# Sentiment-Based Product Recommendation System

## Project Overview

The Sentiment-Based Product Recommendation System leverages Natural Language Processing (NLP) and Recommendation Systems to enhance product suggestions for users. It analyzes customer reviews to understand sentiment (positive or negative) and recommends top-rated products using collaborative filtering techniques.

## Objectives

- Perform sentiment analysis on customer product reviews.
- Build machine learning models to classify sentiments.
- Develop user-based and item-based collaborative filtering recommendation systems.
- Integrate both components to recommend products based on sentiment and user similarity.
- Deploy the end-to-end application using Flask and Heroku.

## Technologies Used

Python, Pandas, NumPy, Scikit-learn, XGBoost, NLTK, Flask, Joblib, Heroku, and GitHub.

## Project Structure

```
Sentiment_Recommendation_System/
│
├── Sentiment_Recommendation_System.ipynb  ← Jupyter Notebook (end-to-end code)
│
├── deployment/
│   ├── app.py              ← Flask backend
│   ├── model.py            ← ML model loading & prediction logic
│   ├── requirements.txt    ← Dependencies list
│   ├── Procfile            ← For Heroku deployment
│   ├── runtime.txt         ← Python version
│   ├── templates/
│   │   └── index.html      ← Frontend HTML page
│   ├── static/
│   │   └── style.css       ← Optional styling
│   └── models/
│       ├── sentiment_model.pkl ← Saved model
│       ├── tfidf.pkl       ← TF-IDF vectorizer
│       └── recommender.pkl ← Optional recommendation file
│
└── README.md  ← Summary and documentation
```

## Overview

- Data Cleaning & Preprocessing
- Text Preprocessing (Stopwords, Lemmatization)
- Feature Extraction using TF-IDF
- Model Building (Logistic, Random Forest, Naive Bayes, XGBoost)
- User-Based & Item-Based Recommendation Systems
- Recommend Top-20 Products
- Fine-tuning with Sentiment
- Deployment with Flask

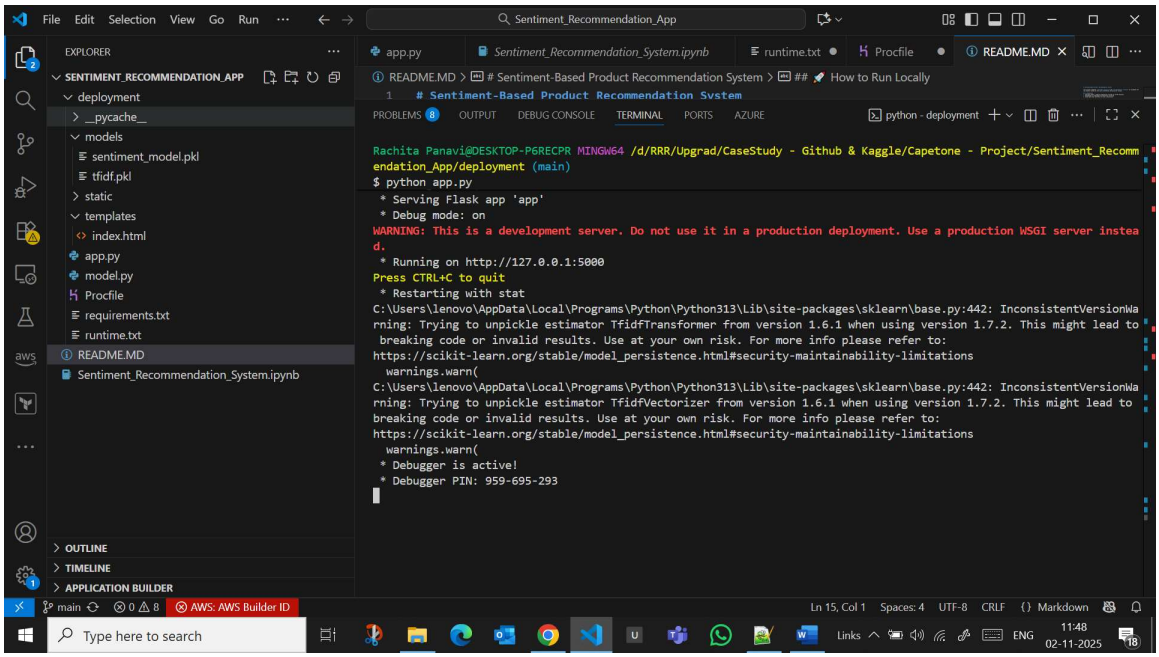
## Deployment Files Description

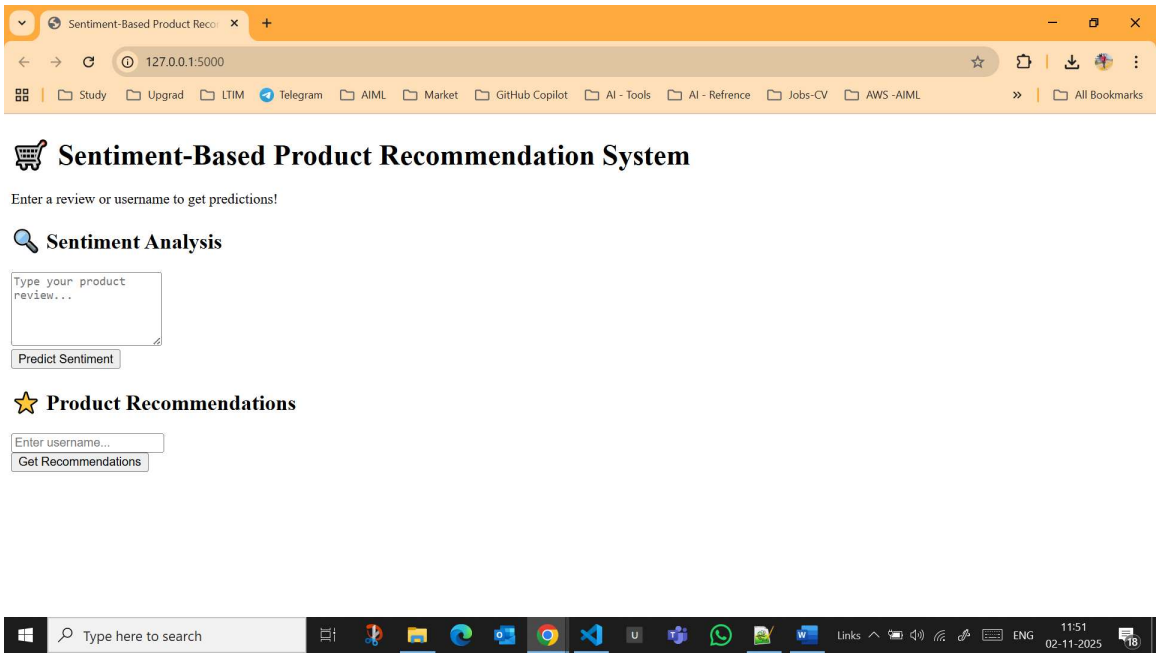
• app.py – Flask routes for sentiment prediction and recommendations.
• model.py – Loads the trained model, TF-IDF, and recommendation data.
• index.html – Web interface for inputs and outputs.
• requirements.txt – Lists dependencies.
• runtime.txt – Specifies Python version compatibility.

## Results Summary

Successfully cleaned and processed the dataset.
Built and evaluated multiple machine learning models for sentiment prediction.
Implemented user-based and item-based recommendation systems.
Deployed the integrated web application using Flask.
Users can input a review or username to get real-time predictions and recommendations.

## Deployment Screen Shot






## 10. Author

Author: Ravi P.

Email: ravipanavirs@gmail.com