

SYSTEM ARCHETECTURE

**USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

****

**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

****

**ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



CODE:

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

from tkinter.filedialog import askopenfilename

import numpy as np

import matplotlib.pyplot as plt

from sklearn.preprocessing import normalize

from sklearn.ensemble import RandomForestClassifier

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import normalize

from sklearn.metrics import precision\_score

from sklearn.metrics import recall\_score

from sklearn.metrics import f1\_score

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

import lightgbm as lgbm

import pandas as pd

from imblearn.over\_sampling import SMOTE

from keras.utils.np\_utils import to\_categorical

from keras.layers import MaxPooling2D

from keras.layers import Dense, Dropout, Activation, Flatten

from keras.layers import Convolution2D

from keras.models import Sequential

from keras.models import model\_from\_json

import pickle

global filename

global X,Y

global classifier

global dataset

global X\_train, X\_test, y\_train, y\_test

accuracy = []

precision = []

recall = []

fscore = []

global le

main = tkinter.Tk()

main.title("AI-Powered System Quantifies Suicide Indicators and Identifies Suicide-Related Content in Online Posts") #designing main screen

main.geometry("1300x1200")

def uploadDataset():

global filename

global dataset

text.delete('1.0', END)

filename = filedialog.askopenfilename(initialdir="Dataset")

text.insert(END,filename+" loaded\n\n")

dataset = pd.read\_csv(filename)

text.insert(END,"Dataset before applying machine translation\n\n")

text.insert(END,str(dataset.head()))

def processDataset():

global X,Y

global dataset

text.delete('1.0', END)

label = dataset.groupby('attempt\_suicide').size()

label.plot(kind="bar")

dataset.fillna(0, inplace = True)

text.insert(END,"All missing values are replaced with 0\n")

text.insert(END,"Total processed records found in dataset : "+str(dataset.shape[0])+"\n\n")

plt.show()

font = ('times', 16, 'bold')

title = Label(main, text='AI-Powered System Quantifies Suicide Indicators and Identifies Suicide-Related Content in Online Posts')

title.config(bg='dark goldenrod', fg='white')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=30,width=110)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=100)

text.config(font=font1)

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Suicide Attempt & Stressed Dataset", command=uploadDataset, bg='#ffb3fe')

uploadButton.place(x=900,y=100)

uploadButton.config(font=font1)

main.config(bg='RoyalBlue2')

main.mainloop()

