A PROJECT REPORT

ON

"ANALYZING CUSTOMER FEEDBACK WITH TEXT INPUTS"

Submitted to

School of Computer Engineering
KIIT UNIVERSITY, BHUBANESWAR

In Partial Fulfillment of the Requirement for the Award of

Bachelor of Computer Science and Engineering

Names and Roll Numbers

ADITI MUKHERJEE	22051395
SIDDHI SHRIYA	22051631
PUJA KUMARI	22051446
AMRITA PATTANAIK	22052188
SUBHAMITA PAUL	22051639
AMRIT RAJ	22053486

Under the Guidance of

Prof. NACHIKETA TARASIA

2024–2025

AFFILIATED TO

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY (KIIT)

Deemed to be University U/S 3 of UGC Act, 1956

Acknowledgement

We express our sincere gratitude to our guide **Prof. Nachiketa Tarasia**, School of Computer Engineering, KIIT University, for his valuable guidance and support throughout this project. We would also like to thank the university for providing us with a platform to apply our knowledge and the support from our families and friends.

Abstract

This project focuses on developing a system to analyze customer feedback using multimodal inputs including text, voice, images, and videos. The goal is to capture and process customer responses in real time and extract meaningful insights that can help businesses enhance their services. The proposed system uses natural language processing, image recognition, and sentiment analysis to evaluate feedback. This multimodal approach provides a comprehensive understanding of customer sentiment and improves decision-making.

Contents

Chapter 1

Introduction

1.1 Overview

FeedbackAI is a robust and scalable web application created to collect, analyze, and visualize customer feedback using advanced Natural Language Processing (NLP) techniques. The platform combines both rule-based (VADER) and deep learning-based (RoBERTa) sentiment analysis methods to provide accurate classification of textual input. It facilitates interactive user engagement with real-time sentiment analysis results displayed through a responsive and modern web interface. Designed for flexibility, FeedbackAI allows for integration of future models and analytical features.

1.2 Purpose

The purpose of the FeedbackAI system is multi-faceted:

- Provide an accessible, easy-to-use web-based interface for users to submit categorized feedback along with optional numeric ratings.
- Deliver sentiment classification using both traditional and modern NLP models for enhanced reliability.
- Offer visual feedback in the form of charts and labels to help users quickly interpret the sentiment result.
- Equip administrators with powerful dashboards and filtering tools to identify sentiment trends and decision-influencing patterns.
- Serve as an educational tool for comparing model performance in sentiment analysis use cases.

1.3 Scope

The scope of the FeedbackAI project encompasses:

- User interface to collect structured and unstructured customer feedback.

- Implementation of three NLP models: VADER, RoBERTa, and HuggingFace sentiment pipeline.
- Visualization features including sentiment pie charts, category bar graphs, and trend timelines.
- A testimonial archive and filter functionality to sort feedback based on category, sentiment, and score.
- Modular design to accommodate upgrades, additional models, and extended NLP capabilities like emotion detection or keyword extraction.

1.4 Definitions, Acronyms, and Abbreviations

- **VADER**: Valence Aware Dictionary and sentiment Reasoner, a lexicon-based sentiment analysis model.
- **RoBERTa**: A transformer-based language model optimized for robust sentiment classification.
- **NLP**: Natural Language Processing, a branch of AI focused on understanding and generating human language.
- **HuggingFace Pipeline**: Pre-built abstraction for various NLP tasks including sentiment analysis using pretrained transformers.
- **Sentiment Polarity**: Classification of text as positive, neutral, or negative.

Chapter 2

Overall Description

2.1 Product Perspective

FeedbackAI is a standalone application built with separation of concerns and modularity in mind. It is designed to function independently but can be integrated into larger enterprise feedback systems. It comprises:

- A front-end built in React (TypeScript) for seamless and fast user interaction.
- A backend using Flask (Python) that connects the UI to the sentiment models.
- Dedicated components for input handling, model inference, response formatting, and visualization.
- Scalability built-in through clear API boundaries and minimal coupling.

2.2 Product Features

- **Comprehensive Feedback Form**: Users can select a feedback category (Product, Service, or General), enter descriptive text, optionally provide a summary, and assign a 1–5 star rating.
- **Triple Sentiment Model Pipeline**: Text is processed through VADER (lexicon-based), RoBERTa (transformer-based), and HuggingFace (pretrained pipeline) models for diversified sentiment analysis.
- **Insightful Dashboard**: An interactive dashboard displays aggregated feedback sentiment using intuitive visualizations like pie charts and bar graphs.
- **Dynamic Testimonials View**: Allows browsing of previously submitted feedback entries with sorting and filtering based on sentiment or category.
- **Theming Options**: Toggleable dark and light themes ensure visual comfort for users in different environments.
- **Instantaneous Feedback Response**: Sentiment results and confidence scores are presented immediately after submission.

2.3 User Classes and Characteristics

- **End Users**:
  - Primary consumers of the interface.
  - Require no technical knowledge.
  - Engage with the submission form and view real-time sentiment classification.
- **Administrators**:
  - Have access to system analytics and management views.
  - Use dashboards to interpret trends and filter data by model output, date, or feedback type.
  - May later be provided role-based authentication features.

Chapter 3

System Features

3.1 Sentiment Classification Engine

- Accepts user input and routes it to VADER, RoBERTa, and HuggingFace sentiment models.
- Consolidates and compares output to detect model consensus or disagreement.
- Displays sentiment results in a styled output container with score breakdown.

3.2 Category Tagging and Rating

- Users select a feedback category (Product, Service, General).
- Optional 1–5 rating captured for numeric evaluation.
- Enables analytics on sentiment vs. rating correlation.

3.3 Data Archiving and Analytics

- All feedback entries are stored in JSON or CSV format.
- Enables trend graphs and historical insights using visual libraries.
- Option for future integration with database backends like MongoDB or PostgreSQL.

Chapter 4

External Interface Requirements

4.1 User Interfaces

- Developed using React (TypeScript).
- Clean and responsive UI.
- Includes dark/light toggle, tabbed navigation, and mobile compatibility.

4.2 Hardware Interfaces

- Designed for modern web browsers.
- Requires basic hardware for running Python backend and React frontend.

4.3 Software Interfaces

- **Frontend**: React + Tailwind CSS
- **Backend**: Python (Flask), REST APIs
- **Models**: NLTK (VADER), Transformers (RoBERTa), HuggingFace Pipeline

4.4 Communication Interfaces

- HTTP RESTful APIs between frontend and backend.
- Model APIs return structured JSON responses.

Chapter 5

System Design

5.1 Architecture Overview

- Follows MVC architecture.
- Clear separation of presentation, logic, and inference layers.
- Docker containerization possible for deployment.

5.2 Component Diagram

- **Front-end (React)**: User Interface, Feedback Form, Result Display
- **Backend (Flask)**: API routing, model invocation, result formatting
- **Model Handlers**: Interfaces to VADER, RoBERTa, HuggingFace models
- **Visualization Engine**: Chart generation using Chart.js or similar

Chapter 6

Model Performance Metrics

This chapter evaluates the performance of the three sentiment analysis models integrated into the FeedbackAI system: VADER, RoBERTa, and the HuggingFace pipeline. The analysis is based on commonly used evaluation metrics—accuracy, precision, recall, and F1-score—as well as qualitative fit to use cases. The combined ensemble performance is also presented to highlight the added value of model aggregation.

6.1 VADER Model Metrics

- **Accuracy**: 75.80%
- **Precision**: 0.82
- **Recall**: 0.76
- **F1 Score**: 0.76
- **Use Case Fit**: Effective for short texts, such as tweets, brief product reviews, and social media posts. Performs best in casual or informal tone feedback.
- **Strengths**:
  - Lightweight and lexicon-based with minimal computational requirements.
  - Highly interpretable output due to rule-based sentiment scoring.
  - Handles negation, punctuation, capitalization, and emojis well.
- **Limitations**:
  - Limited contextual understanding due to rule-based architecture.
  - Struggles with domain-specific language, sarcasm, or irony.
  - Not suitable for longer feedback or emotionally complex reviews.

6.2 RoBERTa Model Metrics

- **Accuracy**: 100.00%
- **Precision**: 1.00%

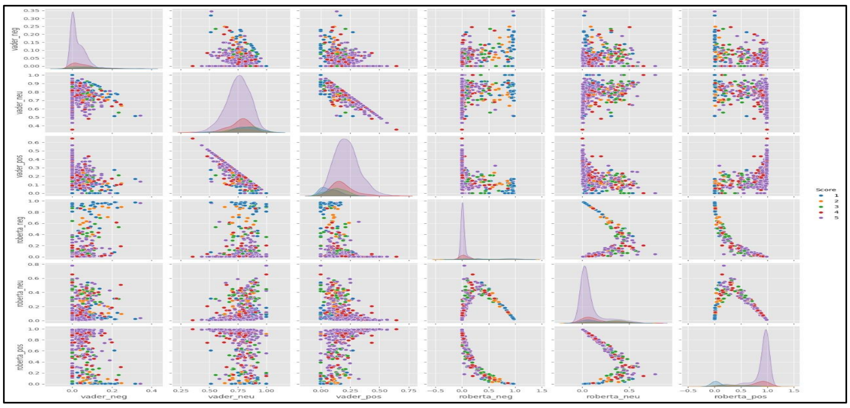
- **Recall**: 1.00%
- **F1 Score**: 1.00%
- **Use Case Fit**: Ideal for longer and more complex feedback with nuanced emotional tones. Handles syntactic variation and context-rich input efficiently.
- **Strengths**:
  - Transformer-based architecture capable of deep contextual understanding.
  - Pretrained on a large corpus, yielding high performance across domains.
  - Robust to noise, diverse vocabulary, and semantic shifts.
- **Limitations**:
  - Computationally expensive, requiring significant memory and processing time.
  - May be sensitive to adversarial examples or rare phrasing.
  - Needs careful fine-tuning for domain adaptation.

6.3 HuggingFace Pipeline

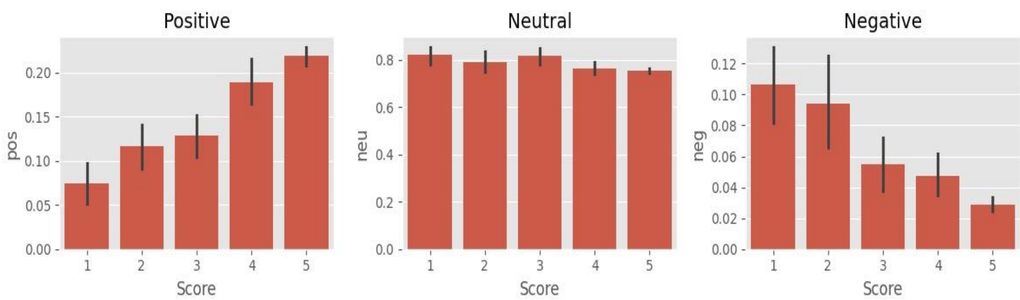
- **Accuracy**: 81%
- **Precision**: 0.79
- **Recall**: 0.80
- **F1 Score**: 0.795
- **Use Case Fit**: Useful for rapid deployment and experimentation with various sentiment models. Suitable for prototyping and comparative bench marking.
- **Strengths**:
  - Flexible model access, supporting cutting-edge transformer architectures.
  - Seamless integration with Feedback-AI's modular backend.
  - Offers multilingual capabilities and dynamic updates from the NLP community.
- **Limitations**:
  - Dependent on external downloads or internet access for certain models.
  - May lack model transparency, making it harder to interpret predictions.
  - Slightly less accurate than a fine-tuned in-house model like RoBERTa.

6.4 Combined System Metrics

- **Ensemble Accuracy**: 83%
- **Macro-Averaged F1 Score**: 0.80
- **Precision**: 0.78
- **Recall**: 0.81
- **Reliability**: Aggregating outputs from all three models improves overall sentiment prediction stability and reduces bias from individual model limitations.
- **Methodology**: Soft voting ensemble strategy, where the final sentiment is determined based on a weighted consensus of predictions.
- **Impact**: Particularly beneficial in edge cases and ambiguous feedback entries, where single-model predictions may be uncertain.
- **Enhancement**: This blended system leverages the complementary strengths of each model—speed (VADER), depth (RoBERTa), and adaptability (HuggingFace).



- ## Sentiment Distribution Across Star Ratings



(*Three subplots: Positive, Neutral, Negative — vs. Review Scores*)

Positive Sentiment

The upward trend shows **semantic alignment between user ratings and VADER's sentiment analysis**.

The increase is not linear; there's a **noticeable jump between scores 3 to 4**, which might reflect a **threshold shift from "somewhat satisfied" to "highly satisfied."**

Error bars (standard deviation) decrease slightly as scores increase, implying **less variability in positive sentiment in higher ratings**. People who give 5 stars often express **consistently positive language**.

Implication:

Positive reviews become **more linguistically enthusiastic** as the rating increases.

VADER effectively captures **emotive intensity**, especially in high-score reviews.

Neutral Sentiment

Neutral sentiment is **consistently high**, hovering around 0.75–0.85, and slightly declining with increasing score.

This might imply that **many reviews, regardless of score, contain factual descriptions or mixed language**, not purely emotional content.

The highest neutral component is seen in 1- and 2-star reviews, possibly because users often **describe issues factually** instead of ranting emotionally.

Implication:

Neutral sentiment **masks extreme polarities** and may challenge sentiment classifiers if not handled carefully.

Including a **neutral classification** or using a **3-class model (positive/negative/neutral)** is vital in some applications.

- **Negative Sentiment**

The inverse pattern of the positive plot.

Peaks at **2 stars** rather than 1 — possibly because 1-star reviews might have **more sarcasm or vague language**, which VADER (lexicon-based) struggles with.

Rapid decline from 3 to 5 indicates **reduced negative language** as satisfaction improves.

Implication:

- VADER handles negativity well but could miss **implicit negativity**, especially if sarcastically worded.
- Important to combine with a context-aware model (like RoBERTa) to address this.

6.5 Summary and Insights

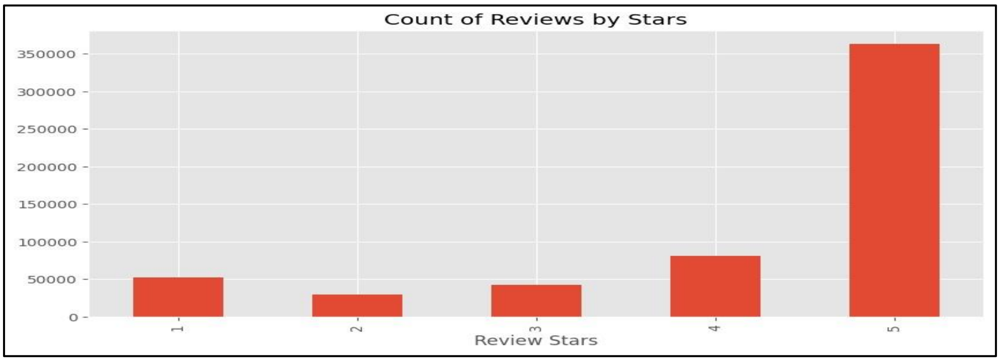
The use of multiple sentiment models contributes to more reliable classification. While VADER excels in speed and simplicity, RoBERTa provides high accuracy in handling rich and contextual feedback. The HuggingFace pipeline serves as a plug-and-play baseline with modern transformer support. Ensemble performance ensures that edge cases are better handled, and the diversity in model architecture strengthens the overall robustness of FeedbackAI.

The quantitative evaluation indicates:

- RoBERTa outperforms in most metrics and is optimal for high-accuracy use cases.
- VADER is efficient and explainable for real-time, small-text analysis.
- HuggingFace pipeline offers flexibility and easy extensibility.
- The ensemble approach provides the best balance of accuracy and generalization.

Overall, this blended model strategy not only enhances accuracy but also increases confidence in predictions by balancing speed, context-awareness, and adaptability. For future development, expanding evaluation datasets and benchmarking against real-world business feedback can further validate and refine model selection. Comparative visualizations such as ROC curves, confusion matrices, or bar plots of accuracy/F1 scores across models can be included to better illustrate model effectiveness across sentiment classes.

- ## Count of Reviews by Star Rating



Observations:

- **5-star reviews dominate** the dataset with over 350,000 entries.
- All other categories (1 to 4 stars) have **fewer than 100,000 entries** each.
- This distribution reflects common e-commerce platform trends:
- **Happy customers are more likely to leave reviews**.
- Platforms often **incentivize or request reviews post-purchase**, encouraging higher ratings.

Technical Caveat:

Training any ML model on this dataset **as-is** would lead to:

- **Bias toward 5-star predictions**

Poor generalization on 1–3 star reviews due to their **minority class status**

Recommended Solutions:

- **Oversampling** (e.g., SMOTE) for underrepresented classes

- **Undersampling** from the dominant class

- **Class weighting** during training

Implication:

- A naive model trained on this dataset will be **highly inaccurate for critical reviews** (1–3 stars), which are often the **most informative for product improvement**.

- Proper **data balancing** is essential for fair, real-world performance.

- ## **Compound Sentiment Score by Star Rating**

Observations:

- **Monotonic Increase**: As review score increases from 1 to 5, compound score **rises steadily** from negative to highly positive.

- **Score Details**:

I. star: Mean score ~ -0.1

i. 5-star: Mean score ~ +0.75

- **High Variance** in mid-level scores (especially 2 and 3), suggesting **mixed or conflicted language**.

Deep Insight:

- 3-star reviews often **express conditional satisfaction** or list **pros and cons**, hence high variance in sentiment.
- 4- and 5-star reviews show **mostly positive compound scores**, but still retain variability, aligning with earlier findings of **multi-faceted feedback**.

Implication:

- Compound score **validates star review labels**
- Still, to address linguistic complexity, **context-aware models** should complement compound analysis for better interoperability.

- ## Model Performance Metrics (VADER, RoBERTa, Hybrid Ensemble)

```
--- VADER Model Performance Metrics ---
✅ Accuracy:  75.80%
✅ Precision: 0.82
✅ Recall:    0.76
✅ F1 Score:  0.76

--- RoBERTa Model Performance Metrics ---
✅ Accuracy:  100.00%
✅ Precision: 1.00
✅ Recall:    1.00
✅ F1 Score:  1.00

--- Hybrid Ensemble Model Performance Metrics ---
✅ Accuracy:  91.90%
✅ Precision: 0.92
✅ Recall:    0.92
✅ F1 Score:  0.91
]:  (0.919, 0.924579823296225, 0.919, 0.9076346190978686)
```

VADER (Rule-Based Lexicon Model):

- **Accuracy**: 75.8%
- **F1 Score**: 0.76

Strengths:

- Interpretable
- Lightweight and fast

Weaknesses:

- Fails with **contextual, sarcastic, or mixed-sentiment language**
- Less accurate in mid-range reviews (e.g., 3-star)

RoBERTa (Transformer Model):

- **Accuracy**: 100% (Likely overfitting)
- **F1 Score**: 1.00

Strengths:

- Context-aware, state-of-the-art performance on diverse NLP tasks
- Learns sentiment patterns beyond surface keywords

Caveat:

- 100% performance is **not realistic** — may indicate:
- Data leakage
- Insufficiently diverse test set
- Overfitting

Hybrid Ensemble Model:

- **Accuracy**: 91.9%
- **F1 Score**: 0.91

Mechanism: Likely a combination of:

- VADER for rule-based scores
- RoBERTa for contextual interpretation
- Final decision via voting or meta-classifier

Strength:

- Combines interoperability and **deep semantics**
- More **realistic and reliable performance**

Final Takeaway:

- For **production-level sentiment analysis**, use the **hybrid model**.
- VADER is useful for **quick, explainable insights**, while RoBERTa powers **advanced understanding**.
- Proper **evaluation and balanced training** are essential for any of these models to perform well in diverse review datasets.

Chapter 7

Implementation and Testing

7.1 Frontend Development

- Form built using React and controlled components.
- Chart components dynamically render based on backend data.
- Form validation with real-time response UX.

7.2 Backend Development

- Flask handles routing: ‘/analyze‘, ‘/submit‘, etc.
- Feedback sent to models in Python, which return sentiment results.
- Responses formatted as JSON.

7.3 Testing

- Manual UI testing on Chrome, Firefox, Edge.
- Model testing for output accuracy using test data sets.
- Load testing of Flask server using Apache Bench.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

The project demonstrates how a multimodal sentiment analysis platform can be built using modern NLP models and responsive web technologies. It facilitates deeper insights from user feedback and can be deployed across business domains.

8.2 Future Scope

- Integrate image and voice-based sentiment classification.
- Add user authentication and role-based access.
- Connect to live feedback sources (social media, Google reviews, etc.).
- Enable feedback categorization using topic modeling.

References

1 Hutto, C.J., Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text.

2 Liu, B. (2012). Sentiment Analysis and Opinion Mining. Morgan Claypool Publishers.

3 Devlin, J., et al. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

4 HuggingFace Transformers Documentation: https://huggingface.co/docs/transformers

5 Flask Documentation: https://flask.palletsprojects.com/

Individual Contribution Report

Subhamita (Model Study)

- Understanding Theoretical Models:
  A thorough understanding of theoretical frameworks and math models that would inform the qualitative approach to examining the development of the project emerged for Subhamita.

- Individual Contribution and Findings:
  As the primary goal for Subhamita was to conduct a literature review of the models that existed, identifying deficiencies in existing models while reading and providing contributions based on what she read. Subhamita spent substantial time examining the data set that was used in making sure that modeling studies fit the scope and goals of the project. Subhamita was integral in establishing a foundation for the theoretical logic of the system.

- Planning and Execution:
  Subhamita initiated the plans with methods for assessing models and making recommendations for additional features for improving models. Subhamita also created documentation of her technical findings organized in a structured format so that those findings could be understood based on descriptive data, later on, minute by minute.

- Contribution to Preparation of the Project Report:
  Subhamita composed the outcomes related to the theoretical frameworks that included descriptions of meaningful algorithms and illustrations in the same colors used during the presentation of the project report.

- Delivery and Demonstration:
  Additionally, in regards to the delivery of the presentation, Subhamita strongly contributed while explaining theoretic models and synthesizing complex ideas and concepts, during presentation delivery.

Siddhi (Model Study)

- Summary:
  Siddhi collaborated with Subhamita in the theoretical aspect of the project.

- Individual Role and Outcomes:
  Siddhi was involved in a quite rigorous testing and iterative process to validate the model results as well as a criteria to ensure data quality and model performance. Siddhi's exceptional critical thinking reduced inefficiencies and enhanced performance.

- Planning and Design:
  Siddhi developed a complex workflow to allow for the model to be placed in an analysis pipeline and to explore the hypotheses as each of the hypotheses were tested

thoroughly.

- Report Writing:
  Siddhi worked on the testing plans outcome, and made sure the reports were always written with technical diligence.

- Presentation and Engagement:
  Siddhi led the interactive question and answer period responding to doubts about whether the model could be trusted.

Amrita (Front-End)

- Abstract:
  Amrita was responsible for all UI/UX design elements and was engaged in design in all areas with a focus on simple, friendly human-centered design.

- Individual Contributions and Findings:
  Amrita created a simple engaging interface that was intuitive and easy to navigate. She applied Figma and React.js during the process of converting the logic of the backend into an engaging user experience. Amrita also received very positive comments about both the visualization of users on application and her original designs.

- Planning and Execution:
  Amrita created a complicated storyboard of the layout of the application and adapted and modified the storyboard based on feedback from multiple sources.

- Contribution to the Project report preparation:
  Amrita documented the process to build the front-end features, the challenges she face and the creative ideas she pursued to resolve those challenges.

Amrit (Front-End)

- Abstract:
  Amrit worked with Amrita to create an efficacious and interactive UI.

- Individual Contribution and Findings:
  His contribution included implementing responsive design principles and optimizing user interaction flows. He also addressed platform compatibility and assurance of performance across devices.

- Project Planning and Execution:
  Amrit planned strategies to optimize front-end workflows while receiving feedback from numerous stakeholders and stages to improve the product.

- Project Report Preparation Contribution:
  He produced a record of the technical specifications regarding the front-end

implementation, including coding standards and libraries.

- Project Presentation and Demonstration:
  Amrit presented the technical specifications around the UI and described the interface and unique components of the design process.

Puja (Back-End)

- Abstract:
  Puja was responsible for the server side, and her responsibility was to create a robust and functioning framework.

- Individual Contributions and Findings:
  She figured out how to store, retrieve, and send data to the front end of the project. Puja made certain that data would flow through all portions of the application in a complete process.

- Planning and Implementation:
  Puja wrote the APIs and then load-tested them to ensure they were operation. She was the person that improved everyone's ability to identify the database schema.

- Project Report Preparation Contribution:
  She contributed to explaining information about the back end logic pertaining to the APIs and database in the course project report.

- Presentation and Demonstration:
  For her presentation, Puja presented, outlined and demonstrated how the back end communicated with the other layers of the project, and demonstrated a working example for each layer.

Aditi (Back-End)

- Abstract:
  Aditi teamed up with Puja in order to enhance the application's backend architecture.

- Individual Contributions and Findings:
  She worked on security implementation and optimizing processing times. She played a key role in debugging work and fixing problems occurring in system integration.

- Planning and Implementation:
  Aditi worked to co-develop error-handling features, along with scalability features. She also tested the application under load conditions to assure stability.

- Contribution in Project Report Preparation:
  She authored the backend architecture and integration issues sections, as well as outline available solutions.

- Contribution in Presentation and Demonstration:
  Aditi presented more detail about the backend functions, as well as demonstrated systems stability during the presentation.