Contents

[**Introduction** 2](#_Toc182987549)

[1.1 Project Title 2](#_Toc182987550)

[Project Overview Statement 2](#_Toc182987551)

[Goals & Objectives 4](#_Toc182987552)

[Project Objectives: 4](#_Toc182987553)

[High-level system components: 4](#_Toc182987554)

[List of optional functional units 5](#_Toc182987555)

[Exclusions: 5](#_Toc182987556)

[Application Architecture: 5](#_Toc182987557)

[Gantt chart: 6](#_Toc182987558)

[Hardware and Software Specification: 6](#_Toc182987559)

[Additional Libraries: 6](#_Toc182987560)

[Tools and technologies used with reasoning: 6](#_Toc182987561)

**Final Project Proposal Guide**

# **Introduction**

This project aims to develop a Flask-based web application for performing sentiment analysis on product reviews. The system will support role-based access, with two user types: Admin and Sentiment User. Admins will have access to dashboards, review statistics, and visualizations, while Sentiment Users can submit reviews for analysis and generate custom replies. Additional functionality includes a sample review page to demonstrate sentiment analysis and bulk sentiment analysis for processing datasets, making the system versatile for business needs.

## Project Title

Sentiment Analysis and Custom Reply Generation for Product Reviews with Role-Based Access Using Flask

## Project Overview Statement

The objective of this project is to create a Flask-based web application that performs sentiment analysis on user-submitted product reviews. By implementing a role-based access system, the application will have two types of users: Admin and Sentiment User. Admins will be able to view overall sentiment trends, review history, and visualize data. Sentiment Users will be able to input text reviews, receive sentiment analysis, and generate custom responses based on detected sentiment and keywords. This project combines web development, role-based access control, and Natural Language Processing (NLP) for a streamlined user experience

­­­­­­­­

|  |  |  |  |
| --- | --- | --- | --- |
| Project Title: **Sentiment Analysis and Custom Reply Generation for Product Reviews with Role-Based Access Using Flask** | | | |
| Group Leader: Muhammad Sohail Amjad | | | |
| |  |  |  |  | | --- | --- | --- | --- | | Name | Registration # | Email Address | Signature | | Muhammad Sohail Amjad | 21\_UE\_04881 | sohailamjad418@gmail.com |  | | Anas Ghafoor | 21\_UE\_04834 | snasghafoor2001@gmail.com |  | | Muhammad Ramzan | 21\_UE\_05430 | ramzan.bsit@gmail.com |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |   Project Members: | | | |
| **Project Goal:**   1. Sentiment Analysis: 2. Role Based Access 3. Custom Reply Generation 4. Sample Review Page 5. Bulk Sentiment Analysis | | | |
| **Objectives:** | | | |
|  | 1 | **Sentiment Analysis**: Build a system to categorize user sentiments as positive, negative, or neutral. |  |
|  | 2 | **Role-Based Access**: Implement secure, tiered access for users based on their roles. |
|  | 3 | **Custom Reply Generation**: Create a system to generate context-aware, personalized responses |
|  | 4 | **Sample Review Page**: to design a sample page for implementing app’s functionality on individual comment and on all comments |
|  | 5 | **Bulk Sentiment Analysis**: Enable efficient processing of large datasets for sentiment analysis. |
|  |  |  |
|  |  |  |
| **Project Success criteria:**  All the objectives are achieved according to goals | | | |
| **Assumptions, Risks and Obstacles:** Implementing app functionality on bulk data and there might be risks and obstacles on analysis of individual comments | | | |
| Organization Address (if any): | | | |
| Type of project: Research Development | | | |
| Target End users: online shoping stores owners | | | |
| Development Technology: Object Oriented Structured | | | |
| Platform: Web based Distributed  Desktop based Setup Configurations  Other | | | |
| Suggested Project Supervisor: | | | |
| Approved By: | | | |
| Date: | | | |

## Goals & Objectives

**Goals:**

1. **Sentiment Analysis:**
   * + Build a Flask-based app to analyze sentiments of product reviews.
   * Classify sentiments into Positive, Neutral, or Negative categories.
2. **Role-Based Access:**
   * Implement two user roles: Admin and Sentiment User.
     + **Admin:** Access dashboards, sentiment statistics, and data visualizations.
     + **Sentiment User:** Submit reviews for sentiment analysis and generate replies.
3. **Custom Reply Generation:**
   * Generate personalized responses to reviews based on sentiment and detected keywords.
4. **Sample Review Page:**
   * Add a page displaying a set of pre-loaded sample comments.
   * Perform real-time sentiment analysis on these comments.
5. **Bulk Sentiment Analysis:**
   * Enable users to upload datasets (e.g., CSV files) for analysis.
   * Visualize results using charts (e.g., sentiment distributions and trends)..

## Project Objectives:

1. Develop a system for sentiment analysis that effectively categorizes user-submitted product reviews.
2. Create a role-based access system to differentiate functionalities for Admins and Sentiment Users.
3. Design and implement a dashboard for Admins with statistical visualizations of review sentiments.
4. Enable sentiment-based custom reply generation to enhance customer engagement.

Implement data visualizations (e.g., pie and line charts) to showcase sentiment distributions and trends over time

## High-level system components:

1. **Sentiment Analysis Module:**
   * **Description:** Analyzes and classifies sentiment as Positive, Neutral, or Negative using NLP models.
   * **Purpose:** Helps businesses understand customer feedback and improve products or services.
2. **Role-Based Access Control:**
   * **Description:** Manages user roles for Admin and Sentiment User using Flask-RBAC.
   * **Purpose:** Allows Admins to view statistics and Sentiment Users to submit reviews and generate replies.
3. **Custom Reply Generation Module:**
   * **Description:** Generates personalized replies based on sentiment classification and detected keywords.
   * **Purpose:** Enhances customer service by providing meaningful responses.
4. **Data Visualization and Dashboard:**
   * **Description:** Displays charts and visual data, including sentiment trends, distributions, and common keywords.
   * **Purpose:** Provides Admins with actionable insights for decision-mak

## List of optional functional units

1. Performance optimization for faster analysis and response generation.
2. Automated sentiment analysis improvements based on user feedback.
3. Compatibility with additional NLP models
4. Enhanced security and auditing controls for role-based access.

## Exclusions:

1. Real-time feedback collection from external platforms.
2. Multilingual support for reviews other than English.
3. Extended functionalities that require external API integrations.

.

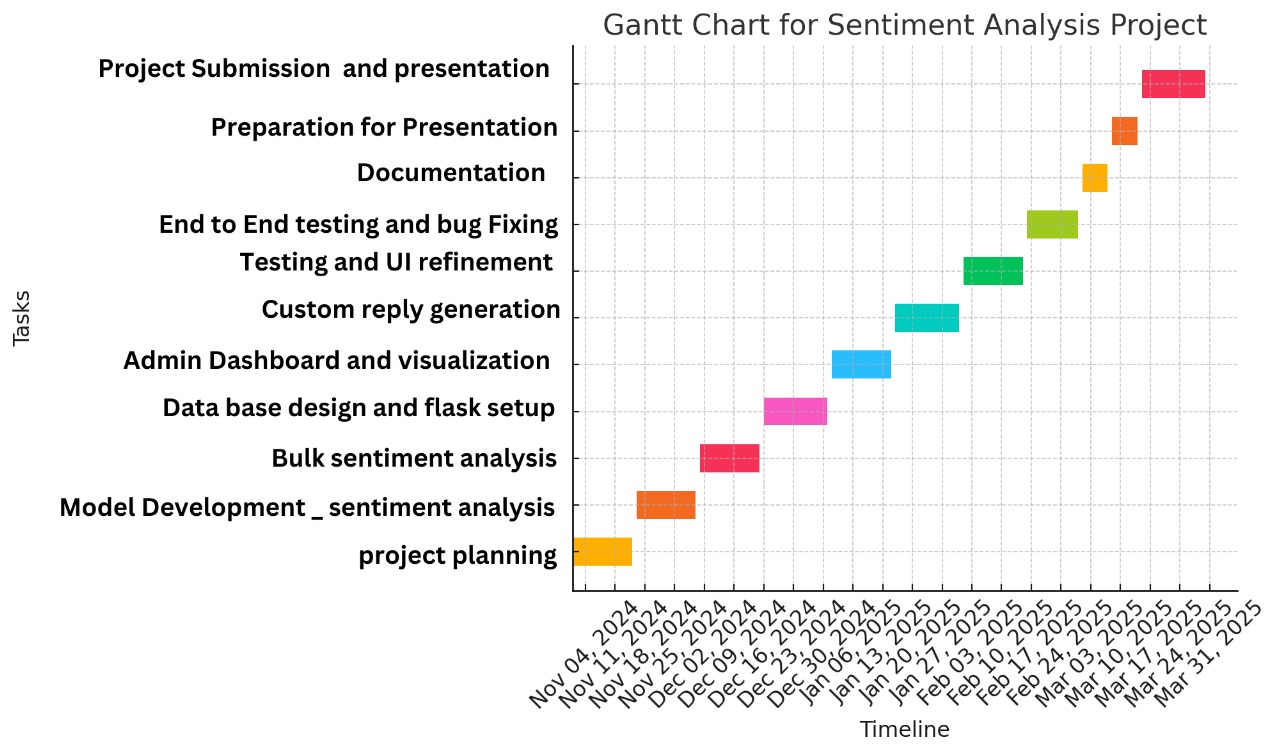
## Application Architecture:

**Frontend (Presentation Layer):** HTML, CSS, Bootstrap, JavaScript for user interface components.

**Backend (Application Layer):** Flask and Python for application logic, sentiment analysis, and custom reply generation.

**Database (Data Layer):** SQLite to store user data, reviews analysis report

## Gantt chart:



## Hardware and Software Specification:

* **Backend:** Flask, Python
* **Database:** SQLite or PostgreSQL
* **Machine Learning:** Pre-trained NLP models (e.g., BERT,LSTM) for sentiment analysis and keyword detection.
* **Frontend:** HTML/CSS, Bootstrap, JavaScript
* **Additional Libraries:** Flask-RBAC, SQLAlchemy, Matplotlib, and Seaborn

.

## Additional Libraries:

Flask-RBAC for role management, SQLAlchemy for database management.

## Tools and technologies used with reasoning:

* **Backend Development:** Flask for APIs and backend logic.
* **Database:** SQLite/PostgreSQL for flexible data management.
* **Machine Learning:** NLP models (e.g., BERT for accuracy in sentiment analysis).
* **Frontend Development:** HTML/CSS, Bootstrap, and JavaScript for a responsive UI.
* **Visualizations:** Matplotlib and Seaborn for insightful charts.
* **Additional Libraries:** Flask-RBAC for role-based access, SQLAlchemy for efficient database management.