

ANALOGIZE

A project report submitted by

ANJANA DAVIS(GTATSCS001)

BIJEESH P B (GTATSC010)

JEEVA V(GTATSCS012)

JAIN JOSEPH(GTATSCS029)

Under the guidance of

Dr.JEEVAMOL JOY

ASSISTANT PROFESSOR

For the award of the Degree of

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

(Calicut University)

2021-2022

Submitted to



Department of Computer Science
Sri. C. AchuthaMenon Government College, Thrissur
Kuttanellur, Kerala

DECLARATION

*We wish to state that the work embodied in this project titled **ANALOGIZE** forms our own contribution to the project work carried out under the guidance of Dr.Jeevamol Joy, assistant professor, Department of computer science SRI.C.Achutha Menon Government College Kuttanellur Thrissur . We hereby declare that this work is submitted to University of Calicut in partial fulfillment of the requirement for the award of the degree of Bachelor of Computer Science and this submission contains no material previously published or written by another person nor material which 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.*

ANJANA DAVIS(GTATSCS001)

BIJEESH P B (GTATSCS010)

JEEVA V(GTATSCS012)

JAIN JOSEPH (GTATSCS029)

Place :

Date :

ANALOGIZE

Contents

ANALOGIZE	page no:
1 Introduction	9
1.1 Name of the System	11
1.2 Features of Existing Systems	11
1.3 Limitations of Existing Systems	11
1.4 Area and Category of the Project Work	12
2 Problem Definition and Methodology	13
2.1 Problem Definition	13
2.2 Motivation	14
2.3 Objectives	14
2.4 Methodology	14
25 Scope	15
3 Analysis	16
3.1 Requirement Analysis	16
3.2 Existing System	16
3.3 Proposed System	16
3.4 Requirement Specification	23
3.4.1 Functional Requirements	23
3.4.2 Non-functional Requirements	24
3.4.3 Hardware Requirements	26
3.4.4 Software Requirements	26
3.4.5 Other Requirements	27
3.5 Feasibility Study	31
3.5.1 Technical feasibility	31
3.5.2 Economic feasibility	32
3.5.3 Operational feasibility	32
4 Design	34
4.1 Introduction	34
4.2 Modularity criteria	34

4.3	Architecture diagrams/DFD	35
4.3.1	DFD Level 0	37
4.3.2	DFD Level 1	37
4.3.3	DFD Level 2	38
4.4	Use Case Diagrams	39
4.5	User interface layout	39
4.5.1	register page	41
4.5.2	Login page	42
4.5.3	Dashboard	43
4.6	Structure of reports being created	43
4.7	Database design	43
4.7.1	List of entities and attributes	45
4.7.2	E-R Diagram	47
4.7.3	Structure of tables	50
5	Implementation	53
5.1	Introduction	53
5.2	Tools/scripts for implementation	53
5.3	Process logic	54
5.4	Coding	54
5.5	Screenshots	61
5.5.1	Home page	61
5.5.2	Registration page	62
5.5.3	login Page	62
5.5.4	Dashboard	63
6	Testing	64
6.1	Introduction	64
6.2	System testing	65
6.2.1	Modularity testing	65
6.2.2	Integration Testing	66

6.3	Test plan and test cases	67
6.3.1	login screen	67
7	Conclusion	68
	References	70

List of Figures

(Section No)	Caption	Page No
4.3.1	DFD level 0	37
4.3.2	DFD level 1	37
4.3.3	DFD level 2	38
4.4	Use case diagram	39
4.5.1	Register page	41
4.5.2	Login page	42
4.5.3	Dashboard	43
4.7.2	E-R diagram	47

List of Tables

(Section.no)	Caption	Page No
4.7.3	Structure of table	
1	Registration table	50
2	Login table	51
3	Post Details table	51
4	Complaint details	51
5	Feedback details	52
6	Admin	52
7	Anomaly post details	52

Chapter 1

1.Introduction

Social media has become a vital part of humans' day to day life. Different users engage with social media differently. With the increased usage of social media, many researchers have investigated different aspects of social media. Many examples in the recent past show, content in social media can generate violence in the user community. Violence in social media can be categorised into aggregation in comments, cyber-bullying and incidents like protests, murders. Identifying violent content in social media is a challenging task: social media posts contain both the visual and text as well as these posts may contain hidden meaning according to the users' context and other background information.

Thuseethan and Vasanthapriyan conducted research to analyze social media as a new trend in Sri Lankan digital journalism (Thuseethan and Vasanthapriyan, 2015). They identified six main categories of social media as social networking, bookmarking sites, social news, media sharing, micro blogging and blogs. The impact on social media has risen all over the world and most of the countries are conducting research to find suitable methods to analyze and classify the content in social media using some mechanisms. One such research (Naher and Minar, 1997) was conducted in Bangladesh to analyze the case studies of the incidents where violence originated from social media websites like Facebook. The recommendation of this research is to use Artificial Intelligence (AI) systems to predict and prevent violent behaviors in social media websites. Popular social media websites are Facebook, Twitter, YouTube, WhatsApp, and Viber. These websites provide a number of ways to spread information.

One such widely used method is TWITTER. Twitter is a free networking microblogging service that allows registered members to broadcast short posts called tweets. Twitter members can broadcast tweets and follow other users tweets by using multiple platforms and devices. The main purpose is to connect people and allow people to share their thoughts with a big audience. Focused communication as its defined within 140 characters is the best thing in twitter. Twitter allows for crisp and targeted communication by restricting its word count to just 140 characters. For people on the go, businessmen and users who tie for time, a simple gist of the latest news put out by their brand is what they seek. The interesting fact is politicians and ministers of India widely use Twitter for their announcements, this will make a clear of how twitter is influenced in our day to day life. These are the main reasons for selecting twitter as the main aspect in our project.

The model or the architecture of the systems heavily uses natural language processing and machine learning approaches. Sentiment analysis and topic classification approaches have also been used in this task. sentimental analysis is used to determine whether a given text contains negative, positive, or neutral emotions. its a form of text analytics that uses natural language processing and machine learning. This paper summarizes and compares these approaches in the context of analyzing a topic through sentimental analysis.

1.1 ANALOGIZE

Analyzing a person or a topic which has been discussed in the twitter platform by using python. The output of analysis will be a statistical score containing negative, positive and neutral emotions. The keypoint is sentimental analysis.

1.2 Features of Existing Systems

Since the Advent of the Internet, humans have used it as a communication tool in the form of mostly text messages and nowadays video and audio streams and as we increase our dependence on technology it becomes increasingly important to better gauge human sentiments expressed with the help of technology. However, in this textual communication data, we lose the access to sentiments or the emotions conveyed behind a sentence, as we often use our hands and facial expressions to express our intent behind the statement. From this textual data, we can gain insights into the individual. Insights which can be used for multiple different uses such as content recommendation based on current mood, market segmentation analysis and psychological analysis in humans.

1.3 Limitations of Existing Systems

Currently many people are using the internet as a central platform to find the information about reality in the world and need to continue. It has been mentioned above that it is difficult to visit all the tweets. It will take more time to analyze a person by his person or to study about a particular topic. Also whose use of our website can see the overall sentiments connected to the topic that have been inputted.

1.4 Area and Category of the Project Work

The project is intended for everyone who has twitter accounts. Anyone can login and then use the website. The project is categorized as a website and can be used by anyone with access to the internet and a web browsing software

Chapter 2

Problem Definition & Methodology

2.1 Problem Definition

Sentiment analysis in the domain of micro-blogging is a relatively new research topic so there is still a lot of room for further research in this area. Decent amount of related prior work has been done on sentiment analysis of user review, documents, web blogs/articles and general phrase level; sentimental analysis. These differ from twitter mainly because of the limit of 140 characters per tweet which forces the user to express opinion compressed in very short text. The best results reached in sentiment classification use supervised learning techniques such as Naive Bayes and support Vector machines, but the manual labeling required for the supervised approach is very expensive. Some work has been done on unsupervised and semi-supervised approaches, and there is a lot of room of improvement. Various researchers testing new features and classification techniques often just compare their results to base-line performance. There is a need for proper and formal comparisons between these results arrived through different features and classification techniques in order to select the best features and efficient classifications techniques for particular applications.

2.2 Motivation

An aspect of social media data such as twitter messages is that it includes rich structured information about the individuals involved in the communication.

It can lead to more accurate tools for extracting semantic information. It provides means for empirically studying properties of social interactions. Freely available, annotated corpus, Pre-written classifier codes in python using NLTK that can be used in NLP in order to promote research that will lead to a better understanding of how sentiment is conveyed in tweets and texts.

2.3 Objectives

The objectives of the study are first, to study the sentimental analysis in microblogging which a view to analyze feedback from a customer of an organization's product; and second, is to develop a program for customers review on a product which allows an organization's or individual to sentiment and analyze a vast amount of tweets into a useful format.

2.4 Methodology

To achieve this objective discussed above in section the following methodology is used:

- A thorough study of existing approaches and techniques in field of sentiment analysis.
- Collection of related data from Twitter with the help of Twitter API
- Pre-processing of data collected from Twitter so that it can be fit for mining.
- To build a classifier based on different supervised machine learning techniques.
- Training and testing of build classifier using large datasets

2.6 Scope

Nowadays most of the research scholars have been working on Twitter and Youtube comments data sets. To perform sentiment analysis the most common source of data set are web page ,social website like facebook,twitter,youtube etc. There is a vast scope for research scholars to increase the accuracy level up to some extent by using well designed sentence structure. But sarcastic comments are the ones which are very difficult to identify.

Scope for ANALOGIZE in various and explained below

Business: Companies use Twitter Sentiment Analysis to develop their business strategies, to assess customers' feelings towards products or brands, how people respond to their campaigns or product launches and also why consumers are not buying certain products.

Politics: In politics Sentiment Analysis Dataset Twitter is used to keep track of political views, to detect consistency and inconsistency between statements and actions at the government level. Sentiment Analysis Dataset Twitter is also used for analyzing election results.

Public Actions: Twitter Sentiment Analysis also is used for monitoring and analyzing social phenomena, for predicting potentially dangerous situations and determining the general mood of the blogosphere.

Chapter 3

Analysis

3.1 Requirement Analysis

Requirement Analysis Requirements analysis focuses on the tasks that determine the needs or conditions to meet the new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements. Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

3.2 Existing System

The existing system is checking all the tweets by reading it. This process will consume a lot of time and need too much manpower. There is no easy method to analyze a particular topic selected by the user.

3.3 Proposed System

Our team gives an opportunity to check a particular topic which is selected by the user. The website contains a search button. We can input a topic and the website will provide a statistical result with figures and numerical scores. Later the user can come to a conclusion about the topic. The proposed system consists of mainly four modules, they are:

- User authentication

- User options modules
- Tweet analysis
- Admin

User authentication

register

User have to register with the username,email,password,address,phone number

Login

Once registered the user can login with username and password

User options

On successful login, the user can do the following :-

- Search tweets
- View abusive contents & percentage
- View polarity score and tweets
- View sentiment analysis
- View sentiment score
- View different graphs
- Send complaints
- Send feedbacks

Machine Learning

Machine learning is a branch of artificial intelligence (AI) focused on building applications. They learn from data and improve their accuracy over time without being programmed to do. Machine learning focuses on the development of computer programs that can access data and use it to learn themselves.

In data science, an algorithm is a sequence of statistical processing steps. In machine learning, algorithms are “trained” to find patterns and features in massive amounts of data in order to make decisions and predictions based on new data. The better the algorithm, the more accurate the decisions and predictions will become as it processes more data.

NLTK

Natural Language Toolkit (NLTK) is a library in Python, which provides a base for building programs and classification of data. NLTK is a collection of resources for Python that can be used for text processing, classification, tagging and tokenization. This toolbox plays a key role in transforming the text data in the tweets into a format that can be used to extract sentiment from them. NLTK provides various functions which are used in pre-processing of data so that data available from twitter become fit for mining and extracting features. NLTK supports various machine learning algorithms which are used for training classifiers and to calculate the accuracy of different classifiers. In our thesis we use Python as our base programming language which is used for writing code snippets. Tweet analysis uses the Tweepy API for generating sentiment scores. It uses Vader sentiment analysis of NLTK for this purpose. NLTK is a library of Python which plays a very important role in converting natural language text to a sentiment either positive or negative. NLTK also provides different sets of data which are used for training classifiers. These datasets are structured and stored in the library of NLTK, which can be accessed easily with the help of Python.

Tweet analysis

Uses the Tweepy API for generating sentiment scores. It uses the Vader sentiment analysis of NLTK for this purpose.

Tweepy api

Tweepy is an open source Python package that gives you a very convenient way to access the Twitter API with Python. Tweepy includes a set of classes and methods that represent Twitter's models and API endpoints, and it transparently handles various implementation details, such as:

- Data encoding and decoding
- HTTP requests
- Results pagination
- OAuth authentication
- Rate limits
- Streams

If you weren't using Tweepy, then you would have to deal with low-level details having to do with HTTP requests, data serialization, authentication, and rate limits. This could be time consuming and prone to error. Instead, thanks to Tweepy, you can focus on the functionality you want to build.

Almost all the functionality provided by Twitter API can be used through Tweepy. The only current limitation, as of version 3.7.0, is that Direct Messages don't work properly due to some recent changes in the Twitter API.

SENTIMENTAL ANALYSIS USING VADER

Interpretation and classification of emotions sentiment analysis is a text analysis method that detects polarity (e.g. a positive or negative opinion) within the text, whether a whole document, paragraph, sentence, or clause.

Sentiment analysis aims to measure the attitude, sentiments, evaluations, attitudes, and emotions of a speaker/writer based on the computational treatment of subjectivity in a text. Sentiment Analysis is a tricky subject. A text may contain multiple sentiments all at once. For instance,

“The acting was good , but the movie could have been better”

The above sentence consists of two polarities!!!

VADER

VADER (Valence Aware Dictionary for Sentiment Reasoning) is a model used for text sentiment analysis that is sensitive to both polarity (positive/negative) and intensity (strength) of emotion. It is available in the NLTK package and can be applied directly to unlabeled text data.

VADER sentiment analysis relies on a dictionary that maps lexical features to emotion intensities known as sentiment scores. The sentiment score of a text can be obtained by summing up the intensity of each word in the text.

For example- Words like ‘love’, ‘enjoy’, ‘happy’, ‘like’ all convey a positive sentiment. Also VADER is intelligent enough to understand the basic context of these words, such as “did not love” as a negative statement. It also understands the emphasis of capitalization and punctuation, such as “ENJOY”

Polarity classification. We won’t try to determine if a sentence is objective or subjective, fact or opinion. Rather, we care only if the text expresses a positive, negative or neutral opinion. Coarse analysis

We won’t try to perform a fine-grained analysis that would determine the degree of positivity/negativity. That is, we’re not trying to guess how many stars a reviewer awarded, just whether the review was positive or negative.

Broad Steps:

First, consider the text being analyzed. A model trained on paragraph-long reviews might not be effective. Make sure to use an appropriate model for the task at hand.

Next, decide the type of analysis to perform. Some rudimentary sentiment analysis models go one step further, and consider two-word combinations, or bigrams. We will be going to work on complete sentences, and for this we're going to import a trained NLTK lexicon called VADER.

DATASETS TO USE

For this model you can use a variety of datasets like amazon reviews, movie reviews, or any other reviews for any product.

```
import nltk
nltk.download('vader_lexicon')
from nltk.sentiment.vader import SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()
```

VADER's `SentimentIntensityAnalyzer()` takes in a string and returns a dictionary of scores in each of four categories:

```
negative
neutral
positive
compound (computed by normalizing the scores above)
```

Twitter api

The Twitter API lets you read and write Twitter data. Thus, you can use it to compose tweets, read profiles, and access your followers' data and a high volume of tweets on particular subjects in specific locations.

API stands for Application Programming Interface. This software provides "middleman services" between two applications that want to communicate with each other. Any requests you make go to the server first and the response given comes through the same route.

This bridges the gap between the two applications (the one on your phone and the company's software) while providing an extra layer of security. For instance, whenever you use a social media application, you interact with an API. Without direct contact between each other, any potential threats posed by one to the other are abated.

An API also provides a list of commands that can be executed. To use a popular analogy, imagine you're ordering food at a restaurant. You're at the table, ordering from the kitchen. The link between you and the kitchen is the waiter who ensures that the kitchen staff gets all the orders and meals are delivered to the right customers. You're also free to ask them what's available before making your order.

Without the waiter, you'd be pushing your way into the kitchen yourself, trying to get something to eat. That's where the API plays its role as the middle man between two pieces of software.

The API provides a list of methods the two applications can use to communicate. They include:

- GET for retrieving data.
- POST for creating data.
- PUT for updating data.
- DELETE for removing data.

Features of a twitter API

There are four main objects: Tweets, Entities, Places, and Users. There are daily restrictions: Calls and changes in the API are restricted by access tokens to protect the platform from abuse. It is based on HTTP (rather than SSL). There are specific

measures to adapt the API operation to the social network including library restrictions, generated paging, and specific parameters.

3.4 Requirement Specification

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function (in a market-driven project, these roles may be played by the marketing and development divisions). Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules. Used appropriately, software requirements specifications can help prevent software project failure. The software requirements specification document lists sufficient and necessary requirements for the project development. To derive the requirements, the developer needs to have a clear and thorough understanding of the products under development. This is achieved through detailed and continuous communications with the project team and customer throughout the software development process.

3.4.1 Functional Requirements

Implementation is the stages of a project when the theoretical design is turned into a working system. If the implementation stage is not properly planned and controlled, it can cause chaos. Thus, it can be considered to be the most crucial stage in achieving a successful new system and in giving the users confidence that the new system will work and be effective. Normally this stage involves setting up a co-ordination committee, which will act as a sounding board of ideas, complaints and problems. The first task is implementation planning ie; decision of the methods and time scale to be adopted. Apart from planning, the two major tasks of preparing for implementation are education and training of administrators and testing of the system. After the implementation phase is completed and the user staff adjusted to

the changes created by the candidate system, evaluation and maintenance is continued to bring the new system standards. The activities of the implementation phase can be summarized as;

- Implementation planning
- Education and training
- System training

System implementation is the final stage that put the utility into action. Implementation is the state in the project where the theoretical design turned into a working system.

3.4.2 Non-functional Requirements

In software engineering, a functional requirement defines a system or its components. It describes the functions that software must perform. Function is nothing but inputs, its behavior, and output. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Functional software requirement helps you to capture the intended behavior of the system. This behavior may be expressed as functions, services, or tasks which system is required to perform.

SL.NO	TYPES OF REQUIREMENTS	DETAILS
1	Performance	The response time should not vary with the increasing size of the data storage.
2	Security requirements	This application should

		not modify any pictures. The username and password should be a unique token id
3	Training requirements	Training has to be provided by the supervisor.

Performance

System performance is the most important quality in non-functional requirements and affects almost all the other preceding ones. Furthermore, reliability, availability, and maintainability (RAM) features fall exclusively under these requirements. System performance defines how fast a system can respond to a particular user's action under a certain workload.

Reliability

Reliability is the probability and percentage of the software performing without failure for a specific number of uses or amount of time

Availability

This feature defines the amount of time the system is running, the time it takes to repair a fault, and the time between lapses.

Security

Security measures ensure your software's safety against espionage or sabotage. These features are necessary even for stand-alone systems; you don't want anyone to have access to your sensitive data.

Maintainability

This feature indicates the average time and ease and rapidity with which a system can be restored after a failure

Portability

Portability in high-level computer programming is the usability of the same software in different environments. The Pre requirement for portability is the generalized abstraction between the application logic and system interfaces. When software with the same functionality is produced for several computing platforms, portability is the key issue for development cost reduction.

3.4.3 Hardware Requirements

The selection of hardware configuration is a very important task related to software development. The processor should be powerful to handle entire operations.

Processor : Intel® Core™ i3 or above

RAM : 4 GB RAM minimum , 8GB recommended

Hard Disk : 150 GB

Keyboard : Standard

Mouse : Normal

Monitor : Plug and play monitor

3.4.4 Software Requirements

The major element in building a system is the selection of compatible software.

The Software requirement of the system on which the project was developed is as follows:

Operating System : Microsoft windows 7 or above

Front End : Python

Back End : SQLite3

IDE : Visual Studio Code Web technology : Django Web Browser

:Google Chrome

3.4.5 Other Requirements

OPERATING SYSTEM

An operating system (OS) is system software that manages computer hardware and software resources and provides common services for computer programs. The operating system used in this project is windows OS

HTML (FRONT END)

The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. With HTML const PYTHON(BACKEND) ructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets.

CSS (FRONT END)

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. 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 which reduces complexity and repetition in the structural content as well as enabling the. CSS file to be cached to improve the page load speed between the pages that share the file and its formatting.

JAVASCRIPT (FRONT END)

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented Capabilities.

PYTHON(BACKEND)

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently whereas other languages use punctuation, and it has fewer syntactic constructions than other languages.Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain

Features of Python:

- Dynamic typed language
- High level language

- Free open source
- Interpreted programming language
- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support
- High-Level Language
- Extensible feature
- Portable language

In most programming languages, whenever a new version releases, it supports the features and syntax of the existing version of the language, therefore, it is easier for the projects to switch to the newer version. However, in the case of Python, the two versions Python 2 and Python 3 are very much different from each other.

Following are some of the applications of Python:

- Web and Internet Development
- Scientific and Numeric
- Education
- Desktop GUI
- Software Development
- Business Application

DJANGO(BACKEND)

Django is a Python-based web framework that allows you to quickly create efficient web applications. It is also called batteries included framework because Django provides built-in features for everything including Django Admin Interface, default database SQLite3, etc. When you're building a website, you always need a similar set of components: a way to handle

user authentication (signing up, signing in, signing out), a management panel for your website, forms, a way to upload files, etc. Django gives you ready-made components to use and that too for rapid development.

Features of Django:

- Excellent documentation and high scalability
- Used by Top MNCs and Companies, such as Instagram, Disqus, Spotify, youtube, Bitbucket, Dropbox, etc. and the list is never-ending.
- Easiest Framework to learn, rapid development and Batteries fully included.
- Python has huge library and features such as Web Scraping, Machine Learning, Image Processing, Scientific Computing, etc.
- One can integrate it all this with web application and do lots of advance stuff

SQLite3(BACKEND)

SQLite3 can be integrated with Python using the sqlite3 module, which was written by Gerhard Haring. It provides an SQL interface compliant with the DBAPI 2.0 specification described by PEP 249. You do not need to install this module separately because it is shipped by default along with Python version 2.5.x onwards. To use sqlite3 module, you must first create a connection object that represents the database and then optionally you can create a cursor object, which will help you in executing all the SQL statements.

Features of SQLite3 :

- totally free
- Server less
- flexible

- Configuration Not Required
- SQLite is a cross-platform DBMS
- Storing data is easy
- Variable length of columns
- Provide large number of API's

SQLite in general is a server-less database that you can use within almost all programming languages including Python. Server-less means there is no need to install a separate server to work with SQLite so you can connect directly with the database.

3.5 Feasibility Study

Feasibility study is a process that identifies, describes and evaluates proposed systems and selects the best system. During the study, the problem definition is solved and all aspects of the problem to be included in the system are determined. Size of project, cost of benefits is also estimated with greater accuracy. A good feasibility study will show the strength and defects before the project is planned or budgeted. It evaluates the project's potential success, because it rationally uncovers strengths and weaknesses of a proposed project.

3.5.1 Technical feasibility

Technical feasibility assesses the current resources (such as hardware and software) and technology, which are required to accomplish user requirements in the software within the allocated time and budget. For this, the software development team ascertains whether the current resources and technology can be upgraded or added in the software to accomplish specified user requirements. The technical feasibility is concerned with determining whether it is possible to do the project with the existing technologies. Technical needs of the system include Python as front-end and SQLite3 as backend. Also, the Django framework gives a suitable

environment for development on the local computer. All this software is freely available; hence the proposed system is technically feasible.

3.5.2 Economical feasibility

The purpose of the economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification and identification. It identifies the financial benefits and costs associated with the development of the system. Economic feasibility is often known as the cost benefit analysis. To carry out an economic feasibility study it is necessary to estimate actual money value against activities needed for implementing the system. While implementing our system we can ensure that the cost of prospective new ventures will ultimately be profitable to the people. So, we can say it is financially feasible. The technology used can be developed with the current equipment and has the technical capacity to hold data required by the old system.

- This technology supports the modern trends of technology.
- Easily accessible, more secure technologies

3.5.3 Operational feasibility

Operational feasibility is the measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirement analysis phase of system development. It focuses on the degree to which the proposed development project fits in with the existing environment and 22 Unification of Images Department of Computer Science, Sri C Achutha Menon Government College objectives it regards to development schedule. Operational feasibility focuses on human, organizational and political. The proposed system

can easily be implemented, as this is based on java coding, HTML. The database was created with Tomcat server which is more secure and easy to handle. The resources that are required to implement/install these are available. So the project is operationally feasible.

Chapter 4

Design

4.1 Introduction

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. System development is the process of creating or altering systems, along with the processes, practices, models, and methodologies used to develop them. The different interfaces of those components and the data that goes through that system. It is meant to satisfy specific needs and requirements of a business or organization through the engineering of a coherent and well-running system. Systems design implies a systematic approach to the design of a system. It may take a bottom-up or top-down approach, but either way the process is systematic wherein it takes into account all related variables of the system that needs to be created from the architecture, to the required hardware and software, right down to the data and how it travels and transforms throughout its travel through the system. Systems design then overlaps with systems analysis, systems engineering and systems architecture.

4.2 Modularity Criteria

The concept of modularity is used primarily to reduce complexity by breaking a system into varying degrees of independence and independence across and hide the complexity of each part behind an abstraction and interface. Effective modular design can be achieved if the partitioned modules are separately solvable, modifiable as well as compilable.

4.3 Architecture Diagrams/DFD

A data-flow diagram (DFD) is a way of representing the flow of data of a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented. There are several notations for displaying data flow diagrams. The notation presented above was described in 1979 by Tom DE Marco as part of Structured Analysis. For each data flow, at least one of the endpoints must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes. The data-flow diagram is part of the structured-analysis modeling tools. When using UML, the activity diagram typically takes over the role of the data flow diagram. A special form of data-flow plan is a site-oriented data-flow plan. Data-flow diagrams can be regarded as inverted Petri nets, because places in such networks correspond to the semantics of data memories. Analogously, the semantics of transitions from Petri nets and data flows and functions from data-flow diagrams should be considered equivalent. Terminologies used in DFD diagrams are:

Process

The process is part of a system that transforms inputs to outputs. The symbol of a process is a circle, an oval, a rectangle or a rectangle with rounded corners. The process is named in one word, a short sentence, or a phrase that is clearly to express its essence.

Data Store

Most information systems capture data for later use. The data is kept in a data store. It is represented by the open-ended box. At the left end is a small box used to number the data store and inside the main part of the rectangle is a meaningful label for the data store.

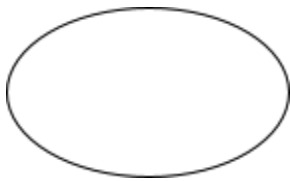
Data Flow

Data flow shows the transfer of information from one part of the system to another. The symbol of the flow is the arrow. The flow should have a name that determines what information (or what material) is being moved. Exceptions are flows where it is clear what information is transferred through the entities that are linked to these flows. Material shifts are modeled in systems that are not merely informative. Flow should only transmit one type of information (material). The arrow shows the flow direction. Flows link processes, warehouses and terminators.

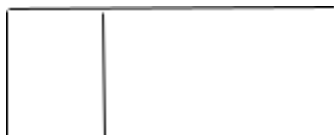
Terminator

The Terminator is an external entity that communicates with the system and stands outside of the system. It can be, for example, various organizations, groups of people (e.g., customers), authorities or a department (e.g., a human-resources department) of the same organization, which does not belong to the model system. The terminator may be another system with which the modeled system communicates.

Symbols used in DFD



Process

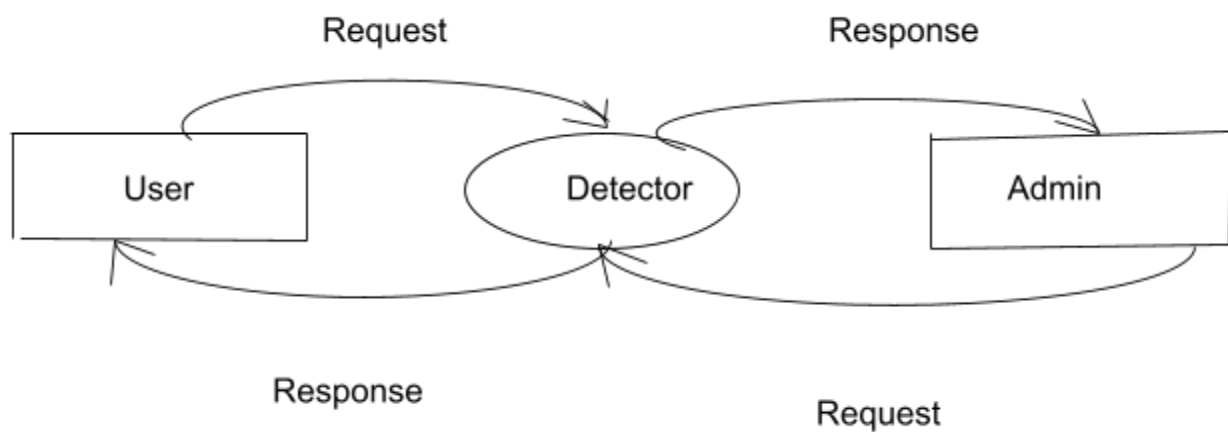


Data store



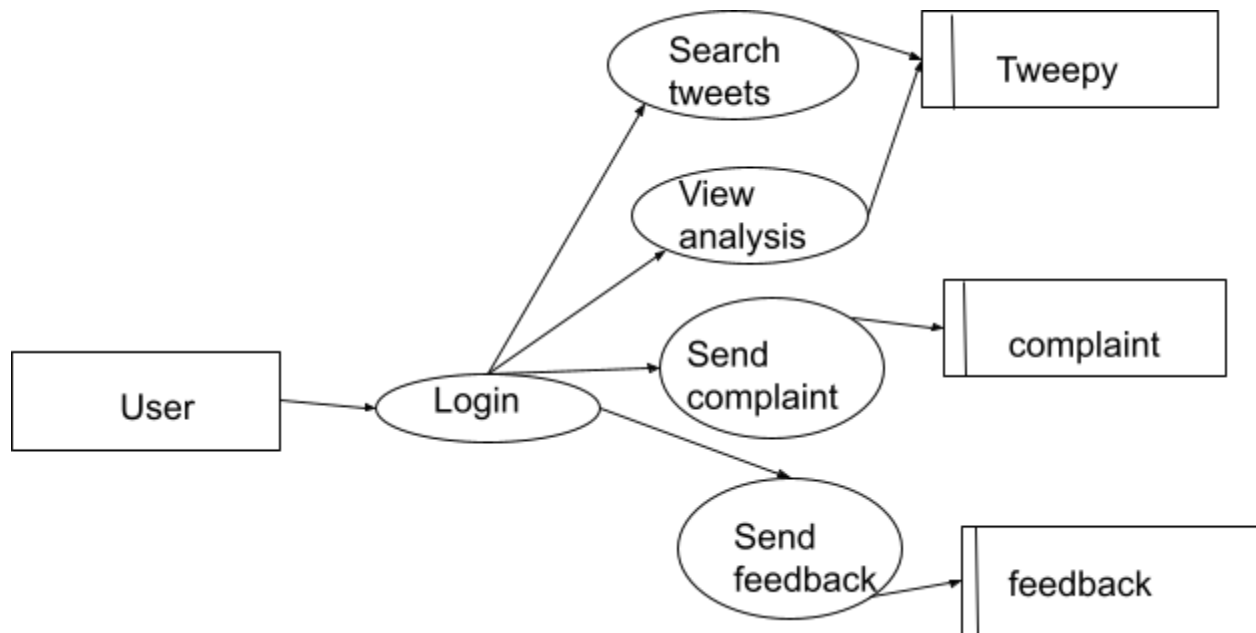
4.3.1 DFD Level 0

The Level 0 DFD represents the information flow through the system at the top most level.



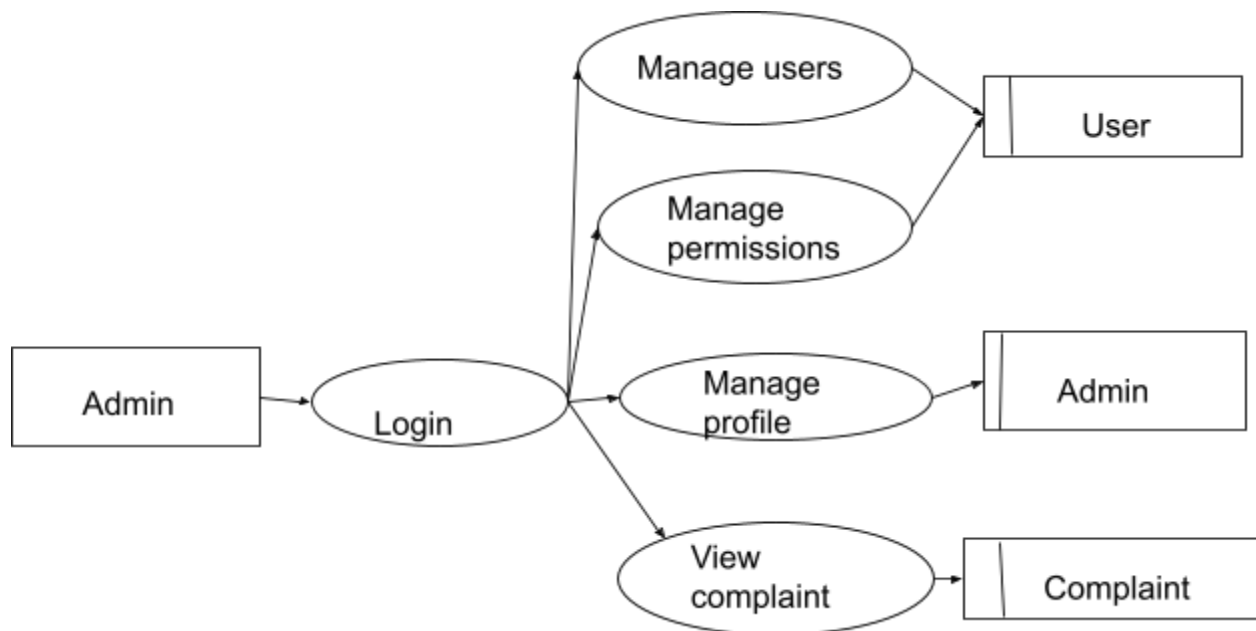
4.3.2 DFD Level 1

The Level 1 DFD defines four processes, two databases and two data entities(sources) and the data flow through the system in a more detailed structure.



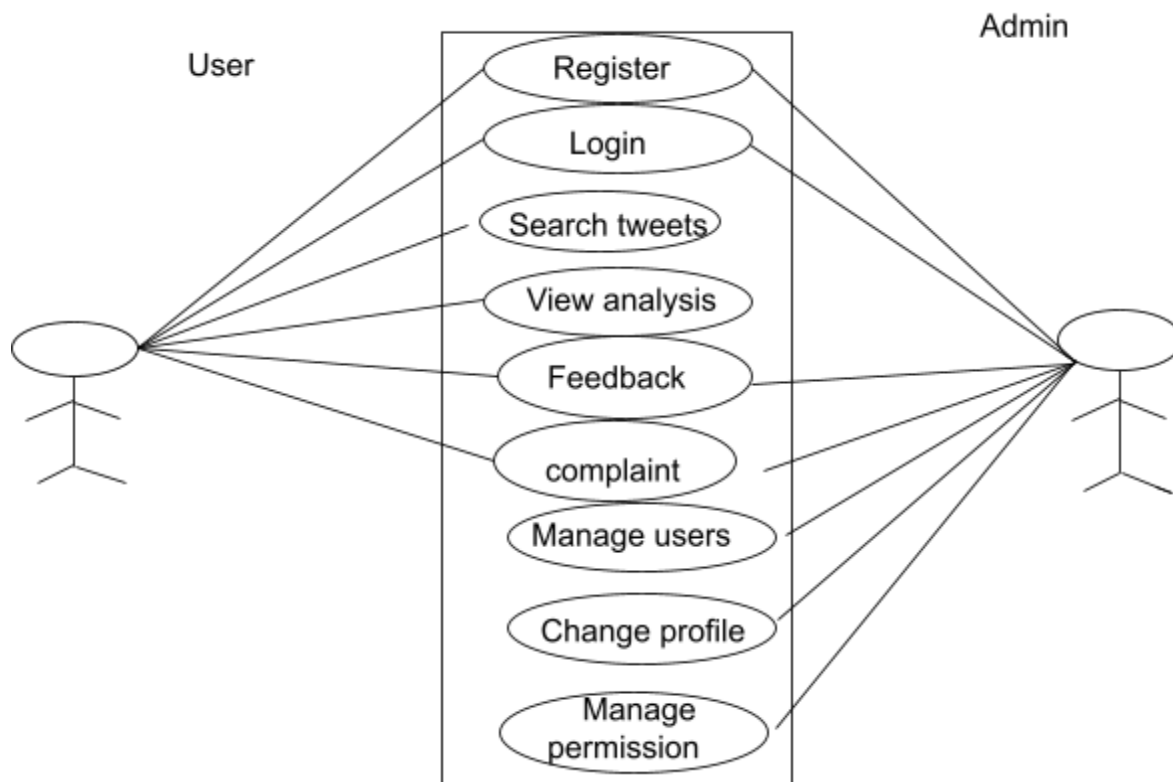
4.3.3 DFD Level 2

The DFD level 2 further describes the detailed flow of data through each module of the system



4.4 Use Case Diagrams

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.



4.5 User Interface Layout

User interface or UI describes the way in which a human interacts with a machine. It is the virtual part of a computer application or operating system through which a client interacts with a computer or software. It determines how commands are given to the system or the program and how data is displayed on the screen.

There are mainly two types of user interfaces. They are:

- Text based User Interface or Command Line Interface
- Graphical User Interface (GUI).

An effective User Interface design must have following principles:

- Structure
- Simplicity
- Visibility
- Feedback
- Flexible
- Tolerance

A layout defines the structure for a user interface in your application, such as in an activity. All elements in the layout are built using a hierarchy of view and view Group objects. A view usually draws something the user can see or interact with. Whereas a view group is an invisible container that defines the layout structure for view and other view group objects.

4.5.1 Register page

Registration

Username:

First name:

Last name:

Email:

Password:

Confirm password:

Phone number:

REGISTER

4.5.2 login page

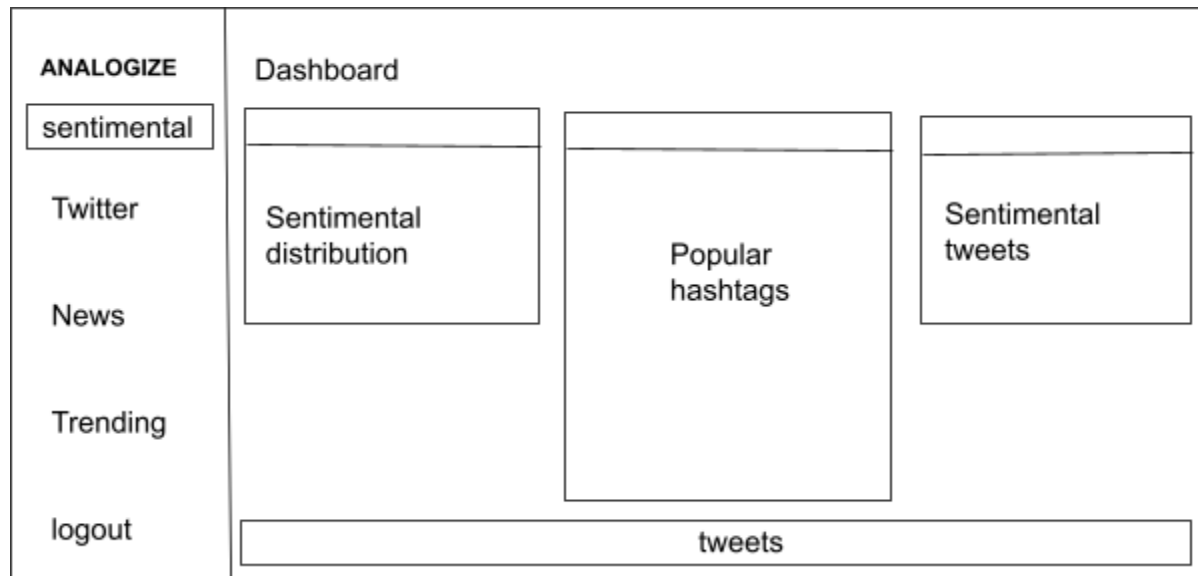
LOGIN

Username

password

LOGIN

4.5.3 dashboard



4.6 Structure of Reports Being Created

List all kinds of reports that you intend to produce. Identify each report with a unique title and list the fields that appear in each of the reports.

4.7 Database Design

A database is a data structure that stores organized information. Most databases contain multiple tables, which may each include several different fields. For example, a company database may include tables for products, employees, and financial records. Each of these tables would have different fields that are relevant to the information stored in the table. Nearly all e-commerce sites use databases to store product inventory and customer information. These sites use a database management system (or DBMS), such as Microsoft Access, FileMaker Pro, or MySQL as the "back end" to the website. By storing website data in a database, the

data can be easily searched, sorted, and updated. This flexibility is important for e-commerce sites and other types of dynamic websites. Early databases were relatively "flat," which means they were limited to simple rows and columns, like a spreadsheet. (See also "flat file database"). However, today's relational databases allow users to access, update, and search information based on the relationship of data stored in different tables. Relational databases can also run queries that involve multiple databases. While early databases could only store text or numeric data, modern databases also let users store other data types such as sound clips, pictures, and videos. The database design has several specific objectives:

- Control redundancy
- Ease of Learning and use
- Data independence
- More information at low cost
- Accuracy and integrity
- Recovery from failure
- Privacy and security
- Performance

Normalization

Normalization technique is the main tool used for the designing of the tables in a database. At the beginning we will have only one table with all the required fields as a unit. In further decompositions we will get different levels of normalized tables with minimum redundancy and interdependency. In the proposed system all the tables have normalized up to second normal form. The different normal forms applied during the database design are, first normal form and second normal form.

Second normal form (2NF) is a normal form used in database normalization.

Second normal form (2NF) is the second step in normalizing a database. A table that is in first normal form (1NF) must meet additional criteria if it is to qualify for

second normal form. Specifically, a table is in 2NF if and only if it is in 1NF and no non-prime attribute is dependent on any proper subset of any candidate key of the table. A non-prime attribute of a table is an attribute that is not a part of any candidate key of the table. All the tables in the proposed system ANALOGIZE have satisfied all these features So all the tables are in the second normal form.

EXCEPTION:

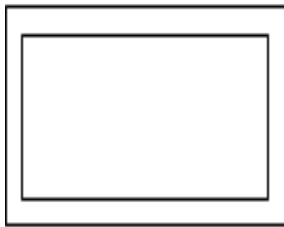
Adhering to the third normal form, while theoretically desirable, is not always practical. If you have a Customers table and you want to eliminate all possible interfiled dependencies, you must create separate tables for cities, ZIP codes, sales representatives, customer classes, and any other factor that may be duplicated in multiple records. In theory, normalization is worth pursuing. However, many small tables may degrade performance or exceed open file and memory capacities. It may be more feasible to apply third normal form only to data that changes frequently. If some dependent fields remain, design your application to require the user to verify all related fields when any one is changed.

4.7.1 List of Entities and Attributes

Entities are objects or concepts that represent important data. Entities are typically nouns such as product, customer, location or promotion. There are three types of entities commonly used in entity relationship diagrams.

**STRONG ENTITY**

These shapes are independent from other entities, and are called parent entities, since they will often have weak entities that depend on them. They will also have a primary key, distinguishing each occurrence of the entity.

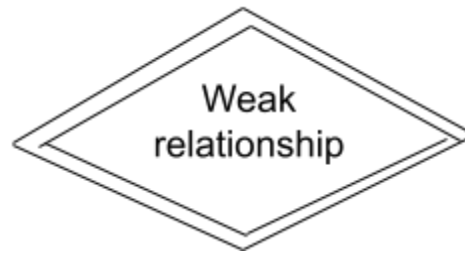
**WEAK ENTITY**

Weak entities depend on some other entity type. They don't have primary keys, and have no meaning in the diagram without their parent entity.


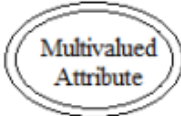
**ASSOCIATIVE KEY**

Associative entities relate the instance of several entity types. They also contain attributes specific to the relationship between those entity instances.

Relationships are meaningful associations between or among entities. They are usually verbs, example-assign, associate or track. A relationship provides usual information that couldn't be discerned with the just entity types. Weak relationships or identifying relationships are connections that exist between a weak entity type and its owner.



ERD attributes are characteristics of the entity that help users to better understand the database. Attributes are included to include details of the various entities that are highlighted in a conceptual ER diagram

Attribute Symbol	Name	Description
	Attribute	Attributes are characteristics of an entity, a many-to-many relationship, or a one to one relationship
	Multivalued attribute	Multivalued attributes are those that can take on more than one value.

4.7.2 E R Diagram

An entity-relationship diagram (ERD) is a data modeling 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.

The elements of an ERD are:

- Attributes
- Relationships

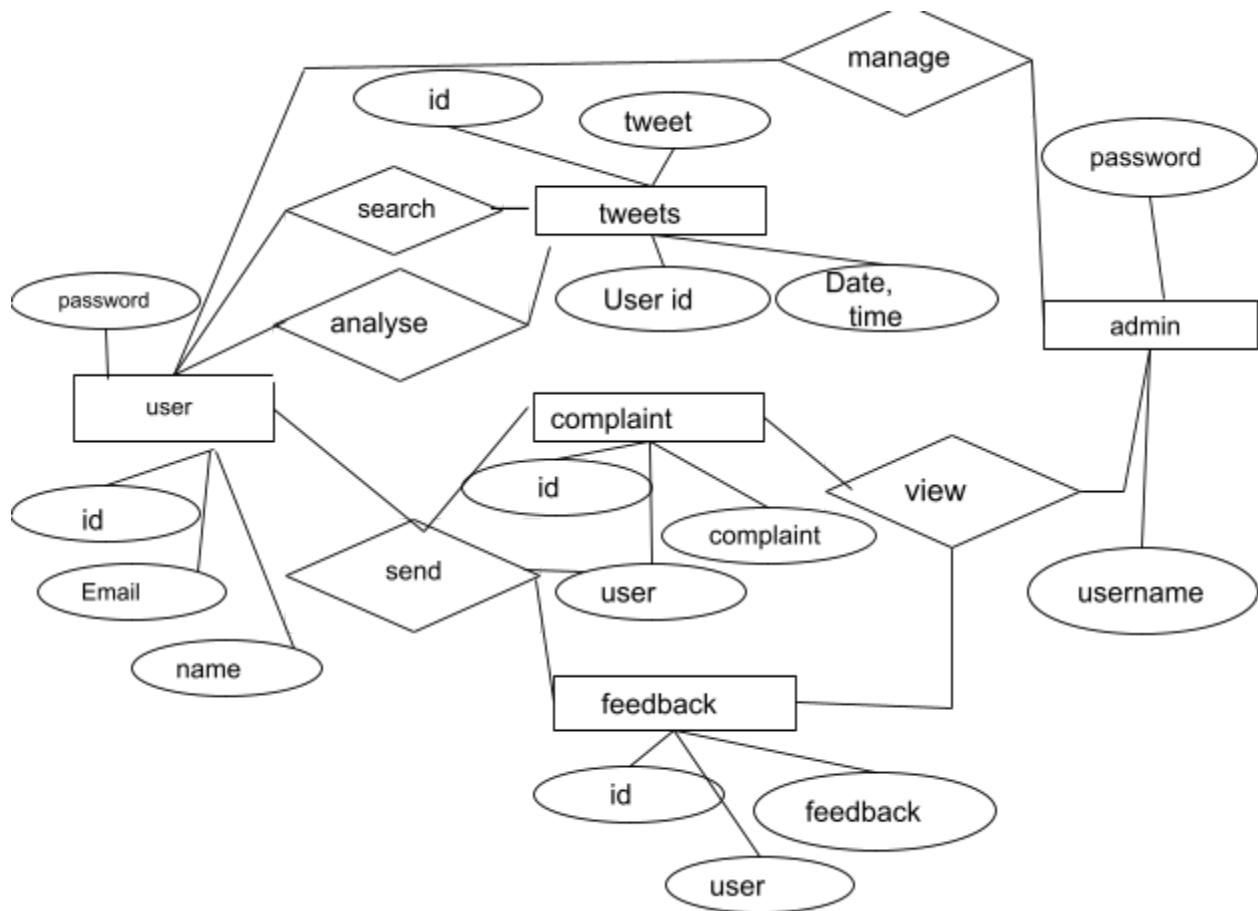
- Entities

Steps involved in creating an ERD include:

- Identifying and defining the entities
- Determining all interactions between the entities
- Analyzing the nature of interactions/determining the cardinality of the relationships.
- Creating the ERD

An entity-relationship diagram (ERD) is crucial to creating a good database design. It is used as a high-level logical data model, which is useful in developing a conceptual design for databases. An entity is a real-world item or concept that exists on its own. Entities are equivalent to database tables in a relational database, with each row of the table representing an instance of that entity. An attribute of an entity is a particular property that describes the entity. A relationship is the association that describes the interaction between entities. Cardinality, in the context of ERD, is the number of instances of one entity that can, or must, be associated with each instance of another entity. In general, there may be one-to-one, one-to-many, or many-to-many relationships. For example, let us consider two real-world entities, an employee and his department. An employee has attributes such as an employee number, name, department number, etc. Similarly, department number and name can be defined as attributes of a department. A department can interact with many employees, but an employee can belong to only one department, hence there can be a one-to-many relationship, defined between department and employee. In the actual database, the employee table will have department number as a foreign key, referencing from department table, to enforce the relationship.

The Entity Relationship diagram of Unification of Images is as follows:



4.7.3 Structure of Tables

Database design is the process of producing a detailed data model of a database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity. The term database design can be used to describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the database system(DBMS).

- Determine the relationships between the different data elements.
- Determine the data to be stored in the database.
- Superimpose a logical structure upon the data on the basis of these relationships

Table no:1

registrations details

field name	type	constraints	size
user_id	int	primary key	5
user_name	varchar	not null	50
password	varchar	not null	50
phone	integer	not null	50
location	varchar	not null	50

Table no:2**login details**

sl.no	field	type	constraints	description
1	user_id	int	foreign key	user id
2	username	varchar	not null	user name
3	password	varchar	not null	password

Table no:3**post details**

fieldname	type	constraints	size
p_id	int	primary key	5
name	varchar	not null	50
image	biob	not null	50
description	varchar	not null	50
out	int	not null	5
time	int	not null	5

Table no:4**complaint details**

field name	type	constraints	size
id	int	primary key	5
user	varchar	foreign key	50
p_id	int	foreign key	50
complaint	int	not null	5

Table no:5**feedback details**

field name	type	constraints	size
id	int	primary key	5
user	varchar	foreign key	50
p_id	int	foreign key	50
feedback	int	not null	50

Table no:6**admin details**

field name	type	constraints	size
username	varchar	not null	50
password	varchar	not null	50

Table no:7**anomaly post details**

field name	type	constraints	size
tweet	varchar	primary key	50
sentiment	varchar	not null	50
score	integer	not null	5

Chapter 5

Implementation

5.1 Introduction

Systems implementation is a set of procedures performed to complete the design contained in the approved systems design document and to test, install, and begin to use the new or revised information System. It is the fifth major step in the development of an Information System. In software development, an implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through computer programming and deployment.

5.2 Tools/Scripts for Implementation

Project planning tools include charts and graphs designed to track progress, repetition-based approaches to testing and adjusting everyday processes, and other actions that allow organizations to manage and improve important projects. Tools for project implementation are defined as the series of systems and methodologies designed to ensure teams are able to accomplish both short- and long-term projects. In our project, we used flow charts, Data Flow Diagrams, Entity-Relationship diagrams as tools for project implementation

5.3 Process Logic

Project logic provides the basis for planning and implementing monitoring and evaluation at project level. Project logic is defined as a conceptual framework of how a program or project is understood, or intended, to contribute to its specified outcomes. It focuses on outcomes rather than process. Process logic is a cause-and-effect explanation of a process. It expresses all the principal definitions and arguments that appear to be true for the process and its events, causes and circumstances.

5.4 Coding

view.py

```
from django.shortcuts import render, redirect
# from django.contrib.auth import login, authenticate
from django.http import HttpResponseRedirect
import requests
import json
import tweepy
from tweepy.auth import OAuthHandler
from textblob import TextBlob
from .fusioncharts import FusionCharts
from collections import OrderedDict
from .news import Analysis

# Create your views here.
def home(request):
    return render(request, 'blog/index.html', {})
```

```

def getSentiments(request):
    auth = OAuthHandler('XRCnQ49KVWgy0IsN5QYBTn5Zm',
        'P6UwYNbfboKQfrr51P3HLjp88Mq32SxNcQt7zsFKDdAZdXrAoW')
    auth.set_access_token('912853951984238592-BODZqgKvgD0QdKD5Rz1grMGP
        CDFiZm4', 'proz3qXFAR7Ie8YOylG86z0uERL8orw0HcClAA2X4CN6t')
    api = tweepy.API(auth)
    public_tweets = []
    if request.method=='GET':
        query = request.GET.get('search',"congress")
        a = Analysis(query)
        full_analysis = a.run()
    else:
        return redirect('home')

    chartData = OrderedDict()
    total_pos = 0
    total_neg = 0
    total_neu = 0

    for tweet in api.search_tweets(q=query, lang="en"):
        blob = TextBlob(tweet.text)
        senti = blob.sentiment.polarity
        sub = float("{0:.2f}".format(blob.sentiment.subjectivity))
        if senti>0.1:
            senti_analysis = 'Positive'
            total_pos += 1
        elif senti<-0.1:

```

```

        senti_analysis = 'Abusive'
        total_neg += 1
    else:
        senti_analysis = 'Neutral'
        total_neu += 1
    public_tweets.append({'tweet':tweet, 'subjectivity':sub, 'senti': senti,
'senti_analysis':senti_analysis})

    chartData[tweet.user.screen_name] = senti
    score=total_pos+(total_neu/2)-total_neg
    percent=(score/20)*100
    if(percent>95):
        percent=percent-10
    print(percent)
    dataSource = OrderedDict()

# The `chartConfig` dict contains key-value pairs data for chart attribute
chartConfig = OrderedDict()
chartConfig["caption"] = "Sentiments of Tweets"
chartConfig["subCaption"] = "sentiments in range 0 to 1. Red: Abusive, Blue:
Positive, Black: Neutral"
chartConfig["xAxisName"] = "Twitter Handles"
chartConfig["yAxisName"] = "Sentiments"
chartConfig["numberSuffix"] = ""
chartConfig["theme"] = "fusion"

# The `chartData` dict contains key-value pairs data
dataSource["chart"] = chartConfig
dataSource["data"] = []

```


Convert the data in the `chartData` array into a format that can be consumed by FusionCharts.

The data for the chart should be in an array wherein each element of the array is a JSON object

having the `label` and `value` as keys.

Iterate through the data in `chartData` and insert in to the `dataSource['data']` list.

for key, value in chartData.items():

 data = {}

 data["label"] = key

 if value>0.1:

 data["color"] = "#0000ff"

 elif value<-0.1:

 data["color"] = "#ff0000"

 else:

 data["color"] = "#000000"

 if value>0:

 data["value"] = value

 else:

 data["value"] = -value

 dataSource["data"].append(data)

datapie = [{

 "label": "Positive",

 "value": total_pos

}, {

```

        "label": "Abusive",
        "value": total_neg
    }, {
        "label": "Neutral",
        "value": total_neu
    }
]

# Create an object for the column 2D chart using the FusionCharts class
constructor

# The chart data is passed to the `dataSource` parameter.
column2D = FusionCharts("column2d", "ex1" , "100%" , "200", "chart-1",
"json", dataSource)

# Create an object for the pie3d chart using the FusionCharts class constructor
pie3d = FusionCharts("pie3d", "ex2" , "100%" , "200", "chart-2", "json",
# The data is passed as a string in the `dataSource` as parameter.
{
    "chart": {
        "caption": "Sentiment Distribution",
        "subCaption" : "Pie Chart",
        "showValues":"1",
        "showPercentInTooltip" : "1",
        "numberPrefix" : "$",
        "enableMultiSlicing":"1",
        "theme": "fusion"
    },
    "data": datapie,
})

```

```

# returning complete JavaScript and HTML code, which is used to generate
chart in the browsers.
# return render(request, 'index.html', {'output' : pie3d.render(), 'chartTitle': 'Pie
3D Chart'})

return render(request, 'search_result.html', {'public_tweets': public_tweets,
'full_analysis': full_analysis, 'percent':percent,'output1': column2D.render(),
'output2' : pie3d.render()})

```

form.py

```

from django import forms
from django.db.models import fields
from .models import*
from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth.models import User

class UserForm(UserCreationForm):
    email=forms.EmailField()
    class Meta:
        model=User
        fields=('username','first_name','last_name','email','password1','password2')
        labels=('password1','Password','password2','Confirm_Password')

class ProfileForm(forms.ModelForm):
    address=forms.Textarea()
    class Meta:
        model=Register
        fields=('address','phone_number')

```

admin.py

```
from django.contrib import admin
from django.contrib.admin.decorators import register
from .models import*
```

```
# Register your models here.
admin.site.register(Register)
```

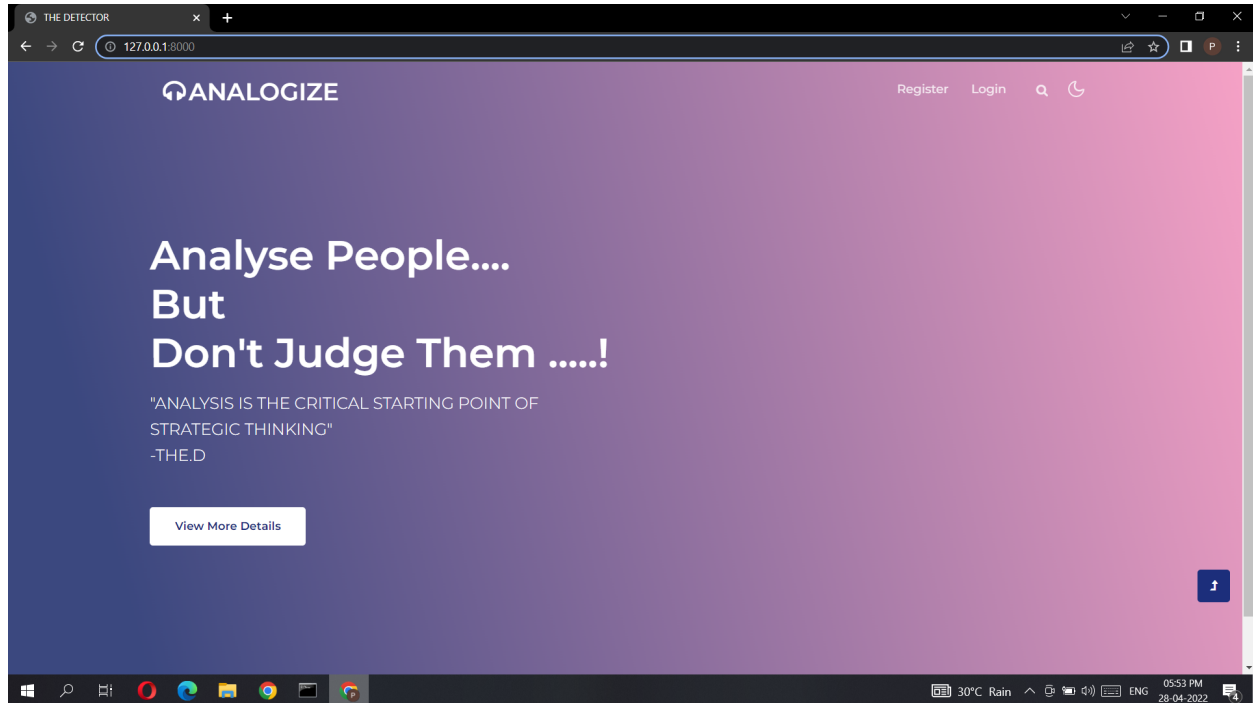
urls.py

```
from django.urls import path
from . import views
```

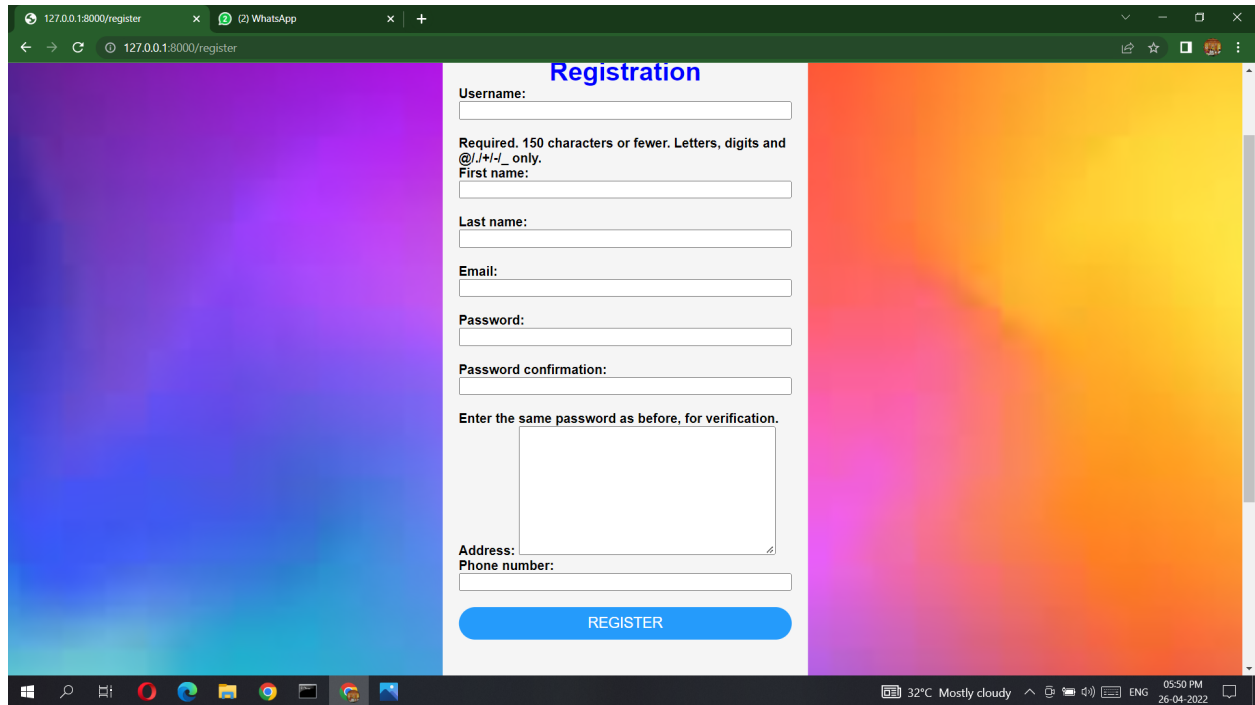
```
urlpatterns = [
    path("", views.index, name='index'),
    path('register', views.register, name='register'),
    path('dashboard', views.dashboard, name='dashboard'),
    path('user_login', views.user_login, name='user_login'),
    path('user_logout', views.user_logout, name='user_logout'),
]
```

5.5 Screen Shots

5.5.1 home page



5.5.2 registration page

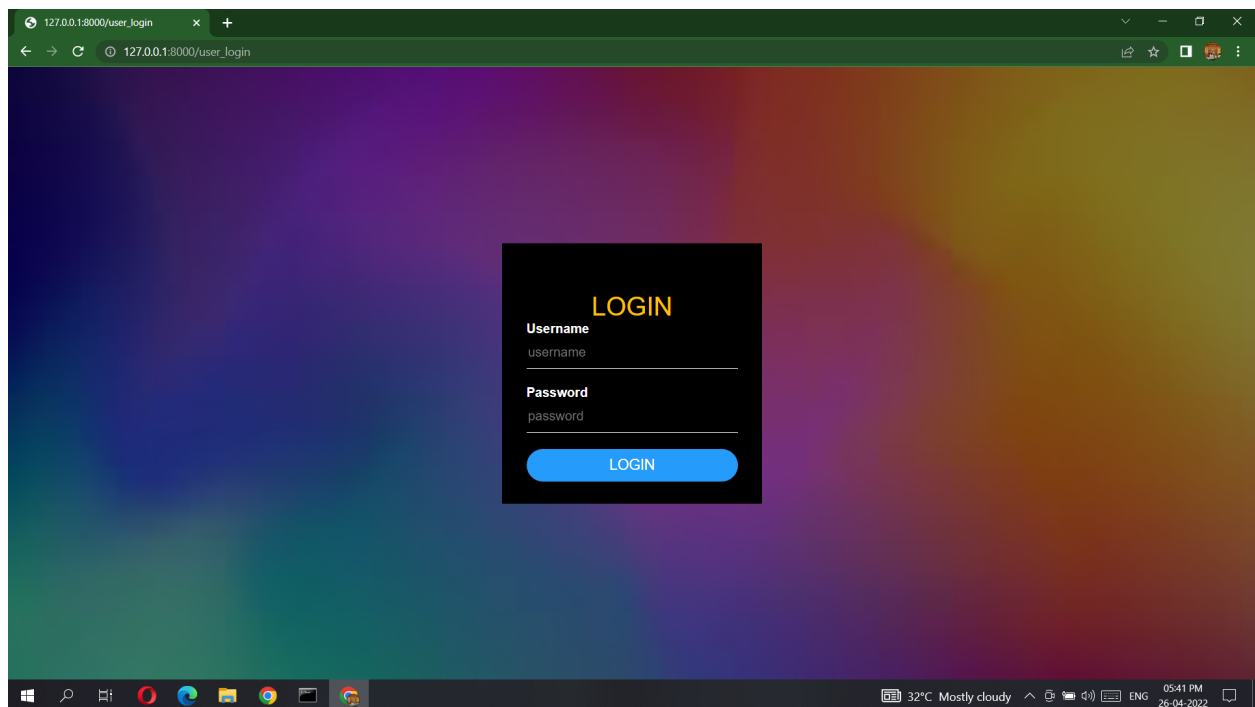


The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/register". The page has a vibrant, multi-colored background. The registration form is centered and contains the following fields and instructions:

- Registration** (Title)
- Username:**
- Required. 150 characters or fewer. Letters, digits and @/!+/-_ only.**
- First name:**
- Last name:**
- Email:**
- Password:**
- Password confirmation:**
- Enter the same password as before, for verification.**
- Address:**
- Phone number:**
- REGISTER** (Button)

The Windows taskbar at the bottom shows the system clock as 05:50 PM on 26-04-2022, with a weather forecast of 32°C Mostly cloudy.

5.5.3 login page

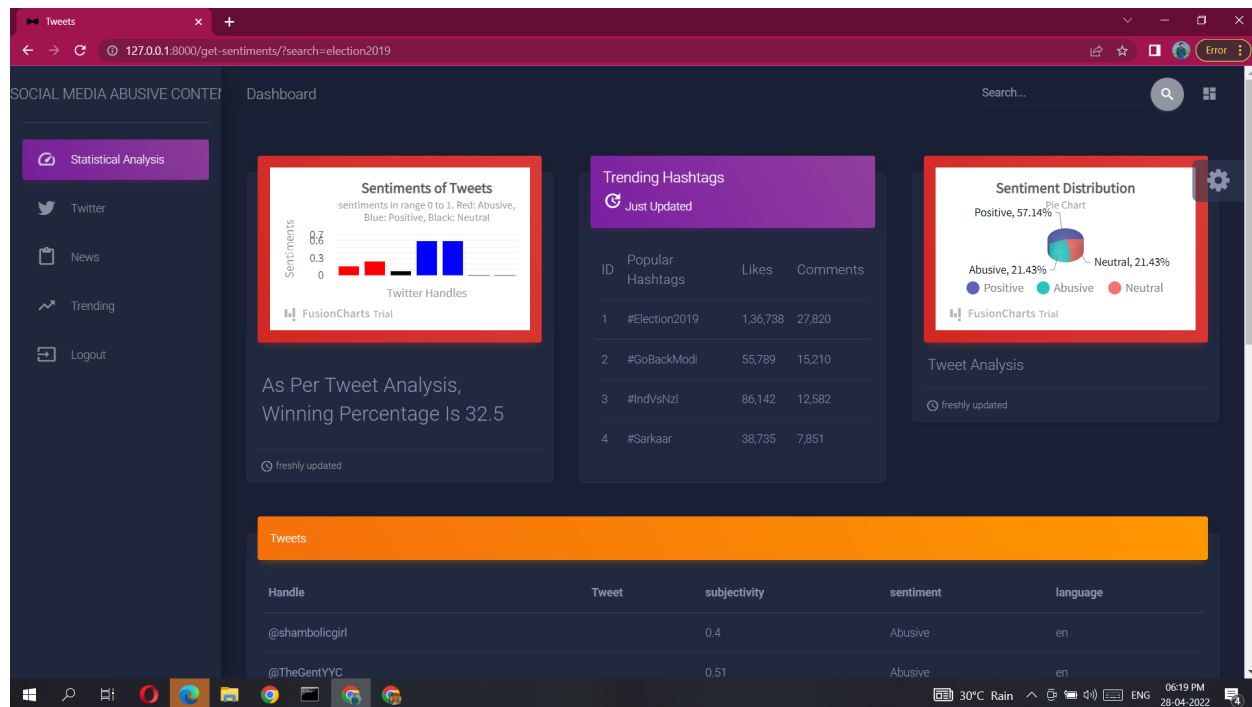


The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/user_login". The page has a dark, multi-colored background. The login form is centered and contains the following fields and instructions:

- LOGIN** (Title)
- Username**
- Password**
- LOGIN** (Button)

The Windows taskbar at the bottom shows the system clock as 05:41 PM on 26-04-2022, with a weather forecast of 32°C Mostly cloudy.

5.5.4 dashboard page



Chapter 6

Testing

6.1 Introduction

Testing is the process of checking whether the developed system is working according to the original objectives and requirements. Software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise, the project is not said to be complete. The system should be tested experimentally with test data so as to ensure that the system works according to the required specification. When the system is found working, test to with actual data and check performance. Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to. the process of executing a program or application with the intent of finding software bugs (error or other defects). It can be stated as the process of validating and verifying that a computer program/application/product.

- Meets the requirements that guided its design and development,
- Works as expected.
- Can be implemented with the same characteristics.
- Satisfies the needs of stakeholders

6.2 System testing

System testing is a critical aspect of Software Quality Assurance and represents the ultimate review of specification, design and coding. Testing is a process of executing a program with the intent of finding an error. A good test is one that has a probability of finding an as yet undiscovered error. The purpose of testing is to identify and correct bugs in the developed system. Nothing is complete without testing. Testing is vital to the success of the system. The entire testing process can be divided into different phases

- Integration testing
- Modularity testing

6.2.1 Modularity Testing

The first level of testing is unit testing. This test focuses on each module like user authentication, user option,tweet analysis,admin. This is also known as “Module Testing”. To check whether each module in the software is proper and it gives desired outputs to the given inputs. All validations and conditions are tested in the module in the first unit. The goal is to test the internal logic of the module. In unit testing each unit is tested during the programming stage itself, in the proposed system we tested the following.

- ☐ The insertion of values in the database.
- ☐ The updating of values into the database.
- ☐ The deletion of values from the database.
- ☐ Response of the system to inputs
- ☐ All the needed forms are checked for validation.
- ☐ The correctness of the phone number format (10-digit number).
- ☐ Whether the entries are in the correct format.

- Email validation (email address must contain at least @ sign and a dot(.). Characters or numbers are within the limit given (proper working of all fields,response of the system is delayed or not etc.).

6.2.2 Integration Testing

The modules are integrated to form a complete software package. It addresses the issues associated with the given problem of verification and program construction. Test that part of the system at some level works together correctly; the purpose of integration is to verify functional, performance and reliability requirements placed on major design items. This testing is conducted on the basis of modules. The integration testing is performed to detect design errors by focusing on testing the interconnection between modules. The objective is to take the unit tested modules combined and tested as a whole.

In the proposed system each single unit is integrated to form the system, that is modules such as user authentication,user option ,tweet analysis,admin are combined and tested as a whole, then tested the following.

- Checked if the response of the system is delayed or not after module integration.
- Linking between all the pages.
- Database connectivity
- Insertion, updating, deletion to the system and from the system are tested
- The modules like user authentication,user option,tweet analysis,admin are combined and integrated into a system and tested the whole system.
- Checked all other functionalities are working properly after integration.

6.3 Test Plan & Test Cases

The primary objectives of test case design methods are to drive a set of test that has the highest likelihood of uncovering the defects. To accomplish this objective, two categories of test case design techniques are used. Black box testing and White box testing.

6.3.1 Login Screen

Sl No	Test Case	Expected result	Observed result	Pass/Fail
1	Without entering username and password, press the login button.	It should prompt message "Invalid entry"	Message is prompted	Pass
2	Enter the correct username and password. Click the login button.	The application should be loaded.	Application loaded without error.	Pass

Chapter 7

Conclusion

The ANALOGIZE can be used for analyzing opinions in microblogs. It highlights the sentiments carried by the text which have been tweeted. The website will provide a complete analysis of a topic in social media. Sentiment analysis is a factor of AI. So that it can reduce the manual work.

All the knowledge I gained is fully applied in the mentioned system. All the suggestions forwarded in the software e proposal have been completed. This system is developed in such a way that the modules developed in the future can be linked easily to the system, without affecting the existing system, since it provides a hierarchical structure.

This system has been developed to satisfy the user's needs. The entire system is user friendly. The provided details by the user will be secured. The performance of the system is provided efficiently. The system was tested with all possible test data and was found to have an effective planning of the functions or processes with a high degree of accuracy and user friendliness.

Future enhancement

ANALOGIZE is a website used for extensive exploration of data stored on the web to identify and categorize the views expressed in a part of the text. It approaches for extracting and analyzing sentiments associated with the polarity of positive, negative or neutral. The system is developed in Python as front end, Django as Web Technology and SQLite3 as backend which makes the system more reliable and compatible with the other environments. The system proves better extensibility and flexibility for future enhancements. Any further requirements application is possible with the features guaranteed. The design of this software is in such a way that the addition of any new module if necessary is possible without affecting the integrity of the present system. In the future, we can implement facilities like

Some of future scopes that can be included in our work are:

- Use of parser can be embedded into system to improve results.
- A android-based application can be made for our work in future.
- We can improve our system so that can deal with sentences with multiple meanings.
- We can also increase the classification categories so that we can get better results.
- We can start work on multi languages like Hindi, Spanish, and Arabic to provide sentiment analysis to more locals.
- finding the target of the sentiment
- dealing with sarcasm
- ideology and it's handling

References

- Social Media as a New Trend Digital Journalism: A Surveillance
- Thuseethan and vasanthapriyan
- Ian Somerville, Software Engineering,
Pearson Education Asia,
6thEdition
- Awad, System Analysis and Design,
Galgotia,
2ndEdition.
- Elmasri, Navathe,
Fundamentals of Database Systems
Pearson Education Asia, 6th Edition
- Akрати Saxena, Harita Reddy, Pratishtha Saxena, *Principles of Social Networking*, vol. 246, pp. 249, 2022.
- <https://www.tutorialspoint.com/python>
- <https://www.stackify.com/learn-python-tutorials>
- <https://www.wiki.python.org/moin/PythonBooks>
- <https://www.realpython.com/best-python-book>