bayes). Feature values must be nonnegative. The model type is selected with an optional parameter "multinomial" or "bernoulli" with "multinomial" as the default. Additive smoothing can be used by setting the parameter (default to 1.01.0). For document classification, the input feature vectors are usually sparse, and sparse vectors should be supplied as input to take advantage of sparsity. Since the training data is only used once, it is not necessary to cache it.

4.1.3 Zookeeper

Apache ZooKeeper is an effort to develop and maintain an open-source server which enables highly reliable distributed coordination. ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services. All of these kinds of services are used in some form or another by distributed applications. Each time they are implemented there is a lot of work that goes into fixing the bugs and race conditions that are inevitable. Because of the difficulty of implementing these kinds of services, applications initially usually skimp on them, which make them brittle in the presence of change and difficult to manage. Even when done correctly, different implementations of these services lead to management complexity when the applications are deployed.

Apache ZooKeeper is a software project of Apache Software Foundation. It is an open-source technology that maintains configuration information and provides synchronized as well as group services which are deployed on Hadoop cluster to administer the infrastructure. The ZooKeeper framework was originally built at Yahoo! for easier accessing of applications but, later on, ZooKeeper was used for organizing services used by distributed frameworks like Hadoop, HBase,

etc., and Apache ZooKeeper became a standard. It was designed to be a vigorous service that enabled application developers to focus mainly on their application logic rather than coordination.

In a distributed environment, coordinating and managing a service has become a difficult process. Apache ZooKeeper was used to solve this problem because of its simple architecture, as well as API, that allows developers to implement common coordination tasks like electing a master server, managing group membership, and managing metadata.Apache ZooKeeper is used for maintaining centralized configuration information, naming, providing distributed synchronization, and providing group services in a simple interface so that we don't have to write it from scratch. Apache Kafka also uses ZooKeeper to manage configuration. ZooKeeper allows developers to focus on the core application logic, and it implements various protocols on the cluster so that the applications need not implement them on their own.

4.1.4 HTML

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you to create. It is constantly undergoing revision and evolution to meet the demands and requirements of the growing Internet audience.

HTML is Hypertext Markup Language.

The definition of HTML is Hypertext Markup Language:

- Hypertext is the method by which you move around on the web — by clicking on special text called hyperlinks which bring you to the next page. The fact that it is hyper just means it is not linear — i.e. you can go to any place on the Internet whenever you want by clicking on links — there is no set order to do things in.
- Markup is what HTML tags do to the text inside them. They mark it as a certain type of text.
- HTML is a Language, as it has code-words and syntax like any other language.

HTML consists of a series of short codes typed into a text-file. The text is then saved as a html file, and viewed through a browser, like Internet Explorer. The tags are what separate normal text from HTML code. You might know them as the words between the ¡angle-brackets¿. The tags themselves don't appear when you view your page through a browser, but their effects do. The simplest tags do nothing more than apply formatting to some text.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content.

4.1.5 CSS

HTML was originally designed as a simple way of presenting information, with the aesthetics of a web page being far less important than the content (and largely being left up to the web browser). Of course, now that the web has become as popular as it has, the presentation of your content has become almost critical to a site's success. CSS is the key presentational technology that is used to design websites. Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content. CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties. A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors, and a declaration block.

Well-authored CSS improves the accessibility of web content, allowing access through myriad devices (handheld PDAs for example) and ensuring that web users with disabilities are still able to receive it. It also eliminates the need for browser-specific hacks and tags, which means your site has a better chance of working across all major browsers. Another of CSS's boons is that you define things once, making it far more efficient than defining everything in HTML on every page. This means:Pages download faster, sometimes by as much as 50%.

You have to type less code, and your pages are shorter and neater.The look of your site is kept consistent throughout all the pages that work off the same style sheet.Updating your design and general site maintenance are made much easier, and errors caused by editing multiple HTML pages occur far less often.CSS files are termed "cascading" style sheets because of two reasons: one style sheet can cascade, or have influence over, multiple pages.

4.1.6 PHP

PHP is a server side scripting language. that is used to develop Static websites or Dynamic websites or Web applications.PHP scripts can only be interpreted on a server that has PHP installed. The client computers accessing the PHP scripts require a web browser only.A PHP file contains PHP tags and ends with the extension ".php".

- PHP is an acronym for "PHP: Hypertext Preprocessor".
- PHP is a widely-used, open source scripting language.
- PHP scripts are executed on the server.
- PHP is free to download and use.
- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.).
- PHP is compatible with almost all servers used today (Apache, IIS, etc.).
- PHP supports a wide range of databases.

- PHP can generate dynamic page content.
- PHP can create, open, read, write, delete, and close files on the server.
- PHP can collect form data.
- PHP can send and receive cookies.
- PHP can add, delete, modify data in your database.
- PHP can be used to control user-access.
- PHP can encrypt data.

4.1.7 Javascript

JavaScript often abbreviated as JS, is a programming language. Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it for client-side page behavior, and all major web browsers have a dedicated JavaScript engine to execute it.

It was originally developed by Netscape as a means to add dynamic and interactive elements to websites. While JavaScript is influenced by Java, the syntax is more similar to C. JavaScript is a clientside scripting language, which means the source code is processed by the client's web browser rather than on the web server. This means JavaScript functions can run after a webpage has loaded without communicating with the server. Like server-side scripting languages, such as PHP and ASP, JavaScript code can be inserted anywhere within the HTML of a webpage. However, only the output of serverside code is displayed in the HTML, while JavaScript code remains fully visible in the source of the webpage. It can also be referenced in a separate .JS file, which may also be viewed in a browser.

JavaScript functions can be called within ¡script¿ tags or when specific events take place. Examples include onClick, onMouseDown, onMouseUp, onKeyDown, onKeyUp, onFocus, onBlur, onSubmit, and many others. While standard JavaScript is still used for performing basic clientside functions, many web developers now prefer to use JavaScript libraries like jQuery to add more advanced dynamic elements to websites.

4.1.8 Flutter

Google has introduced us Flutter technology which is the new open source library to create mobile applications. This technology may be the solution to create a cross-platform mobile application with a beautiful user interface. The way of designing views are similar to the web application and you can find many analogies to CSS and HTML.

Basically, Flutter technology is the mobile app SDK to build high performance, high fidelity, apps for iOS and android both platforms from the single codebase. The major goal of this app is to enable the developers to deliver the high-performance apps which feel natural on different platforms. This difference exists in scrolling behaviors, typography, icons and many more. Unlike other major solutions, this technology is not the framework as it completes the SDK software development kit which already contains everything that the users need to build cross

platforms applications.This technology falls with many major benefits in the mobile app development process as:

Faster mobile app

The mobile apps developed by using the flutter app development are one of the easiest and faster ways in comparison to the applications built on other app development platform. These apps are quite a smooth functioning as they provide great users experience without cutting or hanging and continuously scrolling.

Great for MVP

MVP plays an important role as the foundation for mobile and web application development, therefore if you want your digital products to be developed at the quicker face, and then you must go for flutter app development platform. This will help them to reduce the app development cost at the same time speed up their development processes.

Hot Reload

Hot reload is considered as one of the greatest advantages that flutter technology provide, because of this the app developers are able to look at all the required changes which include minor ones in the code right away. With the help of hot read, the developers can also make quick fixes in the app's code for its smooth functioning.

Less testing efforts

In flutter app development, the user need not really have to create two separate apps for the different platforms like Android or iOS, because this technology provides the single codebase that can be used in more than one platform. This ultimately decreases the testing efforts as the quality assurance team does not need to run the same test on the different platform again and again.

Native features access

Setting up the flutter development is very easy and can be initiated on low-end machines where the app developers can access the native features like geolocation and camera. This cross-platform mobile app development platform allows you to reuse the existing code for both the platforms.

4.1.9 MongoDB

MongoDB is a document-oriented NoSQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, MongoDB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in MongoDB. Collections contain sets of documents and function which is the equivalent of relational database tables. MongoDB is a database which came into light around the mid-2000s.

MongoDB Features:

- Each database contains collections which in turn contains documents. Each document can be different with a varying number of fields. The size and content of each document can be different from each other.
- The document structure is more in line with how developers construct their classes and objects in their respective programming languages. Developers will often say that their classes are not rows and columns but have a clear structure with key-value pairs.
- The rows (or documents as called in MongoDB) doesn't need to have a schema defined beforehand. Instead, the fields can be created on the fly.
- The data model available within MongoDB allows you to represent hierarchical relationships, to store arrays, and other more complex structures more easily.

4.1.10 Cloud

The cloud is a virtual space that exists on the internet. It is a storage space where people can place their digital resources such as software, applications and files. So in simplified terms, we can say that the cloud is a virtual storage space on the internet. A lot of people do get the cloud mixed up with the internet. However, the cloud is only one part of the internet and not the whole thing.

So how does cloud technology work? Cloud computing technology allows people to use the digital resources stored in the virtual space by way of networks – often satellite networks. It allows people to share information and applications across the internet without being the restriction of their physical location. There are some major benefits that cloud computing.

BETTER STORAGE

Cloud storage is not limited by the capacity of any physical device. Unlike the previous storage solutions, there are no limitations of capacity. This means we get better and higher storage and don't have to worry about upgrading the memory of a device.

BETTER SCALABILITY

There are a lot of companies that have fluctuating bandwidth requirements. Cloud is a beneficial tool for them since it allows them to scale up when needed by drawing on the remote servers. Similarly, you can scale down easily when needed. This scalability is often referred to as the 'operational agility' by CIOs of growing brands and credited for the success.

BETTER COLLABORATION WITH REMOTE USERS

Today's economy requires workers to have the knowledge to work with colleagues from multiple remote locations. Cloud computing makes it easy to share and access digital resources from any part of the globe. This leads to better collaboration between global teams.

HIGHLY AFFORDABLE

Cloud computing eliminates the need for physical hardware for storage purposes. This, in turn, reduces the capital expenditure for companies. These funds can then be appropriated towards innovation or research and development to pave new pathways of success.
For small companies that are still trying to find their footing, purchasing a software can be

expensive. Instead, such companies can use the software on a pay-per-use basis from the cloud. It is more like leasing out a service instead of purchasing it.

DISASTER RECOVERY

It is imperative for businesses of all sizes to invest in disaster recovery protocols. However, it requires money and expertise, which is often lacking with small companies. The cloud gives them the ability to implement backup and recovery solutions in cost-effective and hassle-free manner.

4.2 Testing

The aim of testing process was to determine all defects in our project. The program was subjected to a set of inputs and various observations were made and based on these observations it will be decided whether the program behaves as expected or not. Our Project went through three levels of testing:-

1. Unit Testing
2. Integration Testing
3. Regression Testing

4.2.1 Unit Testing

Unit testing is undertaken when a module has been created and successfully reviewed. In order toTest a single module we need to provide a complete environment ie besides the module we would require

- The procedures belonging to other modules that the module under test calls.
- Non Local data structures that module accesses.
- A procedure to call the functions of the module under test with appropriate parameters.

Unit testing was done on each and every module that is described under module description.

1. Test for Login module for app- This form is used for log in of existing user. In this we enter the username and password if both are correct then, account will open otherwise if any data is wrong it will redirected back to login page and again ask for username and password.
2. Test for Login module for app- This form is used for log in of existing user. In this we enter the username and password if both are correct then, account will open otherwise if any data is wrong it will redirected back to login page and again ask for username and password.
3. Test for sending reply in real-time – The reply for tweet will be send by the admin in the real time or not will be checked.
4. Test for Checking the categorization of tweets into feedback and emergency.

4.2.2 Integration Testing

The second step is integration testing. In this type of testing we test various integration of the project module by providing the input. The primary objective is to test the module interfaces in order to ensure that no errors are occurring when one module invokes the another module. Integration Testing is testing in which modules are combined and tested as a group. Modules are typically code modules, individual applications, client and server applications on a network, etc. Integration Testing follows unit testing and precedes system testing. There are fundamentally 2 approaches for doing Integration testing:

- Bottom up testing, as the name suggests starts from the lowest or the innermost unit of the application, and gradually moves up. The Integration testing starts from the lowest module and gradually progresses towards the upper modules of the application. This integration continues till all the modules are integrated and the entire application is tested as a single unit.
- This technique starts from the top most module and gradually progress towards the lower modules. Only the top module is unit tested in isolation. After this, the lower modules are integrated one by one. The process is repeated until all the modules are integrated and tested.

4.2.3 Regression Testing

Any time you modify an implementation within a program, you should also do regression testing. You can do so by rerunning existing tests against the modified code to determine whether the changes break anything that worked prior to the change and by writing new tests where necessary. Adequate coverage without wasting time should be a primary consideration when conducting regression tests. Try to spend as little time as possible doing regression testing without reducing the probability that you will detect new failures in old, already tested code. Some strategies and factors to consider during this process include the following:

- Test fixed bugs promptly. The programmer might have handled the symptoms but not have gotten to the underlying cause.
- Watch for side effects of fixes. The bug itself might be fixed but the fix might create other bugs.
- Write a regression test for each bug fixed.
- If two or more tests are similar, determine which is less effective and get rid of it.
- Identify tests that the program consistently passes and archive them.
- Focus on functional issues, not those related to design.
- Make changes (small and large) to data and find any resulting corruption.
- Trace the effects of the changes on program memory.
- The most effective approach to regression testing is based on developing a library of tests made up of a standard battery of test cases that can be run every time you build a new version of the program. The most difficult aspect involved in building a library of test cases is determining which test cases to include. The most common suggestion from authorities in the field of software testing is to avoid spending excessive amounts of time trying to

decide and err on the side of caution. Automated tests, as well as test cases involving boundary conditions and timing almost definitely belong in your library. Some software development companies include only tests that have actually found bugs. The problem with that rationale is that the particular bug may have been found and fixed in the distant past.

- Periodically review the regression test library to eliminate redundant or unnecessary tests. Do this about every third testing cycle. Duplication is quite common when more than one person is writing test code. An example that causes this problem is the concentration of tests that often develop when a bug or variants of it are particularly persistent and are present across many cycles of testing. Numerous tests might be written and added to the regression test library. These multiple tests are useful for fixing the bug, but when all traces of the bug and its variants are eliminated from the program, select the best of the tests associated with the bug and remove the rest from the library.

5. Implementation

5.1 Implementation Activities

5.1.1 AWS EC2 startup

We will now create 4 instances of Ubuntu Server 16.04 LTS using Amazon EC2.

Select Instance

Go to your AWS Console, Click on Launch Instance and select Ubuntu Server 16.04 LTS.

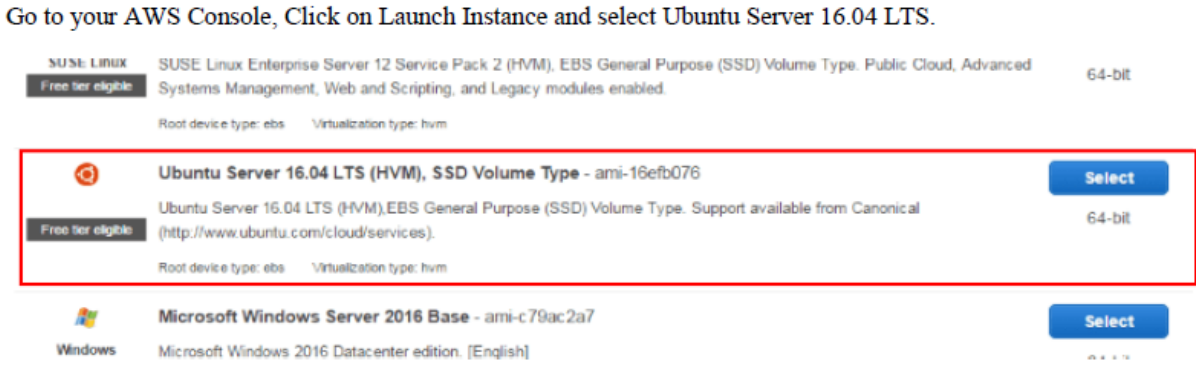

Figure 5.1: AWS EC2 Instance snapshots

Instance Type

For the instance type, we choose t2.micro since that is sufficient for the purposes of the demo.
If you have a need for a high-memory or high-cpu instance, you can select one of those.

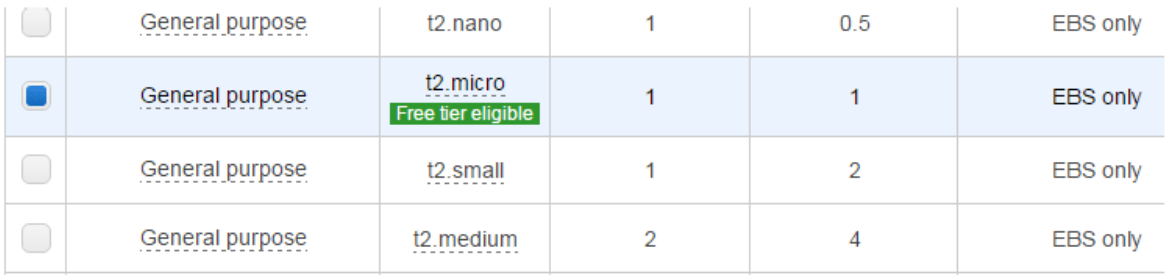

Figure 5.2: AWS EC2 Instance snapshots

Click Next to Configure Instance Details

Instance Details

Here, we request 4 instances of the selected machine type. We also choose a subnet (us-west-1b) just so we can launch into the same location if we need more machines. Click Next to Add

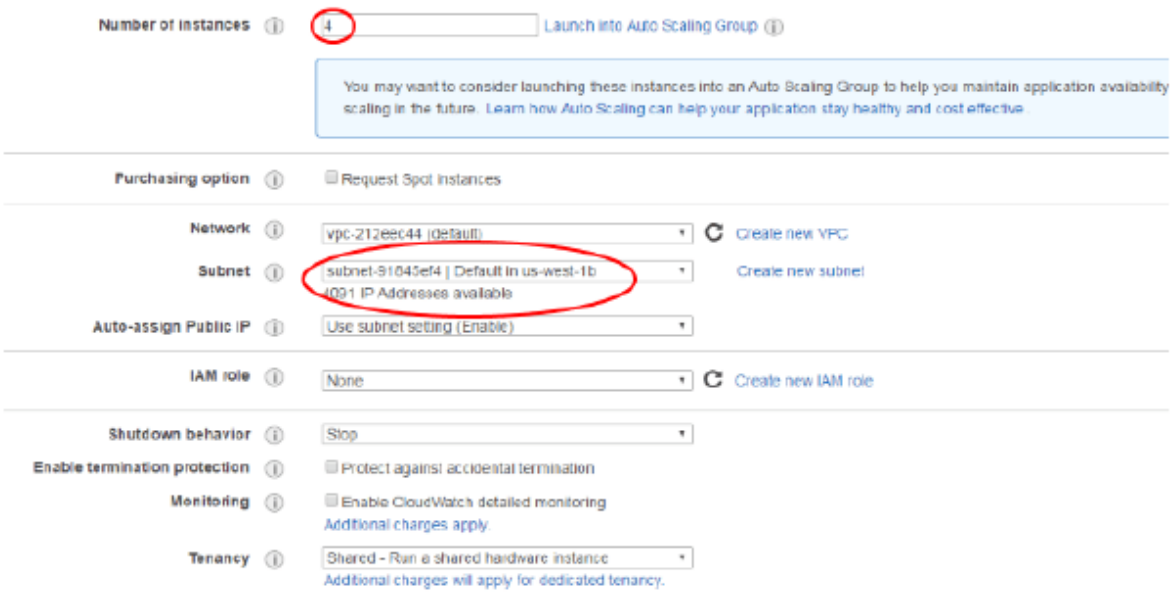

Figure 5.3: AWS EC2 Instance snapshots

Storage.

Storage

For our purpose, the default instance storage of 8GB is sufficient. If you need more storage, either increase the size or attach a disk by clicking "Add Volume". If you add a volume, you will need to attach the volume to your instance, format it and mount it. Since this is a beginner tutorial, these steps are not covered here.

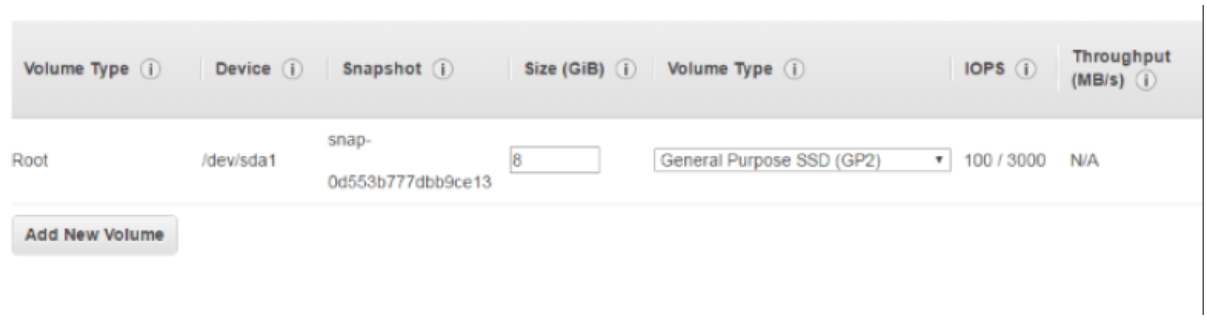

Figure 5.4: AWS EC2 Instance snapshots

Click Next to Add Tags to your instances.

Instance Tags

A tag allows you to identify your instance with a name you can choose.Click Add Tag, set the Key to “Name” and value to “Hadoop”. We will use this tag to relabel our instances as “namenode”, “datanode1” and so on later on. For now leave the value of all the instances as “Hadoop”.

Figure 5.5: AWS EC2 Instance snapshots

Click Next to configure Security Group for the instances.

Security Group

For the security group, we create a completely open security group for the purposes of testing.

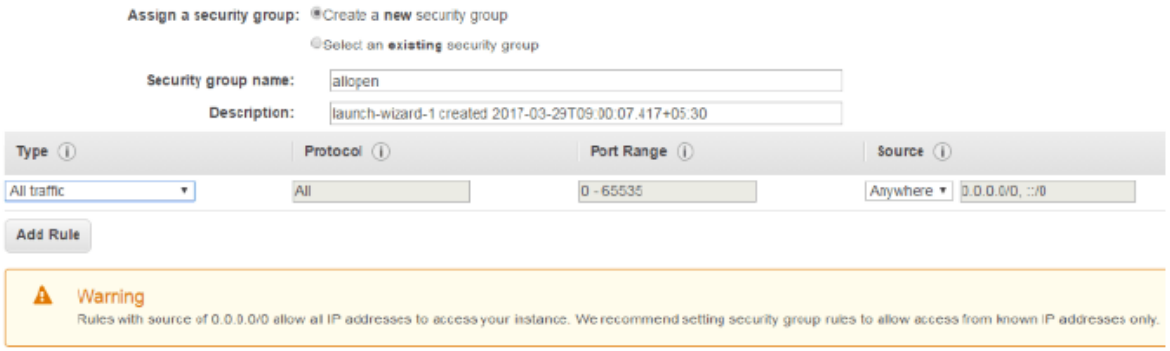

Figure 5.6: AWS EC2 Instance snapshots

Finally we get to the Launch screen.

Launch Instances

Review the information again and click Launch to start the instances. You will need to create a key pair or use an existing key pair. Follow the instructions on the Launch Wizard to create a new key pair. Then click “I acknowledge ..” and then Launch Instances.

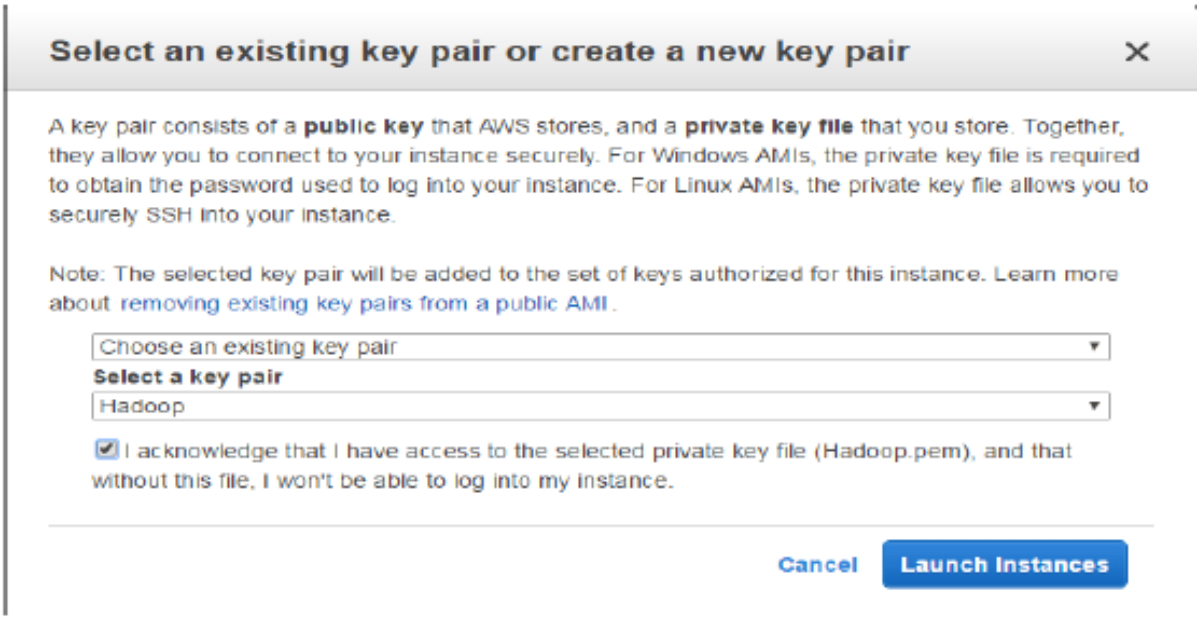

Figure 5.7: AWS EC2 Instance snapshots

You can now go to the instances page and check on the status of the instances.

Naming the Instances

On the instances page, let us setup the names of the instances. These are not DNS names, but names we assign to help us distinguish between them.
Click the pencil icon next to the name and setup the names as shown.

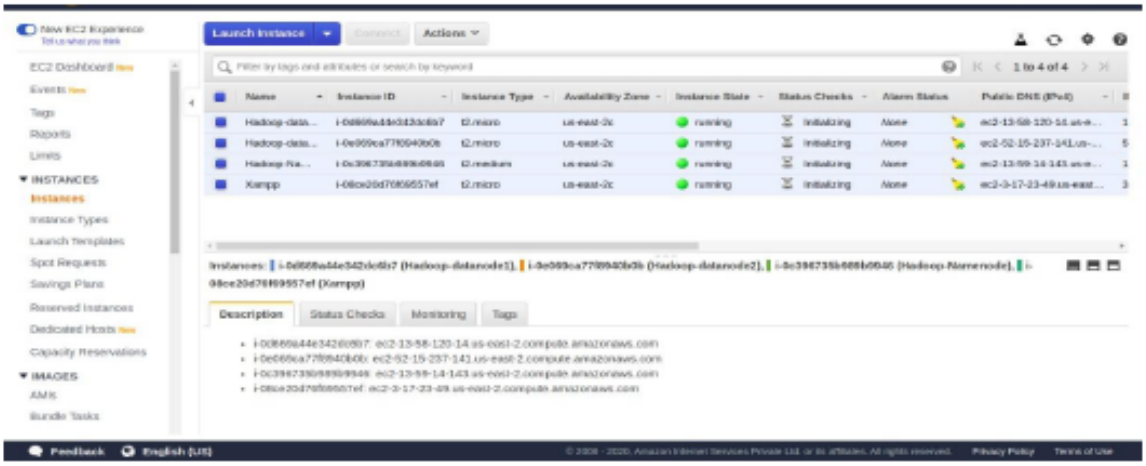

Figure 5.8: AWS EC2 Instance snapshots

Linux MAC connection

Executing following command with Linux or MAC terminal, you can directly access your box.

$$ssh - ikeyname.pem\ ubuntu@IP\_ADDRESS$$

Give only READ permission for .pem file, then execute SSH command again.

$$sudo\ chmod\ 400\ keyname.pem$$

Setting Up Instances

Once the instances are up and running, it is time to set them up for our purpose. This includes the following:
Setup password-less login between the namenode,the datanodes and Xampp.

Install java.

Copy Instance Public DNS Name

We now need to copy the Public DNS Name of each node (1 namenode and 3 datanodes). These names are used in the configuration steps below. Since the DNS is specific to each setup, we refer to the names as follows.

For example, in the description below, if you see ¡nnode¿, substitute with the value of ¡NameNode Public DNS¿. Similarly for ¡dnode1¿ and so on.

Nnode	<NameNode Public DNS>
dnode1	<DataNode1 Public DNS>
dnode2	<DataNode2 Public DNS>
dnode3	<xampp Public DNS>

Figure 5.9: AWS EC2 Instance snapshots

Common Setup on All Nodes

Some setup is common to all the nodes: NameNode and DataNodes. This is covered in this section.

All Nodes: Update the instance

Let us update the OS with latest available software patches.

sudo apt-get update sudo apt-get -y dist-upgrade

After the updates, the system might require a restart. Perform a Reboot from the EC2 Instances page.

All Nodes: Install Java

Let us now install Java. We install the package: openjdk-8-jdk-headless on all the nodes.

sudo apt-get -y install openjdk-8-jdk-headless

All Nodes: Setup JAVA_HOME

On each of the nodes, edit /server/hadoop-2.7.3/etc/hadoop/hadoop-env.sh.

Replace this line:

export JAVA_HOME=$JAVA_HOME

With the following line:

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

Configuring NameNode

After performing configuration common to all nodes, let us now setup the NameNode.

Namenode: Password Less SSH

As mentioned before, we need password-less SSH between the name nodes and the data nodes. Let us create a public-private key pair for this purpose on the namenode.

namenode ssh-keygen

Use the default (/home/ubuntu/.ssh/id_rsa) for the key location and hit enter for an empty passphrase.

Datanodes: Setup Public Key

The public key is saved in /home/ubuntu/.ssh/id_rsa.pub. We need to copy this file from the namenode to each data node and append the contents to /home/ubuntu/.ssh/authorized_keys on each data node.

datanode1> cat id_rsa.pub >> /.ssh/authorized_keys

datanode2> cat id_rsa.pub >> /.ssh/authorized_keys

xampp> cat id_rsa.pub >> /.ssh/authorized_keys

Namenode: Setup SSH Config

SSH uses a configuration file located at /.ssh/config for various parameters. Set it up as shown below. Again, substitute each node's Public DNS for the HostName parameter (for example, replace <nnode> with EC2 Public DNS for NameNode).

```
Host nnode
  HostName <nnode>
  User ubuntu
  IdentityFile ~/.ssh/id_rsa
```

```
Host dnode1
  HostName <dnode1>
  User ubuntu
  IdentityFile ~/.ssh/id_rsa
```

```
Host dnode2
  HostName <dnode2>
  User ubuntu
  IdentityFile ~/.ssh/id_rsa
```

```
Host dnode3
  HostName <xampp>
  User ubuntu
  IdentityFile ~/.ssh/id_rsa
```

At this point, verify that password-less operation works on each node as follows (the first

time, you will get a warning that the host is unknown and whether you want to connect to it. Type yes and hit enter. This step is needed once only):

```
namenode> ssh nnode

namenode> ssh dnode1

namenode> ssh dnode2

namenode> ssh xampp
```

5.1.2 XAMPP Installation

Download XAMPP for 64 bit get https://www.apachefriends.org/xampp-files/7.0.23/xampp-linux-x64-7.0.23-0- installer.run

Make Execute Installation
sudo chmod +x xampp-linux-x64-7.0.23-0-installer.run

Run Installation
sudo ./xampp-linux-x64-7.0.23-0-installer.run

XAMPP instructions
Select the components you want to install; clear the components you do not want to install.
Click Next when you are ready to continue.
XAMPP Core Files : Y (Cannot be edited)
XAMPP Developer Files [Y/n] : Y
Is the selection above correct? [Y/n]: Y

Installation Directory
XAMPP will be installed to /opt/lampp
Press [Enter] to continue:
Do you want to continue? [Y/n]:Y

Run XAMPP
sudo /opt/lampp/lampp start

XAMPP Access Forbidden
Open your browser and access http://IP-ADDRESS/ you will find this Access forbidden screen.

XAMPP Configurations
Edit XAMPP configurations.

/opt/lampp/etc/extra/httpd-xampp.conf

<LocationMatch "^/(?i:(?:xampp—security—licenses—phpmyadmin—webalizer—server-status—server-info))">
Require local
ErrorDocument 403 /error/XAMPP_FORBIDDEN.html.var
¡/LocationMatch¿
¡LocationMatch "^/(?i:(?:xampp—security—licenses—phpmyadmin—webalizer—server-status—server-info))"¿
Order deny,allow
Allow from all
Allow from ::1 127.0.0.0/8
fc00::/7 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
fe80::/10 169.254.0.0/16
ErrorDocument 403 /error/XAMPP_FORBIDDEN.html.var
¡/LocationMatch¿

Restart XAMPP
sudo /opt/lampp/lampp restart

Security Settings
sudo /opt/lampp/xampp security

```
XAMPP: Your XAMPP pages are NOT secured by a password.
XAMPP: Do you want to set a password? [yes]
XAMPP: Your XAMPP pages are NOT secured by a password.
XAMPP: Do you want to set a password? [yes] no
XAMPP: MySQL is accessable via network.
XAMPP: Normally that's not recommended.
       Do you want me to turn it off? [yes] yes
XAMPP: Turned off.
XAMPP: Stopping MySQL...ok.
XAMPP: Starting MySQL...ok.
XAMPP: The MySQL/phpMyAdmin user pma has no password set!!!
XAMPP: Do you want to set a password? [yes] yes
XAMPP: Password:*******
XAMPP: Password (again):*******
XAMPP: Setting new MySQL pma password.
XAMPP: Setting phpMyAdmin's pma password to the new one.
XAMPP: MySQL has no root passwort set!!!
XAMPP: Do you want to set a password? [yes] yes
XAMPP: Write the password somewhere down to
       make sure you won't forget it!!!
XAMPP: Password:*******
XAMPP: Password (again):*******
XAMPP: Setting new MySQL root password.
```

```
XAMPP: Change phpMyAdmin's authentication method.
XAMPP: The FTP password for user 'daemon' is still set to 'xampp'.
XAMPP: Do you want to change the password? [yes] no
XAMPP: Done.
```

5.1.3 Zookeeper Installation

Installation

cd /home/ec2-user
wget http://www-eu.apache.org/dist/zookeeper/zookeeper3.4.9/zookeeper-3.4.9.tar.gz
tar -xvf zookeeper-3.4.9.tar.gz
mv zookeeper-3.4.9 zookeeper
rm -rf zookeeper-3.4.9.tar.gz

Configuration

cd zookeeper
cp conf/zoo_sample.cfg conf/zoo.cfg
cd /opt
sudo mkdir zookeeper
cd zookeeper/
sudo vi myid –¿ set a different number for each server
cd /zookeeper
vi conf/zoo.cfg
Add the below parameters in the file
dataDir=/opt/zookeeper
maxClientCnxns=100
server.1=0.0.0.0:2888:3888 (For the current server)
server.2=¡Private IP of server 2¿:2888:3888
server.3=¡Private IP of server 3¿:2888:3888
sudo bin/zkServer.sh start
sudo bin/zkServer.sh status

You should do these steps in at least 3 servers that is master, slave 1, slave 2. This is for the master server which is server.1.Similarily in server 2,3 the particular server will be current server and the rest two will have the private ip of server.For one server you will get the status as Mode: Leader and for the rest you will get Mode: follower.

5.1.4 Kafka installation

Log in to each EC2 Instance and update the packages.
$ sudo apt-get update
Then repeat the following steps on each of the instances.
1: Download the Kafka latest build.
$ wget http://mirrors.estointernet.in/apache/kafka/2.1.0/kafka_2.11-2.1.0.tgz
2: Extract the Kafka tar file.
$ tar xzf kafka_2.11-2.1.0.tgz

Now, we need to configure each server. First, run the zookeeper in each server. Make sure you have
read this article and configured the zookeeper server.
Assuming you have started the Zookeeper in each instance. Let's configure Kafka.
Switch to the Kafka folder and open the config file.
$ vi config/server.properties
On the first server, add this configuration.

The id of the broker. This must be set to a unique integer for each broker.
For 1st server broker.id=10
For 2nd server broker.id=20
For 3rd server broker.id=30
The following steps will be same in all the 3 servers.We have to repeat these steps each time in different server with unique broker id.

# A comma seperated list of directories under which to store log files
log.dirs=/tmp/kafka-logs
advertised.host.name=¡ip address of host¿
log.dirs=/tmp/kafka-logs

# add all 3 zookeeper instances ip here
zookeeper.connect=¡i¿:2181,¡ip¿:2181,¡ip¿:2181
zookeeper.connection.timeout.ms=6000
To check the proper installation of kafka run this command $ bin/kafka-topics.sh.If the command is executed without any error then the setup is done.

5.1.5 Spark Installation

Step 0: Pre-requisite setup
We also need to install Scala in all of our machines.

```
$ sudo apt install scala
....
....
Setting up scala-parser-combinators (1.0.3-3) ...
Setting up libhawtjni-runtime-java (1.15-2) ...
Setting up scala-library (2.11.12-4~18.04) ...
Setting up scala-xml (1.0.3-3) ...
Setting up libjansi-native-java (1.7-1) ...
Setting up libjansi-java (1.16-1) ...
Setting up libjline2-java (2.14.6-1) ...
Setting up scala (2.11.12-4~18.04) ...
update-alternatives: using /usr/share/scala-2.11/bin/scala to provide
/usr/bin/scala (scala) in auto
mode$ scala -version
Scala code runner version 2.11.12 -- Copyright 2002-2017, LAMP/EPFL
```

Step 1: Installing Spark

On each machine (both master and worker) install Spark using the following commands. You can configure your version by visiting here.

$ wget http://apache.claz.org/spark/spark-2.4.3/spark-2.4.3-bin-hadoop2.7.tgz
Extract the files and move them to /usr/local/spark and add the spark/bin into PATH variable.
$ tar xvf spark-2.4.3-bin-hadoop2.7.tgz
$ sudo mv spark-2.4.3-bin-hadoop2.7/ /usr/local/spark
$ /.bash_profileexport PATH=/usr/local/spark/bin:$PATH$ source /.bash_profile

Step 2: Configuring Master to keep track of its workers

Now that we have Spark installed in all our machines we need to let the Master instance know about the different workers that is available. We will be using a standalone cluster manager for demonstration purposes. You can also use something like YARN or Mesos to handle the cluster. Spark has detailed notes on the different cluster managers that you can use.
We need to modify /usr/local/spark/conf/spark-env.sh file by providing information about Java location and the master node's IP. This will be added to the environment when running Spark jobs.
contents of conf/spark-env.sh
export SPARK_MASTER_HOST=¡master-private-ip¿
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64/
We will also add all the IPs where the worker will be started. Open the /usr/local/spark/conf/slaves file and paste the following.
contents of conf/slaves
<worker-private-ip1>
<worker-private-ip2>

To check the proper installation of spark run this command sh /usr/local/spark/sbin/start-all.sh .If the command is executed without any error then the setup is done.

5.1.6 Other libraries installation

Python installation

sudo apt-get update
sudo apt-get install python3.6
Pip installation
get-pip.py

pip install numpy
pip install tweepy
pip install kafka-python
pip install requests
pip install regex

5.1.7 Create Twitter Api

Go to https://dev.twitter.com/apps/new and log in, if necessary Enter your Application Name, Description and your website address. You can leave the callback URL empty.
Accept the TOS, and solve the CAPTCHA.
Submit the form by clicking the Create your Twitter Application.

Copy the consumer key (API key) and consumer secret from the screen into your application.

After creating your Twitter Application, you have to give the access to your Twitter Account to use this Application.
To do this, click the Create my Access Token.
In order to access the Twitter, that is to get recent tweets and twitter followers count, you need, the four keys such as Consumer Key, Consumer Secret, Acess token, Access Token Secret.

5.1.8 Flutter installation

```
1. Install JDK 6 or later

First, we will install Oracle JDK 8. Run the following
commands on the terminal in Ubuntu.

sudo add-apt-repository ppa:webupd8team/java
sudo apt-get update
sudo apt-get install oracle-java8-installer
sudo apt-get install oracle-java8-set-default

To make sure, its installed successfully,
open a terminal and type
java -version

2. Download and install Android Studio
SYSTEM REQUIREMENTS

Tested on Ubuntu 16.04 LTS 64-bit distribution capable of running 32-bit appli
3 GB RAM minimum, 8 GB RAM recommended;
plus 1 GB for the Android Emulator
2 GB of available disk space minimum,
4 GB Recommended (500 MB for IDE + 1.5 GB for
Android SDK and emulator system image)
1280 x 800 minimum screen resolution

Download the Android Studio package for
Linux and extract it somewhere (e.g home
directory).

To launch Android Studio, open a terminal,
navigate to the android-studio/bin/ directory, and
execute studio.sh.
cd android-studio/bin
./studio.sh

Select whether you want to import previous
Android Studio settings or not, then click OK
```

3. Download and install Flutter

You can download the Flutter SDK from flutter.io for Linux version.

Youve to have a 64 bit os to install it ,
so at first confirm that you have the 64 bit version of Linux.

After downloading the Flutter SDK,
you should extract the file in the desired location ,
for example:
cd ~/Documents
$ tar xf ~/Downloads/flutter_linux_v1.2.1-stable.tar.xz

By running the above commands, you will extract
the Flutter SDK at Documents folder.

Now you need to update the PATH to add the Flutter SDK.

Open a terminal and go to the home directory and
then to current user directory by typing the
following command.
cd /home/{current_user}

Type the following commands to update the path.
gedit .bashrc

A file opens for editing. Be careful while
dealing with this file as this is the file
which contains the
system configurations.

At the end of this file add the path of the
Flutter SDK.

export PATH={path-of-sdk}/flutter/bin:$PATH
Save and close the file and then also close
the terminal.

Then run echo$PATH in another terminal to see
the updated path.

This will show you the updated path which
contains the Flutter SDK.

Now run flutter doctor in the terminal. If no
errors were there then flutter doctor will run
successfully.

4. Download and install Flutter plugin in

```
Android Studio.

Now you just need to install Flutter plugin in
Android Studio. You can do this by going into File-
>Settings->Plugins and then search for Flutter
in Search in repositories and from there you will
install Flutter and Dart plugin for Android Studio.
```

5.2 Documentation(User's Manual)

5.2.1 How to execute the System

To run project once everything is installed:

```
Run command on ec2master

login into database remote database

> mysql -h database-1.ctrbpvenwkxn.us-east-2.rds.
amazonaws.com -P 3306 -u ghost -p

create a database:
>create twitter;
>use twitter;
create tweet table :

>CREATE TABLE tweets (id int AUTO_INCREMENT PRIMARY KEY,
tweet
varchar(280), username varchar(50), pnr bigint(10),
prediction int(1), tweet_id
bigint(10),latitude decimal(10,8)
,longitude decimal(11,8) ,time TIMESTAMP ,
response_status int(1), response varchar(280));

Then create admin database table

Start zookeeper:

>zookeeper-server-start.sh
on ec2slave1:

>zookeeper-server-start.sh
on ec2slave2:

>zookeeper-server-start.sh
on ec2master

start kafka:
nohup is used for teminal to not hang up!
important when aceeesing remote system using local
```

```
terminal.

cd kafka folder (cd means change directory to kafka folder)
>nohup bin/kafka-server-start.sh config/server.properties
on ec2slave1 and ec2slave2:

cd kakfa folder
bin/kafka-server-start.sh config/server.properties &
on ec2master:

create topic only once
cd kafka folder:

bin/kafka-topics.sh --create --zookeeper --partitions 1 --topic twitterstream lo
replication-factor 1

on ec2master ,ec2slave1 ,ec2slave2 in same order:

start kafka consumer
cd to where kafka is installed

>bin/kafka-console-consumer.sh --bootstrap-server localhost:
2181 --topic twitterstream --
from-beginning &
on ec2master:

change directory into IR..... folder
> python kafka_file/stream_data.py &
job submit on spark for training

>spark-submit train_model.py
job submit on spark for prediction

>spark-submit --packages org.apache.spark:spark-
streaming-kafka-0-8_2.11:2.1.0

live_processing3.py
on xampp:

> sudo /opt/lampp/lampp start

Finally open railways/index.php file to interact with UI and manage tweets in re
```

5.2.2 How to enter the Data

The user should have a twitter account. The data is collected by the Tweets done by the people. The data that will be collected will be the PNR number of the train (if tweeted) and the complaint or feedback done by the user.

5.2.3 How to process the data (processing details)

The tweets which contain @RailMinIndia will be processed .The Tweets done by the people will be classified into two categories that is feedback and emergency by the machine learning model.The data and machine pipeline wil do the streamimg of data and add the labelled data into the database.

5.2.4 How to take out the reports

We will do the tweet analysis of both feedback and emergeny tweets.

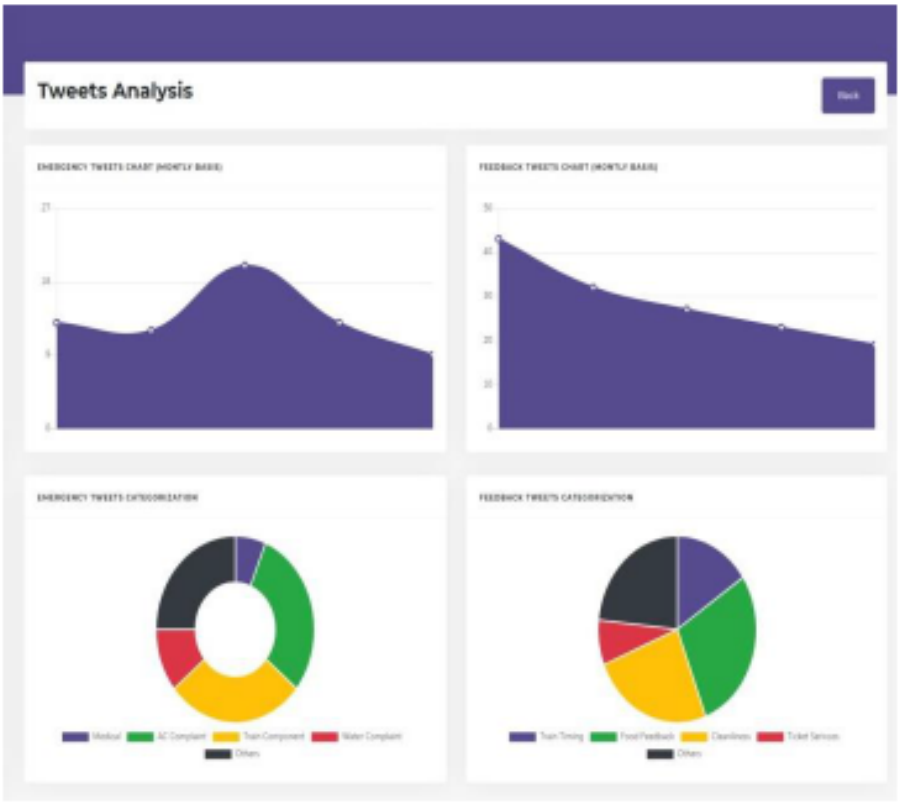

Figure. 3.3.4.4:Graph for Tweet analysis

6. Maintenance Features

6.1 Project Maintenance Plan

A guideline for planning ahead for the work that happens after releasing a product or system to its customers, and an annotated outline for writing a maintenance Plan. The guideline information includes an introduction to what Maintenance is, types of maintenance, and how maintenance planning fits within the development of the system or product to be maintained. Although the development project may be done and released, the deliverable from the project simply enters a new phase of its lifecycle. Whether hardware, software or system, the major project deliverables require ongoing support from the company. That support will include activities such as responding to user issues, maintaining the system physically, incorporating enhancements, and performing preventative maintenance. For the success of the product or system in the field, planning ahead thoroughly for proper maintenance support will be crucial. We can maintain the app and website module –

- Getting the Latest Versions of the Operating Systems.
- Can Provide Financial Benefits in the Long Run.
- The Improvement in Software Libraries.
- Offers an Updated User Interface (UI).
- Getting a Better User Experience (UX).
- Incorporating Technical Enhancements.
- App maintenance also allows your developer team to improve the app's security time to time against internet security threats.
- Frequent app monitoring will also allow you to keep your app in-check, this is very helpful when you don't have the resources to go under-maintenance. In such cases, monitoring the app will allow you to check the app's performance and avoid such expenses.
- Check that all of your pages are loading without errors.
- Run a backup and make sure a previous version of your site is stored.
- Make updates to website software and plugins.
- Check that all of your forms are running properly.
- Remove any spam comments from pages and posts.
- Check your pages to see if there are any broken links.
- Search for 404 errors and fix or redirect.

6.2 Dependencies of maintenance features

The best deal with this software is it has very less maintenance requirements. All you need to do is keep using the software in order to improve its quality. As of now,some of the maintenance features are:-

- Need of providing the training dataset for the machine to learn before using it in real world scenarios. This training dataset will further help in improving the accuracy of the software. Larger the dataset, higher will be the accuracy.
- Another check required for the smooth functioning of the software is to keep it upto- date as per the User Interface of the websites whosoever are used for emotion analysis. This is necessary for the successful parsing of the extracted data.
- Need to update the code with the newer version of the libraries .

7. Advantages and Limitations of the developed system

7.1 Advantages

- The project reduces the work complexity of scanning through thousands of useless data to find particular information but from here we can directly find the relevant tweets that needs attention.
- Reduced operational time .
- Increase fault-tolerance.
- Increased accuracy and reliability.
- This software package can be readily used by non-programming personal avoiding human handled chance of error.
- Automation of Feedback and complaint addressing.
- No Need to do Paper Work.
- Management of passenger data .
- The whole project is deployed on the cloud therefore
    1. Better Storage
    2. Disaster recovery
    3. Highly affordable
    4. Better collaboration with remote users
- Manual work reduced
- Reply is given in the real-timeSimplified Management of complaint addressing.
- Our system is capable of running on the least configuration computer systems in just a fewseconds unlike other deep learning applications that require a whole lot of computationpower as well as time
- Uses software which is open source.
- Easy to understand
- This Application has low cost of maintenance.

7.2 Limitations

- The most significant limitation is that complete project depends on tweets done by the users, thus if tweets are not available the following system cannot be implemented.
- Internet connection is required.

8. Conclusion and Suggestions for Further Work

8.1 Conclusion

With the advent of Artificial Intelligence, it is not so far that in near future the AI Assistants will be doing most of human tasks with much more efficiency and accuracy. With the help of machine learning model we have classify the tweets in two major catogeries – Feedback and Emergency.

We have made a app and website also.The user can directly complain any issue related to railway by log and they will receive the reply in real time. This project will reduce the manual work and the actions against the complain can be taken fast.

We believe this project if properly utilized will save time, reduce the amount of work the administration has to do. It will replace the stationery material with electronic apparatus. The system should also serve as a major tool in improving the efficiency of action taken against the complain . Hence a system with expected results has been developed but there is still room for improvement.

In terms of experience gained through the duration of this project study, we have been able to acquire broader knowledge about the management of complains in railway and how the actions are taken.We believe this project will serve the customers efficiently.

8.2 Suggestions for further work

The machine learning model will be flexible and can be applied on various other local authorities like Nagar Nigam, State electricity board and various other authorities. We just have to filter the tweets on the basis of that particular authority.

References

1. https://www.novixys.com/blog/setup-apache-hadoop
-cluster-aws-ec2/
2. https://www.9lessons.info/2015/12/amazon-ec2-setup
-with-ubuntu-and-xampp.html
3. https://github.com/airavata-courses/TeamSangam/
wiki/Zookeeper-Installation-on-EC2
4. https://codeforgeek.com/how-to-setup-multi-node
-multi-broker-kafka-cluster-in-aws/
5. https://blog.insightdatascience.com/simply-install-
spark-cluster-mode-341843a52b88
6. https://gist.github.com/aryanc55/21122bcce026e7fe4383e6d13c66b992
7. https://docs.python-guide.org/starting/install3/linux/
8. https://pip.pypa.io/en/stable/installing/
9. https://pypi.org/project/tweepy/
10. https://pypi.org/project/numpy/
11. https://pypi.org/project/kafka-python/
12. https://pypi.org/project/requests/
13. https://pypi.org/project/regex/
14. https://pypi.org/project/MySQL-python/
15. https://console.firebase.googlr.com/u/0/?pli=1
16. https://flutter.dev/
17. https://flutter.dev/docs/development/ui/widgets
18. https://aws.amazon.com/console/
19. Papers on spark(Used in chapter IV and V)
- A Large-Scale Sentiment Data Classification for
Online Reviews Under
Apache Spark Al-Saqqa , Ghazi Al-Naymat ,
Arafat Awajan
DOI 10.1016/j.procs.2018.10.166
- Learning Distributed Discrete Bayesian Network
Classifiers under MapReduce
with Apache Spark
Jacinto Arias, Jose A. Gamez, Jose M. Puerta
DOI: 10.1016/j.knosys.2016.06.013
- Real-Time Sentiment Analysis of Saudi Dialect
Tweets Using SPARK Adel
Assiri, Ahmed Emam, Hmood Al-dossari
DOI: 10.1109/BigData.2016.7841071
- Real-Time Sentiment Analysis of Twitter Streaming
data for Stock Prediction
Sushree Das et al
DOI : 10.1016/j.procs.2018.05.111
20 . Papers on Navie Bayes(Used in chapter IV)
- An Ensemble Classification System for Twitter
Sentiment Analysis

Ankit, Nabizath Saleena
DOI : 10.1016/j.procs.2018.05.109
– A BigData approach for sentiment analysis of
twitter data using Naive Bayes
and SVM Algorithm
Ms. Priyanshu Jadon Dr. Deepshikha Bhatia
Dr. Durgesh Kumar Mishra
DOI: 10.1109/WOCN45266.2019.899510
21. Papers on Kafka(Used in chapter IV and V)
– A Performance Evaluation of Apache Kafka in Support
of Big Data
Streaming Applications
Paul Le Noac h, Alexandru Costan, Luc Boug
DOI: 10.1109/BigData.2017.8258548
– A Study of Apache Kafka in Big Data Stream Processing
Bhole Rahul Hiraman, Chapte Viresh M., Karve Abhijeet C.
DOI: 10.1109/ICICET.2018.8533771
22. https://www.w3schools.com/css/
23. https://www.w3schools.com/php/php_ajax_intro.asp
24. https://www.w3schools.com/php/php_mysql_intro.asp
25. https://www.w3schools.com/js/
26. https://www.w3schools.com/bootstrap/
27. https://stackoverflow.com/questions/6550337/
twitter-oauth-php-need-good-basicexample-to-get-started

9. Appendix(Source Code)

9.1 Machine learning pipeline files

For training machine learning model.
File name: train_model.py

```
import os
import sys
#This section for Windows users only
"""os.chdir(r"C:\spark\spark-files")
os.curdir
if 'SPARK_HOME' not in os.environ:
  os.environ['SPARK_HOME'] = 'C:\spark'
SPARK_HOME = os.environ['SPARK_HOME']
sys.path.insert(0,os.path.join(SPARK_HOME,"python"))
sys.path.insert(0,os.path.join(SPARK_HOME,"python","lib"))
sys.path.insert(0,os.path.join(SPARK_HOME,"python","lib",
"pyspark.zip"))
sys.path.insert(0,os.path.join(SPARK_HOME,"python","lib","py4j
-0.10.4-src.zip"))"""
from pyspark import SparkContext
from pyspark import SparkConf
from pyspark.mllib.feature import HashingTF
from pyspark.mllib.feature import IDF
import operator
conf=SparkConf()
conf.set("spark.executor.memory", "1g")
conf.set("spark.cores.max", "2")
conf.setAppName("IRApp")
sc = SparkContext('local', conf=conf)
tweetData = sc.textFile("data/tweets_formatted_data.csv")
tweetData.take(2)
fields = tweetData.map(lambda x: x.split(","))
fields.take(1)
documents = fields.map(lambda x: x[1].lower().split("␣"))
documents.take(1)
documentNames = fields.map(lambda x: x[0])
hashingTF = HashingTF(100000)
article_hash_value = hashingTF.transform(documents)
article_hash_value.cache()
idf = IDF().fit(article_hash_value)
```

```
tfidf = idf.transform(article_hash_value)
xformedData=tweetData.zip(tfidf)
xformedData.cache()
xformedData.collect()[0]
from pyspark.mllib.regression import LabeledPoint
def convertToLabeledPoint(inVal) :
 origAttr=inVal[0].split(",")
 sentiment = 0.0 if origAttr[0] == "feedback" else 1.0
 return LabeledPoint(sentiment, inVal[1])
tweetLp=xformedData.map(convertToLabeledPoint)
tweetLp.cache()
tweetLp.collect()
from pyspark.mllib.classification import NaiveBayes, NaiveBayesModel
model = NaiveBayes.train(tweetLp, 1.0)
predictionAndLabel = tweetLp.map(lambda p: \
 (float(model.predict(p.features)), float(p.label)))
predictionAndLabel.collect()
#Forming confusion matrix
from pyspark.sql import SQLContext
sqlContext = SQLContext(sc)
predDF = sqlContext.createDataFrame(predictionAndLabel.collect(), \
 ["prediction","label"])
 predDF.groupBy("label","prediction").count().show()
#saving the model
#model.save(sc,"IRModel")
import pickle
with open('IRModel', 'wb') as f:
pickle.dump(model, f)
```

9.2 Streaming and processing

9.2.1 Streaming tweets

Source code for creating a kafka producer to stream live filtered twitter tweets and insert in kafka topic
File name: kafka_filesstream.py

```
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream
import json
import pprint
import csv

access_token = "****************************************"
access_token_secret =  "****************************************"
consumer_key =  "****************************************"
consumer_secret =  "****************************************"


class StdOutListener(StreamListener):
    def on_data(self, data):
        decoded = json.loads(data)
        tweet = 'Failed'
        try:
            #pprint(data)
            tweet = decoded.extended_tweet['text']
        except:
            tweet = decoded["text"]

        print(tweet)
        clss = input()
        fields = []
        if clss == 'e':
            fields = ['emergency',tweet]
        else :
            field = ['feedback',tweet]
        with open(r'new_data.csv', 'a') as f:
            writer = csv.writer(f)
            writer.writerow(fields)

        return True

l = StdOutListener()
auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
stream = Stream(auth, l, tweet_mode='extended')

stream.filter(track=['RailwaySeva',
```

```
                'indian_railways',
                'RailMinIndia',
                'indian_railway',
                'irctc',
                'indianrailway18',
                'IRCTCofficial'],
            stall_warnings=True,
            languages = ['en']
            )
```

9.2.2 Processing tweets

Source code for classifying live stream of tweets from kafka topic 'irtcptwitter'
File name: live_processing.py

```
from pyspark import SparkConf, SparkContext
from pyspark.mllib.classification import  NaiveBayesModel
from pyspark.streaming import StreamingContext
from pyspark.streaming.kafka import KafkaUtils
from pyspark.mllib.feature import HashingTF
from pyspark.mllib.feature import IDF
from pyspark.mllib.regression import LabeledPoint

from pymongo import MongoClient
import datetime
from pprint import pprint
import preprocessor
import re

import operator
import pickle
import json
import dns

mongo_username =  'user'
mongo_password = 'pass'
mongo_host = 'localhost'
mongo_host = 'mongodb+srv://user:passt@irtcpdb
mongo_port = 27017

def insert_tweet(tweet,
                 username,
                 profile_image,
                 pnr,
                 prediction,
                 tweet_id):
    document = {"tweet_id":tweet_id,
                "tweet":tweet,
                "username":username,
                "profile_image_url":profile_image,
```

```
                "pnr":pnr,
                "timestamp": datetime.datetime.utcnow()}
    #pprint(document)
    if(prediction==0):
        try:
            #client = MongoClient('mongodb://{}:{}@{}:{}'
            .format(mongo_username,mongo_password,
             mongo_host,str(mongo_port)))
            db = client.irtcp
            inserted_object = db.feedback.insert_one(document)
            .inserted_id
            print('Inserted : ', inserted_object)
        except:
            print('document insertion FAILED')
        finally:
            client.close()
    else:
        try:
            #client = MongoClient('mongodb://{}:{}@{}:
            {}'.format(mongo_username,mongo_password
            ,mongo_host
            ,str(mongo_port)))
            client = MongoClient('mongodb+srv://user:pass
                @irtcpdb-vdq17.mongodb.net/test?retryWrites=true&w=
                majority&connectTimeoutMS=90000')
            db = client.irtcp
            inserted_object = db.emergency.insert_one(document)
            .inserted_id
            print('Inserted : ', inserted_object)
        except:
            print('document insertion FAILED')
        finally:
            client.close()

from pyspark.streaming import StreamingContext
conf = SparkConf().setMaster("spark://HOST:PORT").
setAppName("Streamer")
sc = SparkContext(conf=conf)
sc.setLogLevel("ERROR")
val = sc.parallelize("abd")


ssc = StreamingContext(sc, 10)
ssc.checkpoint("checkpoint")
kstream = KafkaUtils.createDirectStream(
ssc, topics = ['irtcptwitter'], kafkaParams =
{"metadata.broker.list": 'localhost:9092'})
tweets = kstream.map(lambda x: json.loads(x[1]))
```

```
import joblib
loadedModel = joblib.load('NB.model')

bc_model = sc.broadcast(loadedModel)


def process_data(data):

        print("Processing data ...")

        if (not data.isEmpty()):
            nbModel=bc_model.value
            hashingTF = HashingTF(2**14)
            tf = hashingTF.transform(data.map(lambda x: preprocessor.
            clean(x[0]).encode('utf-8','ignore')))
            tf.cache()
            idf = IDF(minDocFreq=2).fit(tf)
            tfidf = idf.transform(tf)
            tfidf.cache()
            prediction=nbModel.predict(tfidf)

            temp = []
            i=0
            for p,q,r,s in data.collect():
                temp.append([])
                temp[i].append(p) #.encode('utf-8','ignore'))
                temp[i].append(q)
                temp[i].append(r)
                #if re.search("pnr|PNR", tweet_text) and
                re.search("\d{10}", tweet_text):
                    #pnr = re.findall("\d{10}", txt)
                #else:
                    #pnr = []
                temp[i].append('0') #PNR
                temp[i].append(s)
                i+=1
            i=0
            for p in prediction.collect():
                temp[i].append(p)
                i+=1

            #print(temp)
            for i in temp:
                insert_tweet(str(i[0]),str(i[1]),str(i[2]),i[3]
                ,int(i[5]),str(int(i[4])))
        else:
            print("Empty RDD !!!")
            pass
```

```
def tweet_mapper(tweet):

    try:
        #if re.search("pnr|PNR", tweet_text) and re.search("\d{10}",
        tweet_text):
            #pnr = re.findall("\d{10}", txt)
        #else:
            #pnr = []
        tup = (tweet.extended_tweet['full_text'], tweet['user']
        ['screen_name']
        , tweet['user']['profile_image_url_https']
        , tweet['id'])
        print('full text')
        return tup
    except:
        print('no full text')
        return (tweet['text'], tweet['user']['screen_name'],
        tweet['user']['profile_image_url_https'], tweet['id'])


txt = tweets.map(lambda tweet: tweet_mapper(tweet))

txt.foreachRDD(process_data)

#text = tweet_text.map(lambda x: x.encode('utf-8','ignore'))
#text.foreachRDD(process_data)

ssc.start()
ssc.awaitTerminationOrTimeout(1000)
ssc.stop(stopGraceFully = True)
```

9.3 Website

File name: index.php

```
<html>
    <head>
        <title>IR Complaint &amp;amp; Management System</title>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width,
        initial-scale=1">
        <link rel="stylesheet" href="https://maxcdn.
        bootstrapcdn.com/bootstrap/
        3.3.7/css/bootstrap.min.css">
        <script src="https://ajax.googleapis.com/ajax/
        libs/jquery/3.2.0/jquery.min.js"></script>
        <script src="https://maxcdn.bootstrapcdn.com/
        bootstrap/3.3.7/js/bootstrap.min.js"></script>
        <style>

        tr{
            border-bottom: 3px groove;
        }
        td {
            padding:20px;
        }
        .active{color:white;}
        .line{border-right: 2px groove;min-height:98.5vh
        ;overflow-x:hidden;}
        .left{
            width:70%;
            float:left;
            }
        .right{
            float: left;
            width:25%;
            }
        .centered {
            position: absolute;
            top: 40vh;
            left: 0%;
            // transform: translate(-50%, -50%);
            }
        .bg {
            position: relative;
            text-align: center;
            color: white;
            }
        html, body {
            overflow: auto;
            }
```

```
.mystyle{
    top:0;
    position: fixed;
    display: block;
    width: 100%;
    }
.newmystyle{
    position: fixed;
    top:50px;
    margin-left: 70%;
}
</style>
<script>
window.setInterval(function() {
    if (window.XMLHttpRequest) {
        xmlhttp = new XMLHttpRequest();
    } else {
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            bind();
            document.getElementById("feedback-content")
            .innerHTML = this.responseText;
        }
    };
    xmlhttp.open("GET","feedback.php",true);
    xmlhttp.send();
},5000);

window.setInterval(function() {
    if (window.XMLHttpRequest) {
        xmlhttp = new XMLHttpRequest();
    } else {
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            bind();
            document.getElementById("emergency-content")
            .innerHTML = this.responseText;
        }
    };
    xmlhttp.open("GET","emergency.php",true);
    xmlhttp.send();
    },5000);
```

```
    function func(){
        var windowHeight = window.innerHeight;
        var scrolltop = window.pageYOffset;
        var element = document.getElementById("main");
        var newelement = document.getElementById
        ("rightDiv");
        console.log(windowHeight - scrolltop);
        if(windowHeight - scrolltop < '20'){
            element.classList.add("mystyle");
            newelement.classList.add("newmystyle");
            //document.getElementById('main').style
            .padding-top = 50 px;
        } else {
            element.classList.remove("mystyle");
            newelement.classList.remove("newmystyle");
        }
    }

// Code for Accordion

    $(document).ready(function bind(){

    $('body').on('click','.add',function(){
    //... event handler code ....
        document.getElementById("selection").
        innerHTML = "";
        var $this = $(this),
        myCol = $this.closest("td"),
        myRow = myCol.closest("tr"),
        targetArea = $("#selection");
        targetArea.append(myRow.children()
        .not(myCol)
        .text() + "<br />");
        document.getElementById("idvalue").value =
        myRow.children(".value").
        children(".myval").
        val();
        document.getElementById
        ("screen_name").value = myRow.children
        (".value2").children(".myval2")
        .val();
    });
  });


  function sendReply(form){
var tweet_id = form.idvalue.value;
var tweet_reply = form.tweet_reply.value;
var tweet_name = form.screen_name.value;
```

```
    console.log("tweet_id="+tweet_id+"&tweet_reply="+
    tweet_reply+"&screen_name="+tweet_name);

    if (tweet_reply) {
        console.log("tweet_id="+tweet_id+"&tweet_reply=
        +tweet_reply+"&screen_name="+tweet_name);

        if (window.XMLHttpRequest) {
            xmlhttp = new XMLHttpRequest();
        } else {
            xmlhttp = new ActiveXObject("Microsoft.
            XMLHTTP");
            }
            xmlhttp.onreadystatechange = function() {
            if (this.readyState == 4) {
                if (this.status == 200) {
                    alert('Tweet Sent !');
                    form.tweet_reply.value= "";
                } else {
                    alert('Tweet Not Sent !');
                }
            }
        };
        var query = "tweet_id="+tweet_id+"&tweet_reply="
        +tweet_reply+"&screen_name="+tweet_name;
        xmlhttp.open("GET","twitter/index.php?"+query,true);
        xmlhttp.send();

    }

}

</script>
</head>

<body class="line" onscroll="func()">
    <div class="bg" id="header">
        <img src="train.jpg" alt="" style=
        "width:100%;height:100vh;">
        <div class="centered"><p style="font-size:60px;
        ">Railway Complaint & Feedback Managemet
        System</p></div>
    </div>
    <div class="navbar navbar-inverse"
    style="width:101%;
    padding-bottom:1px;" id="main">
        <ul style="font-size: 24px; color:
        gray;margin-top:
        5px;cursor: pointer;">
```

```
            <li class="active" style="display: inline;
            margin-right: 20px"; data-toggle="tab"
            href="#emergency">
                Emergency
            </li>
            <li style="display:inline;" data-toggle="tab"
            href="#feedback">
                Feedback
            </li>
            <li class="active" style="display: inline;float
            :right;margin-right:50px;";><a href=
            "responses.php"
            target="_blank">
                Previous Complaints</a>
            </li>

        </ul>
    </div>
    <div class='tab-content left' style="min-height:90vh;">
        <div id="emergency" class='tab-pane fade in active'>
            <div id='emergency-content' style="width
            :90%;padding: 50px;margin:auto;background:
            white; box-shadow: 0
            4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0
            , 0, 0.19);">
                <table>

                <?php
                    function callAPI($collection){
                        $url = "https://webhooks.mongodb-stitch
                        .com/api
                        /client/v2.0/app/irtcp-qsdyo/service/
                        getData
                        /incoming_webhook/webhook0?collection=
                        ".$collection;
                        $curl = curl_init();
                        curl_setopt($curl, CURLOPT_URL, $url);
                        curl_setopt($curl, CURLOPT_
                        RETURNTRANSFER, true);
                        $result = curl_exec($curl);
                        if(!$result){die("Connection Failure");}
                        curl_close($curl);
                        return $result;
                    }
                    $get_data_emergency = callAPI("emergency");
                    $response_emergency = json_decode
                    ($get_data_emergency,true);
                    $data_emergency = $response_emergency
                    ['documents'];
```

```
                foreach ($data_emergency as $tweet) {
                    echo "<tr><td style='color:black'
                    width='180%'>";
                    echo $tweet["tweet"];
                    echo "</td><td width='20%'><button
                    type='button' class='btn btn-info add'>
                    <b>Reply</b></button></td>";
                    echo "<td class='value'><input type
                    ='hidden' id='tweet_value' class=
                    'myval' name='tweet_value' value=".
                    $tweet["tweet_id"]."></td>";
                    echo "<td class='value2'><input
                    type='hidden' id='user_name' class=
                    'myval2' name='user_name' value="
                    .$tweet["username"]."></td></tr>";

                }
            ?>

            </table>
        </div>
    </div>
    <div id="feedback" class='tab-pane fade'>
        <div id='feedback-content' style="width:90%;
        padding: 50px;margin:auto;background:white;
        box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2),
        0 6px 20px 0 rgba(0, 0, 0, 0.19);">
            <table>
                <?php
                    $get_data_feedback = callAPI
                    ("feedback");
                    $response_feedback = json_decode
                    ($get_data_feedback,true);
                    $data_feedback = $response_feedback
                    ['documents'];
                    foreach ($data_feedback as $tweet)
                    {
                        echo "<tr><td style='color:black'
                        width='180%'>";
                        echo $tweet["tweet"];
                        echo "</td><td width='20%'><button
                        type='button' class='btn btn-info add'>
                        <b>Reply</b></button></td>";
                        echo "<td class='value'><input
                        type='hidden' id='tweet_value'
                        class='myval' name='tweet_value'
                        value=".$tweet["tweet_id"]."></td>";
                        echo "<td class='value2'><input
                        type='hidden' id='user_name'
```

```
                                    class='myval2' name='user_name'
                                    value=".$tweet["username"]."></td></tr>";

                                }
                            ?>
                        </table>
                    </div>
                </div>
            </div>
            <div class="container-fluid right" style="padding-top:50px
            ;" id="rightDiv">
                <div id="selection" style="min-height:200px;box-shadow
                : 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba
                (0, 0, 0, 0.19);font-size: 16px;padding:10px;"></div>
                <h3 style="color:#c6c6c6;">Type your response...</h3>
                <form>
                    <input type="hidden" name="idvalue" id="idvalue"
                    value="">
                    <input type="hidden" name="screen_name" id="
                    screen_name" value="">
                    <textarea id="tweet_reply" name="tweet_reply"
                    rows="10" style="width:100%;"></textarea>
                    <button type="button" class="btn btn-info"
                    style="width:100%;margin-top: 10px;" onclick=
                    "sendReply(this.form)">Reply</button>
                </form>
            </div>
        </body>
    </html>
    <?php
        function callAPI($collection){
            $url = "https://webhooks.mongodb-stitch.com/api/
            /webhook0?collection=".$collection;
            $curl = curl_init();
            curl_setopt($curl, CURLOPT_URL, $url);
            curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
            $result = curl_exec($curl);
            if(!$result){die("Connection Failure");}
            curl_close($curl);
            return $result;
        }
        $get_data = callAPI("emergency");
        $response = json_decode($get_data,true);

        $data = $response['documents'];
        foreach ($data as $tweet) {
            echo $tweet["tweet"] . "\n";
        }
```

```
    foreach ($data as $tweet) {
        echo "<tr><td style='color:black' width='180%'>";
        echo $tweet["tweet"];
        echo "</td><td width='20%'><button type='button'
        class='btn btn-info add'><b>Reply</b></button></td>";
        echo "<td class='value'><input type='hidden' id='tweet_value'
        class='myval' name='tweet_value' value=".$tweet["tweet_id"].">
        </td></tr>";
        $i++;
    }
?>
<?php
    function callAPI($collection){
        $url = "https://webhooks.mongodb-stitch.com/api/
        webhook0?collection=".$collection;
        $curl = curl_init();
        curl_setopt($curl, CURLOPT_URL, $url);
        curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
        $result = curl_exec($curl);
        if(!$result){die("Connection Failure");}
        curl_close($curl);
        return $result;
    }
    $get_data = callAPI("feedback");
    $response = json_decode($get_data,true);

    $data = $response['documents'];
    foreach ($data as $tweet) {
        echo $tweet["tweet"] . "\n";
    }

    foreach ($data as $tweet) {
        echo "<tr><td style='color:black' width='180%'>";
        echo $tweet["tweet"];
        echo "</td><td width='20%'><button type='button'
        class='btn btn-info add'><b>Reply</b></button></td>";
        echo "<td class='value'><input type='hidden'
        id='tweet_value' class='myval' name='tweet_value' value="
        .$tweet["tweet_id"]."></td></tr>";
        $i++;
    }
?>

<?php

require 'vendor/autoload.php';
use Abraham\TwitterOAuth\TwitterOAuth;

define('CONSUMER_KEY', '**************************');
```

```
define('CONSUMER_SECRET', '************************');
define('ACCESS_TOKEN', '************************');
define('ACCESS_TOKEN_SECRET', '********************');

$connection = new TwitterOAuth(CONSUMER_KEY, CONSUMER_SECRET
, ACCESS_TOKEN, ACCESS_TOKEN_SECRET);
$content = $connection->get("account/verify_credentials");
$qtweet = $_REQUEST['reply'];
$tweetid = $_REQUEST['tweet_id'];
$connection->post('statuses/update', array('status' =>
$qtweet, 'in_reply_to_status_id' => $tweetid,'auto_populate
_reply_metadata'=>'true'));
var_dump($connection->getLastHttpCode());
?>

<?php

define('CONSUMER_KEY', '**************************');
define('CONSUMER_SECRET', '************************');
define('ACCESS_TOKEN', '**************************');
define('ACCESS_TOKEN_SECRET', '********************');

$connection = new TwitterOAuth(CONSUMER_KEY,
CONSUMER_SECRET, ACCESS_TOKEN, ACCESS_TOKEN_SECRET);
$content = $connection->get("account/verify_
credentials");/*
$qtweet = $_REQUEST['reply'];
$tweetid = $_REQUEST['tweet_id'];
$connection->post('statuses/update', array('status'
=> $qtweet, 'in_reply_to_status_id' => $tweetid,'
auto_populate_reply_metadata'=>'true'));
var_dump($connection->getLastHttpCode());
*/

$status_id = '480775609728385024';

$twitt_reply = '@username the replay text';


$responce = $connection->post('statuses/update',
array('in_reply_to_status_id'=> $status_id , 'status'
=> $twitt_reply ));
var_dump($responce->getLastHttpCode());
?>
```

9.4 Mobile application

File name: lib/main.dart

```
import 'package:flutter/material.dart';
import './home.dart';
import './strings.dart';

void main() => runApp(IrtcpApp());

class IrtcpApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: Strings.appTitle,
      theme: ThemeData(
        primaryColor: Colors.white,
        accentColor: Colors.blue,
        accentColorBrightness: Brightness.dark,
      ),
      home: Home(),
    );
  }
}
```

File name: lib/views/reply.dart

```
import 'dart:convert';

import 'package:flutter/material.dart';
import 'package:hello/presentation/custom_icons_icons.dart';
import 'package:intl/intl.dart';

import '../models/tweets.dart';

class Reply extends StatelessWidget {
  final Documents tweet;
  Reply({this.tweet});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        color: Colors.blue,
        padding: EdgeInsets.fromLTRB(16.0, 16.0, 16.0, 16.0),
        child: ListView(
          children: <Widget>[
            Stack(
              children: [
                Align(
                  alignment: Alignment.topRight,
```

```
    child: Container(
      padding: EdgeInsets.all(24.0),
      child: Icon(
        CustomIcons.twitter,
        color: Colors.white,
        size: 100.0,
      ),
    ),
  ),
  Align(
    alignment: Alignment.topLeft,
    child: Container(
      child: IconButton(
        icon: Icon(Icons.arrow_back_ios),
        color: Colors.white,
        iconSize: 48.0,
        onPressed: () {
          Navigator.pop(context);
        },
      ),
    ),
  ),
  Align(
    alignment: Alignment.topLeft,
    child: Column(
      crossAxisAlignment: CrossAxisAlignment
      .start,
      children: [
        SizedBox(height: 100.0),
        RichText(
          text: TextSpan(
            text: 'On␣',
            style: TextStyle(
              fontSize: 18.0,
              color: Colors.white,
            ),
            children: <TextSpan>[
              TextSpan(
                  text: DateFormat('MMMM␣d,␣y')
                          .format(DateTime
                          .fromMillisecond
                          sSinceEpoch(int.parse(
                              tweet.timestamp.date.
                              numberLong))) +
                      ',',
                  style: TextStyle(
                    fontSize: 30.0,
                    fontWeight: FontWeight.w500,
                    color: Colors.white,
```

```
                        ))
                ],
              ),
            ),
            SizedBox(
              height: 14.0,
            ),
            Row(
              children: <Widget>[
                Container(
                  padding: EdgeInsets.all(8.0),
                  child: CircleAvatar(
                    backgroundColor: Colors.blue,
                    radius: 30.0,
                    child: CircleAvatar(
                      backgroundImage:
                          NetworkImage(tweet.profile
                          ImageUrl),
                      radius: 30.0,
                    ),
                  ),
                ),
                RichText(
                  text: TextSpan(
                    text: '@' + tweet.username + '␣',
                    style: TextStyle(
                      fontSize: 34,
                      fontWeight: FontWeight.w500,
                      color: Colors.white,
                    ),
                  ),
                ),
              ],
            ),
            // RichText(
            //   text: TextSpan(
            //     text: 'tweeted : ',
            //     style: TextStyle(
            //       fontSize: 18.0,
            //       color: Colors.white,
            //     ),
            //   ),
            // ),
          ],
        ),
      )
    ],
  ),
  Card(
```

```
    shadowColor: Colors.black,
    shape: RoundedRectangleBorder(
      side: BorderSide(color: Colors.blue),
      borderRadius: BorderRadius.circular(16.0),
    ),
    elevation: 18.0,
    child: Container(
      padding: EdgeInsets.all(8.0),
      child: Text(
        utf8.decode(tweet.tweet.toString().codeUnits),
        style: TextStyle(
            fontSize: 38.0,
            fontWeight: FontWeight.w900,
            fontStyle: FontStyle.italic,
            color: Colors.deepOrangeAccent[200],
            fontFamily: 'Open Sans'),
      ),
    ),
  ),
  SizedBox(
    height: 14.0,
  ),
  Row(
    children: <Widget>[
      Container(
        padding: EdgeInsets.all(8.0),
        child: CircleAvatar(
          backgroundColor: Colors.blue,
          radius: 30.0,
          child: CircleAvatar(
            backgroundImage: AssetImage('assets/logo.png'),
            radius: 30.0,
          ),
        ),
      ),
      RichText(
        text: TextSpan(
          text: '@IndianRailways ',
          style: TextStyle(
            fontSize: 34,
            fontWeight: FontWeight.w500,
            color: Colors.white,
          ),
        ),
      ),
    ],
  ),
  Card(
    shadowColor: Colors.black,
```

```
                shape: RoundedRectangleBorder(
                  side: BorderSide(color: Colors.blue),
                  borderRadius: BorderRadius.circular(16.0),
                ),
                elevation: 18.0,
                child: Container(
                  padding: EdgeInsets.all(8.0),
                  child: TextField(
                    autocorrect: true,
                    maxLength: 240,
                    maxLines: null,
                    keyboardType: TextInputType.multiline,
                    textInputAction: TextInputAction.send,
                    textCapitalization: TextCapitalization.sentences,
                    textAlign: TextAlign.justify,
                    decoration: InputDecoration.collapsed(hintText:
                    'Your Reply'),
                    style: TextStyle(
                        fontSize: 38.0,
                        fontWeight: FontWeight.w900,
                        fontStyle: FontStyle.italic,
                        color: Colors.blue,
                        fontFamily: 'Open Sans'),
                  ),
                ),
              ),
              SizedBox(
                height: 14.0,
              ),
              Row(
                mainAxisAlignment: MainAxisAlignment.end,
                children: [
                  FloatingActionButton.extended(
                    onPressed: () {},
                    backgroundColor: Colors.deepOrangeAccent[200],
                    label: Text('Send'),
                    icon: Icon(
                      CustomIcons.send,
                      color: Colors.white,
                    ),
                  )
                ],
              ),
            ],
          ),
        ),
      );
  }
}
```

File name: lib/views/tweet.dart

```
import 'package:flutter/material.dart';
import 'package:hello/presentation/custom_icons_icons.dart';

class Tweet extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        color: Colors.blue,
        padding: EdgeInsets.fromLTRB(16.0, 16.0, 16.0, 16.0),
        child: ListView(
          children: <Widget>[
            Stack(
              children: [
                Align(
                  alignment: Alignment.topRight,
                  child: Container(
                    padding: EdgeInsets.all(24.0),
                    child: Icon(
                      CustomIcons.twitter,
                      color: Colors.white,
                      size: 100.0,
                    ),
                  ),
                ),
                Align(
                  alignment: Alignment.topLeft,
                  child: Container(
                    child: IconButton(
                      icon: Icon(Icons.arrow_back_ios),
                      color: Colors.white,
                      iconSize: 48.0,
                      onPressed: () {
                        Navigator.pop(context);
                      },
                    ),
                  ),
                ),
              ],
            ),
            SizedBox(
              height: 14.0,
            ),
            Row(
              children: <Widget>[
                Container(
                  padding: EdgeInsets.all(8.0),
                  child: CircleAvatar(
```

```
          backgroundColor: Colors.blue,
          radius: 30.0,
          child: CircleAvatar(
            backgroundImage: AssetImage('assets/logo.png'),
            radius: 30.0,
          ),
        ),
      ),
      RichText(
        text: TextSpan(
          text: '@IndianRailways ',
          style: TextStyle(
            fontSize: 34,
            fontWeight: FontWeight.w500,
            color: Colors.white,
          ),
        ),
      ),
    ],
  ),
  Card(
    shadowColor: Colors.black,
    shape: RoundedRectangleBorder(
      side: BorderSide(color: Colors.blue),
      borderRadius: BorderRadius.circular(16.0),
    ),
    elevation: 18.0,
    child: Container(
      padding: EdgeInsets.all(8.0),
      child: TextField(
        autocorrect: true,
        maxLength: 240,
        maxLines: 9,
        keyboardType: TextInputType.multiline,
        textInputAction: TextInputAction.send,
        textCapitalization: TextCapitalization.sentences,
        textAlign: TextAlign.justify,
        decoration: InputDecoration
          .collapsed(hintText: 'Your Tweet'),
        style: TextStyle(
            fontSize: 38.0,
            fontWeight: FontWeight.w900,
            fontStyle: FontStyle.italic,
            color: Colors.blue,
            fontFamily: 'Open Sans'),
      ),
    ),
  ),
  SizedBox(
```

```
              height: 14.0,
            ),
            Row(
              mainAxisAlignment: MainAxisAlignment.end,
              children: [
                FloatingActionButton.extended(
                  onPressed: () {},
                  backgroundColor: Colors.deepOrangeAccent[200],
                  label: Text('Send'),
                  icon: Icon(
                    CustomIcons.send,
                    color: Colors.white,
                  ),
                )
              ],
            )
          ],
        ),
      ),
    );
  }
}
```

File name: lib/views/emergency.dart

```
import 'package:flutter/material.dart';
import 'package:hello/presentation/custom_icons_icons.dart';
import 'package:hello/strings.dart';
import 'dart:convert';
import 'package:http/http.dart' as http;

import '../models/tweets.dart';
import './reply.dart';

class Emergency extends StatefulWidget {
  final String collection;
  Emergency(this.collection);

  @override
  State<StatefulWidget> createState() => EmergencyState(collection);
}

class EmergencyState extends State<Emergency> {
  Future<Tweets> futureTweets;
  String collection;
  EmergencyState(this.collection);

  @override
  void initState() {
    super.initState();
    futureTweets = _fetchTweets(collection);
```

```
  }

  @override
  Widget build(BuildContext context) {
    return Center(
      child: FutureBuilder(
        future: futureTweets,
        builder: (context, snapshot) {
          if (snapshot.hasData) {
            return ListView.builder(
              itemCount: snapshot.data.documents.length,
              itemBuilder: (context, position) {
                return getTweetView(context, snapshot.
                data.documents[position],
                    snapshot.data.documents);
              },
            );
          }
          return CircularProgressIndicator();
        },
      ),
    );
  }
}

Future<Tweets> _fetchTweets(String collection) async {
  final response = await http
      .get(Strings.fetchDocumentsBaseUrl + 'collection=' + collection);
  if (response.statusCode == 200) {
    return Tweets.fromJson(json.decode(response.body));
  } else {
    throw Exception('Can\'t Connect');
  }
}

Future<String> deleteTweet(Documents tweetDoc) async {
  final response = await http
      .get(Strings.deleteDocumentBaseUrl + 'tweetId=' + tweetDoc.tweetId);
  if (response.statusCode == 200) {
    return response.body;
  } else {
    throw Exception('Deleting tweet failed');
  }
}

Widget getTweetView(
    BuildContext context, Documents tweetDoc,
    List<Documents> documents) {
  return Dismissible(
```

```
key: Key(tweetDoc.tweetId),
onDismissed: (direction) {
  documents.remove(tweetDoc);
  deleteTweet(tweetDoc);
},
direction: DismissDirection.horizontal,
background: Container(
  foregroundDecoration: BoxDecoration(color:
  Colors.transparent),
  padding: EdgeInsets.all(16.0),
  child: Row(
    mainAxisAlignment: MainAxisAlignment.start,
    children: <Widget>[
      Icon(
        CustomIcons.delete,
        color: Colors.red,
        size: 48.0,
      ),
    ],
  ),
),
secondaryBackground: Container(
  padding: EdgeInsets.all(16.0),
  child: Row(
    mainAxisAlignment: MainAxisAlignment.end,
    children: <Widget>[
      Icon(
        CustomIcons.delete,
        color: Colors.red,
        size: 48.0,
      ),
    ],
  ),
),
child: Card(
  shadowColor: Colors.black,
  shape: RoundedRectangleBorder(
    side: BorderSide(color: Colors.blue),
    borderRadius: BorderRadius.circular(16.0),
  ),
  margin: EdgeInsets.fromLTRB(16.0, 8.0, 16.0, 8.0),
  color: Colors.white,
  elevation: 8.0,
  child: Column(
    children: <Widget>[
      Row(
        children: <Widget>[
          Container(
            padding: EdgeInsets.all(8.0),
```

```
        child: CircleAvatar(
          backgroundColor: Colors.blue,
          radius: 25.0,
          child: CircleAvatar(
            backgroundImage: NetworkImage(tweetDoc
            .profileImageUrl),
            radius: 24.0,
          ),
        ),
      ),
      Expanded(
        child: Text(
          '@' + tweetDoc.username,
          textScaleFactor: 1.5,
          style: TextStyle(
            color: Colors.blue,
          ),
        ),
      ),
      Container(
        padding: EdgeInsets.all(16.0),
        child: Text(
          DateTime.now()
                  .difference(DateTime.fromMicros
                  econdsSinceEpoch(
                      int.parse(tweetDoc.timestamp
                      .date.numberLong) *
                          1000))
                  .inDays
                  .toString() +
              'd',
          style: TextStyle(color: Colors.grey),
        ),
      ),
    ],
  ),
  Container(
    padding: EdgeInsets.fromLTRB(8.0, 0, 8.0, 8.0),
    child: Text(
      utf8.decode(tweetDoc.tweet.toString().codeUnits),
      style: TextStyle(
        color: Colors.grey,
      ),
      textScaleFactor: 1.5,
    ),
  ),
  Row(mainAxisAlignment: MainAxisAlignment.end,
  children: <Widget>[
    FloatingActionButton(
```

```
                elevation: 8.0,
                mini: true,
                heroTag: tweetDoc.iId.oid,
                tooltip: 'Reply',
                onPressed: () {
                  Navigator.push(
                    context,
                    MaterialPageRoute(
                        builder: (context) => Reply(tweet:
                        tweetDoc)),
                  );
                },
                focusColor: Colors.blue,
                child: Icon(
                  CustomIcons.send,
                  color: Colors.white,
                ),
              ),
            ]),
          ],
        ),
      ),
    );
  }
```