**Project Overview:**

The AI-driven Guest Personalization System is designed to revolutionize the hospitality experience by utilizing Large Language Models (LLMs) and sentiment analysis. The system analyzes guest feedback, understands preferences, and delivers personalized recommendations to enhance guest satisfaction. By implementing real-time alerts and actionable insights, hotel management can proactively address concerns and optimize services.

This solution enables Cefalù Hotels to capture and analyze guest emotions, offering tailored experiences based on individual needs. The system integrates seamlessly with CRM platforms, ensuring scalability and modularity for different hotel operations. Hotel staff receive automated alerts when negative reviews are detected, allowing immediate resolution of guest complaints. Guests benefit from dynamic, AI-driven recommendations that refine their stay preferences based on past interactions.

With features such as sentiment classification, real-time notifications, preference-based analysis, and a user-friendly Streamlit UI, this project bridges the gap between technology and hospitality. The recommendation system employs content-based filtering and collaborative filtering to provide activity suggestions tailored to each guest. Additionally, automated email and Slack notifications keep the management informed about critical feedback, helping them take swift action.By integrating advanced AI capabilities into guest services, Cefalù Hotels can elevate customer satisfaction, streamline operations, and maintain a high standard of hospitality excellence.

**Product Owner:** Arya Dileep Kumar

**Team:**Arya Dileep Kumar

**Stakeholders:** Cefalu Hotel Management

### **Epic 1: Setting up the Local Environment and preparing the dataset**

**User Story 1.1 - Environment Setup:** As a developer, I want to set up a local environment with all necessary dependencies so that I can develop and test the application efficiently.

**User Story 1.2 - Guest Database preparation:** Fetch, preprocess and clean the dataset for sentiment analysis, personalized recommendations and enhance the guest experience.  
**Acceptance Criteria:** Installed Python, virtual environment, required libraries, and database setup.  
**Status:** Completed  
**Key Tasks:**

* Install Python and necessary libraries
* Set up a virtual environment
* Configure SQLite database
* Verify installation by running a sample script

### **Epic 2: Feedback Analysis and Sentiment Classification**

**User Story 2.1 - Feedback Processing:** As a hotel management team, I want to analyze customer feedback so that I can improve services based on customer sentiment.  
**Acceptance Criteria:** Process user feedback, classify sentiment (Positive, Neutral, Negative), and generate automated responses.  
**Status:** Completed  
**Key Tasks:**

* Collect user feedback through the UI
* Implement sentiment analysis using LLM
* Store feedback and sentiment data in the database
* Display results in UI with an appropriate response

### **Epic 3: Recommendation System Development**

**User Story 3.1 - Personalized Guest Recommendations**As a guest, I want to receive personalized recommendations based on my preferences so that I can have a better stay experience.  
**Acceptance Criteria:** Generate content-based and collaborative recommendations based on guest activity history and preferences.  
**Status:** Completed  
**Key Tasks:**

* Implement TF-IDF and cosine similarity for content-based filtering
* Use activity table for collaborative filtering
* Integrate recommendations into the UI with real-time updates
* Ensure recommendation logic adapts dynamically to user interactions

### **Epic 4: Alert System (Email & Slack)**

**User Story 4.1 - Department Alert System**As a hotel administrator, I want to receive alerts for negative feedback so that I can address guest concerns promptly.  
**Acceptance Criteria:** Send alerts via email and Slack when negative feedback is detected.  
**Status:** Completed  
**Key Tasks:**

* Implement email alert functionality
* Integrate Slack webhook for automated alerts
* Extract department responsible for the issue using LLM
* Ensure notifications contain guest ID, review, and suggested resolution

## **Sprint Plan**

| **Sprint** | **Goal** | **Key Tasks** | **Deliverables** |
| --- | --- | --- | --- |
| **Sprint 1** | Local env setup and data collection | i) Install Python, set up virtual environment ii) configure database, collect sample data | i) Environment setup ii) Initial dataset |
| **Sprint 2** | Feedback Analyzer | i) Implement sentiment analysisii)store feedback in the database iii) display sentiment in UI | i)Sentiment analysis module ii) Department responsible for negative review |
| **Sprint 3** | Develop recommendation system and Dynamic guest profile management system | i) Implement TF-IDF, cosine similarity, collaborative filtering and content based filteringii) Integrate with UI iii) Log Ad interaction and sentiment in database in real-time | i) Recommendation engine ii) Real-time logging of guest Interaction and data |
| **Sprint 4** | Add email and Slack alerts | i) Integrate slack and email notification system, extract responsible departmentii) test alerts | i) Fully functional alert system |

## 

## **Testing Plan**

**Unit Testing:** Each module (sentiment analysis, recommendations, alerts) will be tested individually to ensure correct outputs.  
**Integration Testing:** Ensure all components work together without errors, including UI, database, and backend interactions.  
**Performance Testing:** Measure the response time of the recommendation engine and alert system under varying loads.  
**User Testing:** Gather feedback from test users to refine recommendation quality and UI experience.

## **Key Metrics**

* **Recommendation Accuracy:** Precision, Recall, and F1 Score of personalized recommendations.
* **Sentiment Classification Accuracy:** F1 Score of sentiment predictions.
* **Profile Update Latency:** Average time to update guest preferences based on interactions.
* **Feedback Processing Time:** Time taken to analyze feedback and trigger alerts.
* **Alert Response Time:** Time taken for email and Slack alerts to be sent after detecting negative feedback.