A MINI PROJECT REPORT

on

**Text-Emotion Recognition**

**Submitted by**

**Vijay Kumar Sharma(161500615)**

**Vinay Singh Parmar (161500620)**

**Ravi Kumar Mishra (161500442)**

**To**

**Ms. Shalini Agrawal**

Department of Computer Engineering & Applications

**Institute of Engineering & Technology**



**GLA University**

**Mathura- 281406, INDIA**

**December, 2018**

**Table of Contents**

|  |  |
| --- | --- |
| Declaration | **III** |
| Certificate | **IV** |
| Acknowledgement | **V** |
| Abstract | **VI** |
| 1. **Introduction** | **7** |
| 1.1 Purpose | 7 |
| 1.2 Scope | 7 |
| 1. **Software Requirement Analysis** | **8** |
| 2.1 Define the problem | 9 |
| 2.2 Define the modules and their functionalities (SRS) | 10 |
| 1. **Software Design** | **11** |
| 3.1 Data Flow Diagram | 11 |
| 3.2 Use-Case Diagram | 12 |
| 1. **Testing** | **13** |
| 4.1 White Box Testing | 13 |
| 4.2 Black box Testing | 13 |
| 1. **Implementation and User Interface** | **14** |
| * 1. HTML | 15 |
| * 1. CSS | 16 |
| 1. **References/Bibliography** | **18** |
| 1. **Appendices** | **19** |

**Department of Computer Engineering and Applications**

**GLA University, Mathura**

**17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,**

**Mathura – 281406**



**Declaration**

We hereby declare that the work which is being presented in the Mini Project “**Text-Emotion Recognition”,** in partial fulfillment of the requirements for Mini-Project LAB, is an authentic record of our own work carried under the supervision of **Ms.Shalini Agrawal, Asst. Professor, GLA University, Mathura**.

**Vinay Singh Parmar**

**Ravi Kumar Mishra**

**Vijay Kumar Sharma**



**Department of Computer Engineering and Applications**

**GLA University, Mathura**

**17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,**

**Mathura – 281406**

**CERTIFICATE**

This is to certify that the project entitled **“Text-Emotion Recognition ”** carried out in Mini Project – II Lab is a bonafide work done by **Vijay Kumar Sharma (161500615), Vinay Singh Parmar(161500620) and Ravi Kumar Mishra (161500442)** and is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology (Computer Science & Engineering).

**Signature of Supervisor:**

**Name of Supervisor:**

**Date:**

**ACKNOWLEDGEMENT**

It gives us a great sense of pleasure to present the report of the B. Tech Mini Project undertaken during B. Tech. Third Year. This project in itself is an acknowledgement to the inspiration, drive and technical assistance contributed to it by many individuals. This project would never have seen the light of the day without the help and guidance that we have received.

Our heartiest thanks to **Dr. (Prof). Anand Singh Jalal,** Head of Dept., Department of CEA for providing us with an encouraging platform to develop this project, which thus helped us in shaping our abilities towards a constructive goal.

We owe special debt of gratitude to **Ms. Shalini Agrawal,** Assistant Professor Department of CEA, for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. He has showered us with all his extensively experienced ideas and insightful comments at virtually all stages of the project & has also taught us about the latest industry-oriented technologies.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Vijay Kumar Sharma

Vinay Singh Parmar

Ravi Kumar Mishra

**Abstract**

Sentiment analysis deals with identifying and classifying opinions or sentiments expressed in source text. Social media is generating a vast amount of sentiment rich data in the form of tweets, status updates, blog posts etc. Sentiment analysis of this user generated data is very useful in knowing the opinion of the crowd. Twitter sentiment analysis is difficult compared to general sentiment analysis due to the presence of slang words and misspellings. The maximum limit of characters that are allowed is 140. Knowledge base approach and Machine learning approach are the two strategies used for analyzing sentiments from the text. In this paper, we try to analyze the twitter posts about electronic products like mobiles, laptops etc using Machine Learning approach. By doing sentiment analysis in a specific domain, it is possible to identify the effect of domain information in sentiment classification. We present a new feature vector for classifying the tweets as positive, negative and extract peoples' opinion about products.

**Chapter 1: Introduction**

Emotions in Social Psychology, in which it explained the emotion system and formally classified the human emotions through an emotion hierarchy in six classes at primary level which are Love, Joy, Anger, Sadness, Fear and Surprise. Certain other words also fall in secondary and tertiary levels. People are able to perfectly distinguish the expressed emotions because they understand the meaning of the words and phrases. They also are able to generate expressions and sentences for different emotions.

**1.1 Purpose**

The purpose of this Project document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements.

**1.2 Scope**

(1) Emotions Recognition System

(2) Our software will be able to recognize different types of emotions like Happy, Sad, Angry and Neutral.

(3) Applications

(a) Health care-patient feelings about treatment.

(b) It can be used in call centers.

**1.3 Overview**

Emotion Recognition from text is a recent field of research that is related to sentimental analysis. Emotion analysis aims to detect and recognize types of feeling through the Expression of text. Such as anger, happy, sad, and neutral.

### Chapter 2: Software Requirement Analysis

### 2.1 Why emotion detection?

Emotion Detection and Recognition from text is a recent field of research that is closely related to Sentiment Analysis. Sentiment Analysis aims to detect positive, neutral, or negative feelings from text, whereas Emotion Analysis aims to detect and recognize types of feelings through the expression of texts, such as anger, fear, happiness, sadness*,* and surprise. Emotion detection may have useful applications, such as:

* Gauging how happy our citizens are. Different indexes have different definitions; most evolve around economic, environmental, health, and social factors. Since the mid-2000s, Government and organizations around the world are paying increasing attention to the happiness index.
  + This metric is defined as the overall index scores that rank countries based on their efficiency, as well as how many long and happy lives each country produces per unit of environmental output. This is unusual because the majority of indexes are based upon economic measures.
  + Societal Wellbeing metrics. The UK government measures people’s wellbeing; their statistics can be found here. Other countries and cities such as Seattle, Dubai, and South Korea, have similar measures.
* Pervasive computing, to serve the individual better. This may include suggesting help when anxiety is detected through speech, or to check the tone of an email before sending it out.
* Understanding the consumer. Improving perception of a customer with the ultimate goal to increase brand reputation and sales.

**2.2 Modules and their functionalities:**

**2.2.1 Numpy:**

Numpy , short for Numerical Python, has long been a cornerstone of numerical com‐

puting in Python. It provides the data structures, algorithms, and library glue needed

for most scientific applications involving numerical data in Python. NumPy contains,

among other things:

•A fast and efficient multidimensional array object n-d array.

•Functions for performing element-wise computations with arrays or mathematical operations between arrays

•Tools for reading and writing array-based data-sets to disk

•Linear algebra operations, Fourier transform, and random number generation

Beyond the fast array-processing capabilities that NumPy adds to Python, one of its

primary uses in data analysis is as a container for data to be passed between algorithms and libraries. For numerical data, NumPy arrays are more efficient for storing and manipulating data than the other built-in Python data structures. Also, libraries written in a lower-level language, such as C or Fortran, can operate on the data stored in a NumPy array without copying data into some other memory representation.

Thus, many numerical computing tools for Python either assume NumPy arrays as a

primary data structure or else target seamless interoperability with NumPy.

**Pandas:**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word

Panel Data – An Econometrics from Multidimensional data. In 2008, developer WesMcKinney started developing pandas when in need of high performance, flexible tool for analysis of data.Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.Python with Pandas is used in a wide

range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

**Key Features of Pandas:**

• Fast and efficient Data Frame object with default and customized indexing.

• Tools for loading data into in-memory data objects from different file formats.

• Data alignment and integrated handling of missing data.

• Reshaping and pivoting of date sets.

• Label-based slicing, indexing and subsetting of large data sets.

• Columns from a data structure can be deleted or inserted.

• Group by data for aggregation and transformations.

• High performance merging and joining of data.

• Time Series functionality.

**Scikit-learn:**

Since the project’s inception in 2010, scikit-learn has become the premier general-

purpose machine learning toolkit for Python programmers. In just seven years, it has

had over 1,500 contributors from around the world. It includes submodules for such

models as:

• Classification: SVM, nearest neighbors, random forest, logistic regression, etc.

• Regression: Lasso, ridge regression, etc.

• Clustering: k-means, spectral clustering, etc.

• Dimensionality reduction: PCA, feature selection, matrix factorization, etc.

• Model selection: Grid search, cross-validation, metrics.

• Preprocessing: Feature extraction, normalization.

Along with pandas, statsmodels, and IPython, scikit-learn has been critical for enabling Python to be a productive data science programming language.

**Natural Language ToolKit** (**NLTK**):-

**Natural Language processing** is about developing applications and services that are able to understand human languages. a comprehensive Python library for natural language processing and text analytics. Originally designed for teaching, it has been adopted in the industry for research and development due to its usefulness and breadth of coverage. NLTK is often used for rapid prototyping of text processing programs and can even be used in production applications.

Key features of NLTK:-

1.Tokenize Text Using Pure Python

2.Count Word Frequency

3.Remove Stop Words Using NLTK

**Joblib:-**

Joblib is a set of tools to provide lightweight pipelining in Python. In particular:

1. Transparent disk-caching of functions and lazy re-evaluation (memoize pattern)

2. Easy simple parallel computing

Joblib is optimized to be fast and robust in particular on large data and has specific optimizations for numPy arrays. It is BSD-licensed.

Features of joblib:-

1. Transparent and fast disk-caching of output value
2. Embarrassingly parallel helper
3. Fast compressed Persistence.

**Chapter 3:Software Design**

**3.1 Data Flow Diagram:-**

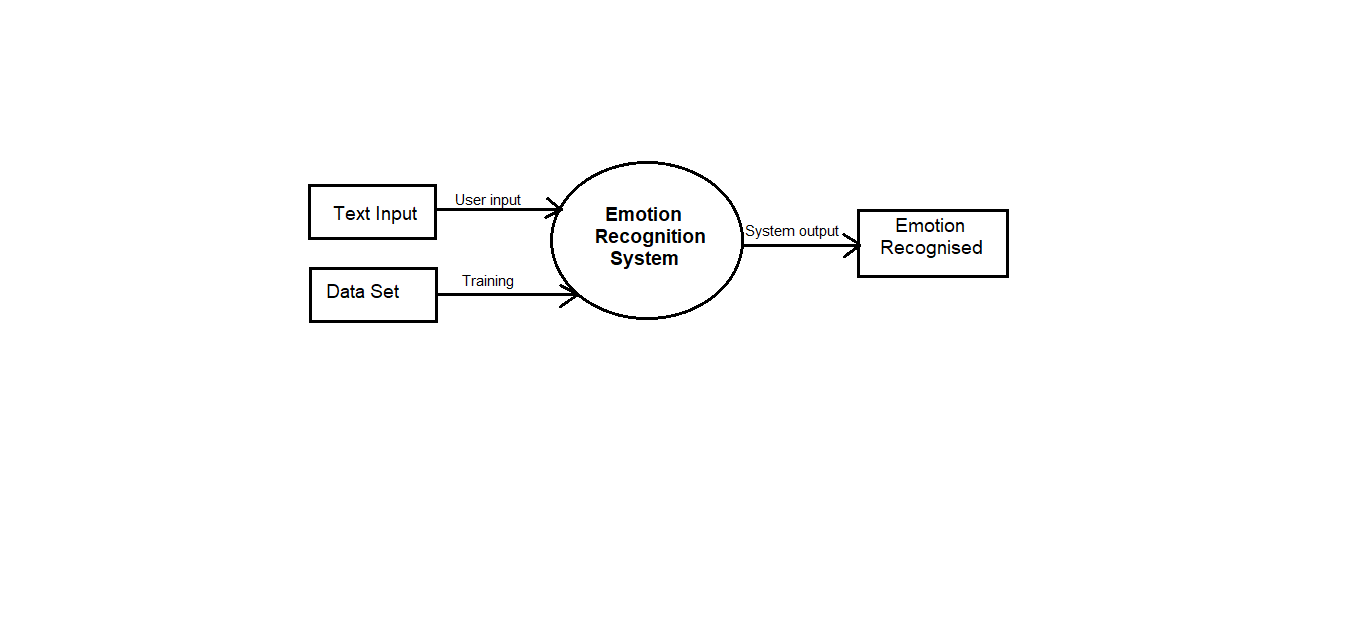


Fig. 3.1 Data flow diagram for Emotion Recognition

**3.2 Use Case Diagram:-**

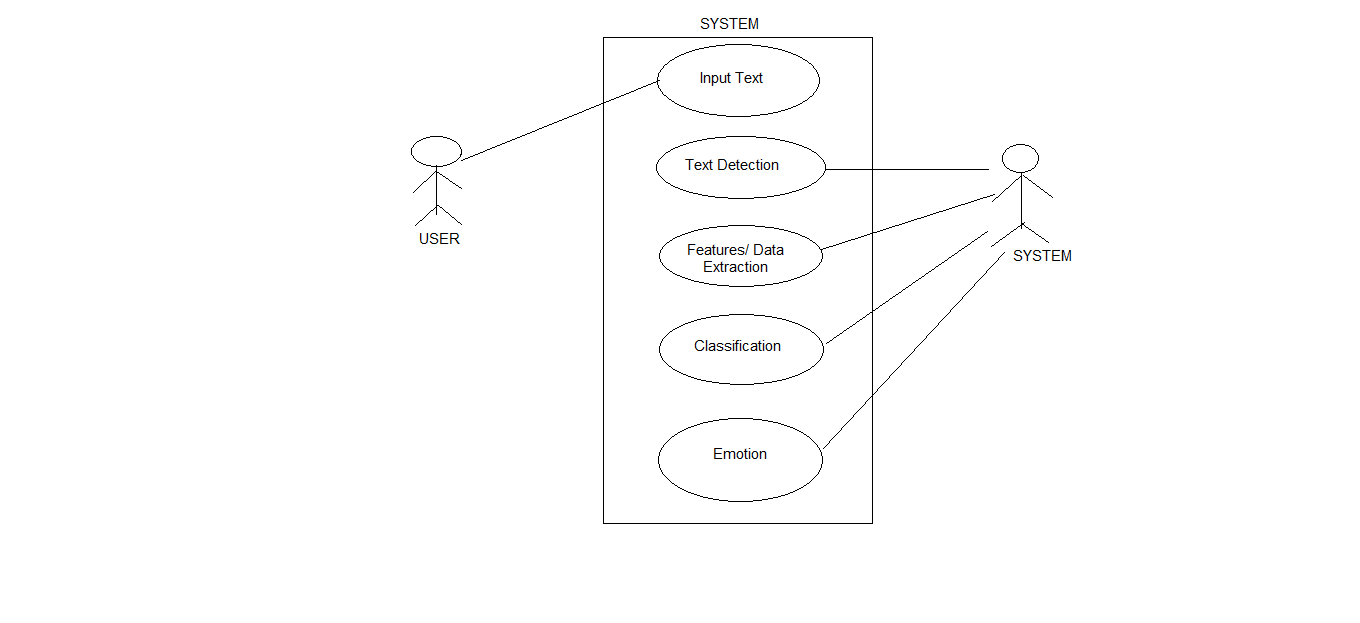
****

Fig. 3.2 Use case diagram for Emotion Rocognitition

**Chapter 4:Testing**

### 4.1 Introduction:

### Black box test cases:

Test cases Input text Expected output

1. Today I am very happy. Happy

2. I will beat you. Angry

3. He is in problem. Worry

**White box test cases:**

Test Cases Input text Expected output

1. Hello! I am there . Neutral

2. This is very sad moment for me. Sad

3. This is awesome. Happy

### Chapter 5: Implementation and User Interface

We have used HTML, CSS, Flask to make this system user friendly.

**5.1 HTML**

Hypertext Markup Language (HTML) is the standard [markup language](https://en.wikipedia.org/wiki/Markup_language" \o "Markup language) for creating [web pages](https://en.wikipedia.org/wiki/Web_page" \o "Web page) and [web applications](https://en.wikipedia.org/wiki/Web_application" \o "Web application). With [Cascading Style Sheets](https://en.wikipedia.org/wiki/Cascading_Style_Sheets" \o "Cascading Style Sheets) (CSS) and [JavaScript](https://en.wikipedia.org/wiki/JavaScript" \o "JavaScript), it forms a triad of [cornerstone](https://en.wikipedia.org/wiki/Cornerstone" \o "Cornerstone) technologies for the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web" \o "World Wide Web).

[Web browsers](https://en.wikipedia.org/wiki/Web_browser" \o "Web browser) receive HTML documents from a [web server](https://en.wikipedia.org/wiki/Web_server" \o "Web server) or from local storage and [render](https://en.wikipedia.org/wiki/Browser_engine" \o "Browser engine) the documents into multimedia web pages. HTML describes the structure of a web page [semantically](https://en.wikipedia.org/wiki/Semantic_Web" \o "Semantic Web) and originally included cues for the appearance of the document.

[HTML elements](https://en.wikipedia.org/wiki/HTML_element" \o "HTML element) are the building blocks of HTML pages. With HTML constructs, [images](https://en.wikipedia.org/wiki/HTML_element" \l "Images_and_objects" \o "HTML element) and other objects such as [interactive forms](https://en.wikipedia.org/wiki/Fieldset" \o "Fieldset) may be embedded into the rendered page. HTML provides a means to create [structured documents](https://en.wikipedia.org/wiki/Structured_document" \o "Structured document) by denoting structural [semantics](https://en.wikipedia.org/wiki/Semantics" \o "Semantics) for text such as headings, paragraphs, lists, [links](https://en.wikipedia.org/wiki/Hyperlink" \o "Hyperlink), quotes and other items. HTML elements are delineated by tags, written using [angle brackets](https://en.wikipedia.org/wiki/Bracket" \l "Angle_brackets" \o "Bracket). Tags such as <img /> and <input /> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a [scripting language](https://en.wikipedia.org/wiki/Scripting_language" \o "Scripting language) such as [JavaScript](https://en.wikipedia.org/wiki/JavaScript" \o "JavaScript), which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The [World Wide Web Consortium](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium" \o "World Wide Web Consortium) (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

**5.2 CSS**

Cascading Style Sheets (CSS) is a [style sheet language](https://en.wikipedia.org/wiki/Style_sheet_language" \o "Style sheet language) used for describing the [presentation](https://en.wikipedia.org/wiki/Presentation_semantics" \o "Presentation semantics) of a document written in a [markup language](https://en.wikipedia.org/wiki/Markup_language" \o "Markup language) like [HTML](https://en.wikipedia.org/wiki/HTML" \o "HTML)

CSS is a cornerstone technology of the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web" \o "World Wide Web), alongside HTML and [JavaScript](https://en.wikipedia.org/wiki/JavaScript" \o "JavaScript).

CSS is designed to enable the separation of presentation and content, including [layout](https://en.wikipedia.org/wiki/Page_layout" \o "Page layout), [colors](https://en.wikipedia.org/wiki/Color" \o "Color), and [fonts](https://en.wikipedia.org/wiki/Typeface" \o "Typeface).

This separation can improve content [accessibility](https://en.wikipedia.org/wiki/Accessibility" \o "Accessibility), provide more flexibility and control in the specification of presentation characteristics, enable multiple [web pages](https://en.wikipedia.org/wiki/Web_page" \o "Web page) to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or [screen reader](https://en.wikipedia.org/wiki/Screen_reader" \o "Screen reader)), and on [Braille-based](https://en.wikipedia.org/wiki/Braille_display" \o "Braille display) tactile devices. CSS also has rules for alternate formatting if the content is accessed on a [mobile device](https://en.wikipedia.org/wiki/Mobile_device" \o "Mobile device).

The name *cascading* comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

The CSS specifications are maintained by the [World Wide Web Consortium](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium" \o "World Wide Web Consortium) (W3C). Internet media type ([MIME type](https://en.wikipedia.org/wiki/MIME_media_type" \o "MIME media type)) text/css is registered for use with CSS by [RFC 2318](https://tools.ietf.org/html/rfc2318) (March 1998). The W3C operates a free [CSS validation service](https://en.wikipedia.org/wiki/W3C_Markup_Validation_Service" \l "CSS_validation" \o "W3C Markup Validation Service) for CSS documents.

In addition to HTML, other markup languages support the use of CSS including [XHTML](https://en.wikipedia.org/wiki/XHTML" \o "XHTML), [plain XML](https://en.wikipedia.org/wiki/Plain_Old_XML" \o "Plain Old XML), [SVG](https://en.wikipedia.org/wiki/Scalable_Vector_Graphics" \o "Scalable Vector Graphics), and [XUL](https://en.wikipedia.org/wiki/XUL" \o "XUL).

**5.3 Flask**

Flask is a micro web framework written in [Python](https://en.wikipedia.org/wiki/Python_(programming_language)" \o "Python (programming language)). It is classified as a [microframework](https://en.wikipedia.org/wiki/Microframework" \o "Microframework) because it does not require particular tools or libraries.

It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more regularly than the core Flask program.

Flask is commonly used with MongoDB, which gives it more control over databases and history.

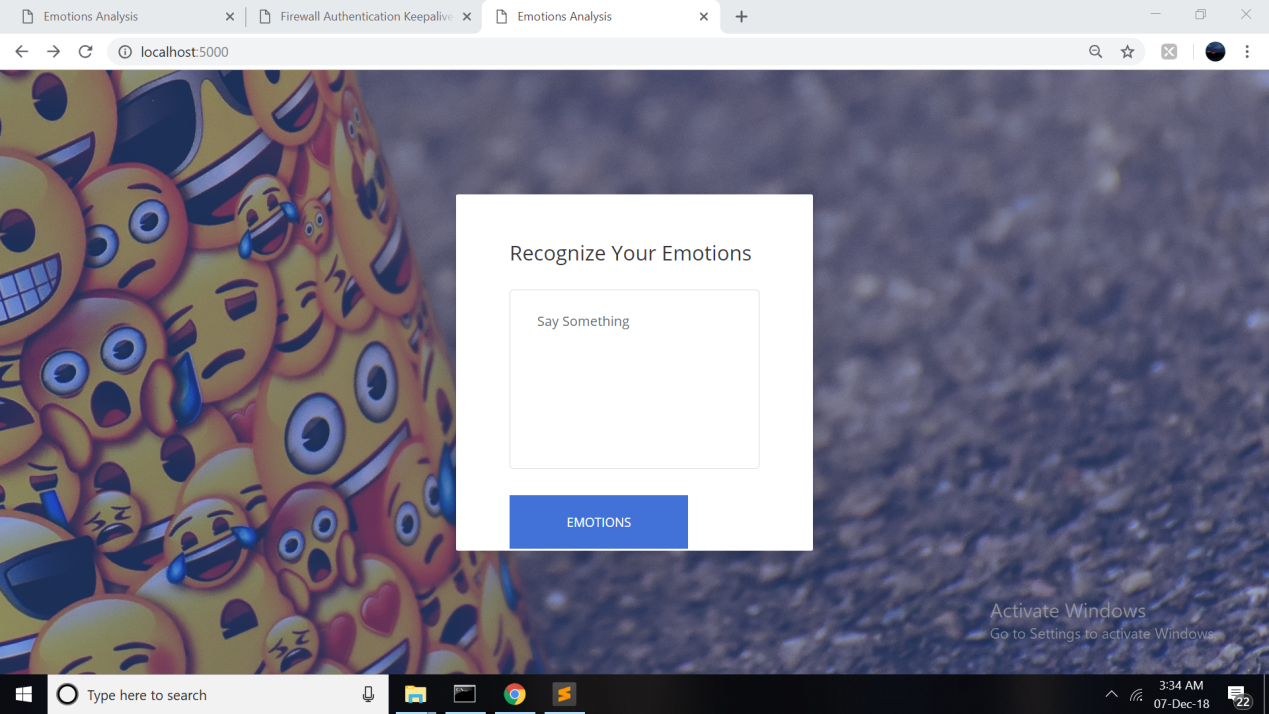
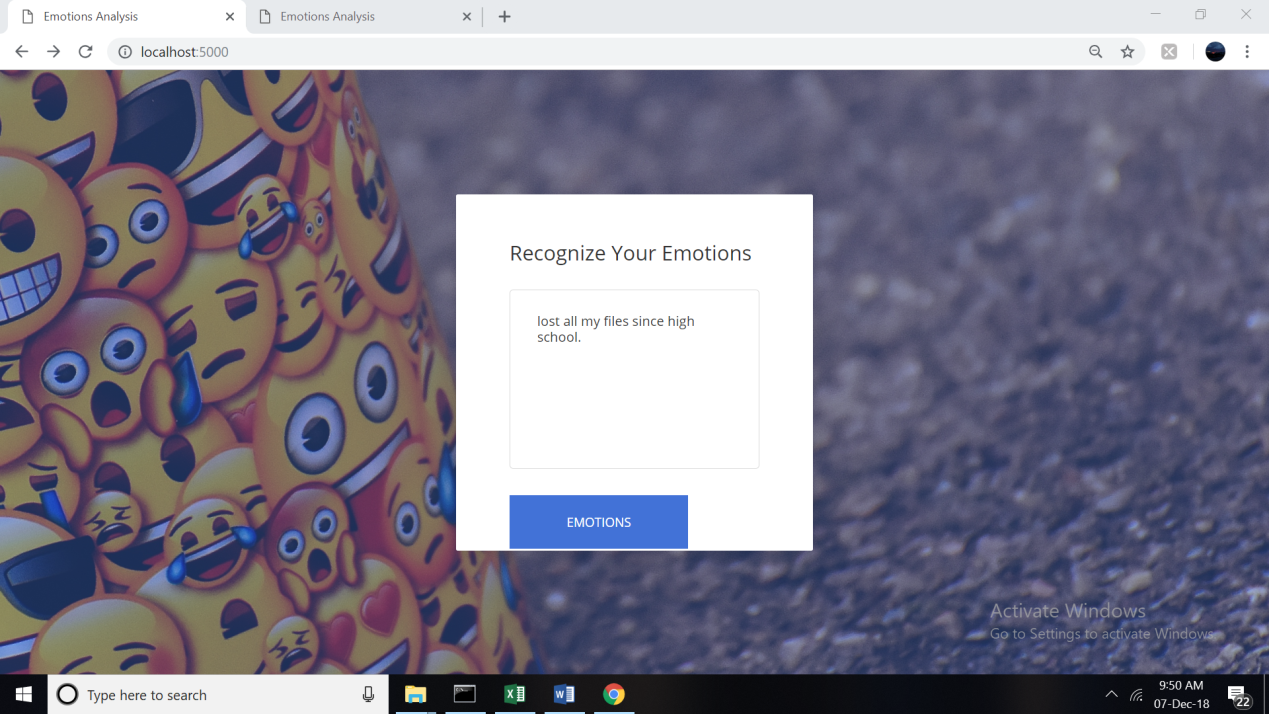


Fig. 5.1Layout for Emotion Recognition System.



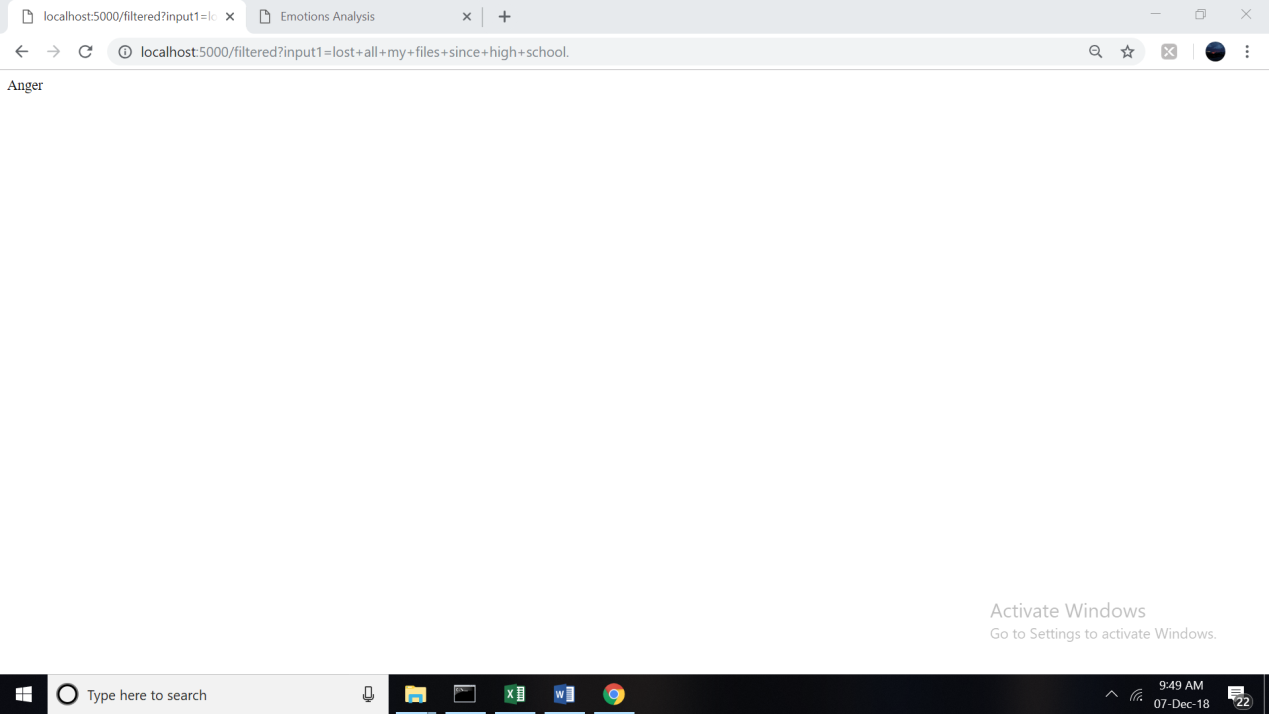


Fig. 5.2 Layout for Emotion Recognition System.

**References/Bibliography**

[www.kaggle.com](http://www.kaggle.com)

[www.udemy.com](http://www.udemy.com)

[www.w3school.com](http://www.w3school.com)

### Chapter 6: Appendix

Training the model:

**Code For Reading the Data set for train the Model:**

import pandas as pd

dataset=pd.read\_csv("main11.csv",encoding='latin1')

X1=dataset.iloc[:,1].values

Y1=dataset.iloc[:,0].values

**To LabelEncode the X1 Variable**

from sklearn.preprocessing import LabelEncoder

labelEncoder\_X=LabelEncoder()

X1=labelEncoder\_X.fit\_transform(X1)

**Remove the Useless words like Names from the variable Y1**

for i in range(0,25896):

lis=Y1[i].split(" ")

for k in lis:

if(k[0]=="@"):

lis.remove(k)

Y1[i]=" ".join(lis)

Emotion=re.sub('[^a-zA-Z]'," ",Y1[i])

ps=PorterStemmer()

Emotion=[ps.stem(word) for word in Emotion if not word in set(stopwords.words("english"))]

cr.append(" ".join(Emotion))

**CountVectorize the List :**

from sklearn.feature\_extraction.text import CountVectorizer

cv=CountVectorizer()

x=cv.fit\_transform(cr).toarray()

**Feautre Scalling:**

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

xP = sc.fit\_transform(x)

**Training the model:**

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n\_estimators = 90, criterion = 'entropy', random\_state = 0)

classifier.fit(xP,X1)

**To store the Trained Model in Pickle file (.PKL)**

joblib.dump(classifier,"modelllll.pkl")

joblib.dump(cv,"countvectorizer.pkl")

joblib.dump(sc,"feature.pkl")