**TRUSTWISE ASSIGNMENT**

**Objective:**

The goal of thus project is to design and deploy a REST API that processes a given text which could be an output from Large Language Model – LLM, evaluates it using machine learning models, and returns a score based on predefined criteria. Additionally, all received texts and their corresponding scores are logged into a database and displayed through a user-friendly interface.

**TECH STACK USED:**

**Backend:** Flask (python)

**Frontend:** React.js and Tailwind CSS

**Database:** Mongo DB

**Graphs:** Chart.js

**Containerization:** Docker

**UI DESIGN:**

1. **Home page:**

Home page allows users to enter the text for analysis. Once the text is submitted, the backend processes it and returns a score, it is displayed on the same page.

A screenshot of a computer

Description automatically generated

*Home page*

1. **Analyze Page:**

This page has two sections in it.

1. **Analysis Trends:**

This section has a graphical representation of the scoring trends

A screenshot of a computer screen

Description automatically generated

*Analysis trends graphical representation*

1. **Analysis History:**

This section lists past analyses, showing previously processed texts and their scores. Users can also search the history using Search filter and Date filter.

A screenshot of a computer

Description automatically generated

*Analysis history along with the filter being applied*

**REPOSITORY STRUCTURE:**

Kruthi\_trustwise\_assignment/

|------backend/

| |----models/

| |----app.py

| |----config.py

| |----database.py

| |----Dockerfile

| |…

|-------frontend/

| |----src/

| |----Dockerfile

| |----nginx.conf

| |…

|------docker-compose.yml

|------README.md

***To run the repo:***

Download the zip file extract and give the below commands provided

1. Activate the virtual env (venv/Scripts/activate) in the backend and run it using (python app.py)
2. Run the frontend code using npm start

**BACKEND IMPLEMENTATION:**

cd backend

To start the implementation first we need to go into the backend folder in the repository for that use the below command

python3 -m venv venv

This command creates a virtual environment for the backend folder

venv\Scripts\activate

This command activates the virtual environment

pip install flask flask-cors pymongo transformers torch python-dotenv

This command is used to install all the dependencies

**flask:** The web framework for building APIs

**flask-cors:** Enables CORS for frontend-backend communication

**pymongo:** To connect with Mongo DB

**transformers:** Provides pre-trained NLP models

**torch:** Deep learning framework

**python-dotenv:** Manages environment variables

python app.py

This is used to run the code in the backend server

models/

|----\_init\_.py

|----toxicity.py

|----vectara.py

This is the structure of the models folder where toxicity.py and vectara.py are used in loading the huggingface models and getting the toxicity and vectara scores respectively

**backend/.env:** environment variables are stored here

**backend/config.py:** It manages configuration values like Mongo DB connection settings from environment variables

**backend/database.py:** It handles the insertion and retrieval of documents.

**backend/app.py:** this file is used to initialise the models and the database. /analyze and /history API endpoints are implemented in this file and stores the responses in the database.

Download **Postman** to test and check if the API endpoints are working

<http://localhost:5000/api/analyze> - to check analyze endpoint

<http://localhost:5000/api/history> - to check history endpoint

A screenshot of a computer

Description automatically generated

*Postman testing outputs*

Download **Mongo DB** and connect it to see the tested texts and their responses

A screenshot of a computer

Description automatically generated

*Stored data in Mongo DB*

**FRONTEND IMPLEMENTATION:**

To create a basic react app in frontend terminal use  
npx create-react-app .

Structure:

src/

|----components/

|-----Footer.js

|-----Navbar.js

|----pages/

|----AnalyzePage.js

|----LandingPage.js

|----App.js

|----index.js

…

**Footer.js:** Has the footer section’s implementation

**Navbar.js:** It has the navigation bar’s implementation

**LandingPage:** This is the home page implementation where the input text is taken and score is given.

**AnalyzePage:** This page has the implementation of the graphical representation of the scoring trends and the analysis history

**App.js:** It sets up the structure of the app and handles routing using react router.

**DOCKER FOR CONTAINERIZATION:**

1. Download docker desktop and create a Dockerfile in both frontend and backend folders.
2. Build the docker-compose file and run the container by using

docker-compose up --build

new port frontend: <https://localhost:3001>

new port backend: <https://localhost:5001>

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated