TED UNIVERSITY

CMPE 491

AI-driven Review Aggregator and Analyzer for Strategic Product Insights
High-Level Design Report

Submission Date:
25.05.2024

Team Members:

Harun UĞURLU
Murat Efe DOĞAN
Ezgi Gülce YAZICI

Supervisor:

Dr. Venera ADANOVA

Jury Members:

Prof. Dr. Müslim BOZYİĞİT
Dr. Emin KUĞU

Table of Contents

1. Introduction

1.1 Purpose of the System

The purpose of this system is to provide a solution for ensuring customer experience. In today's digital world, customer experience and feedback are more important than ever. For a business to become successful, listening to customer feedback and ensuring customer reviews is a must. Consumers, with access to a wealth of information in today's digital age, are likely to research products or services online and read reviews they are considering buying. Therefore, it is vital that businesses actively monitor customer reviews to deliver the best customer experience [1].

The system will gather online customer reviews/feedback, analyze them, and provide insights in a user-friendly dashboard based on the analysis results by utilizing web scraping techniques, natural language processing, and data visualization

1.2 Design Goals

The design of the proposed system will be guided by several factors:

- **Scalability:** The system will be expected to handle increasing volumes of data without breaking down by scaling seamlessly.
- **Reliability:** The system will be expected to maintain a minimal downtime and ensure high availability and handle system failures.
- **Maintainability:** The system will be designed in a modular sense to ensure the components are isolated from each other, ensuring easy updates/modifications to them without affecting each other.
- **Performance:** The system will be implemented efficiently, ensuring high-performance response times and throughput.
- **Usability:** The system will provide a user-friendly, intuitive, and accessible dashboard.
- **Security:** The system will provide secure authentication and authorization mechanisms, and will keep sensitive data encrypted both at rest and in transit.

1.3 Overview

This document outlines the architecture and design considerations for our project. It includes an introduction, the system's purpose, design goals, and key terms, followed by an overview of the current architecture. The proposed software architecture section details the new design, subsystem decomposition, hardware/software mapping, data management, security measures, global control mechanisms, and boundary conditions. Additionally, the document specifies subsystem services, provides a glossary for clarity, and includes references. This overview ensures that all stakeholders understand the system's design and operational requirements.

2. Current Software Architecture

Since the project is a new development, there is no existing design to build upon or replace. We will design the system considering the functional and non-functional requirements, making sure it satisfies them.

3. Proposed Software Architecture

3.1 Overview

The system architecture for gathering, analyzing, and presenting the customer reviews is divided into three main services: frontend for the dashboard, backend for core business logic, data management, gathering the reviews using web scraping, and communication with other services (including the AI service), AI service for review analysis, a NoSQL database to store raw, unprocessed data like customer reviews, and a relational database to store everything else such as customer review analysis results, tenant information, product information, etc.. Each service has distinct responsibilities and they interact with each other to achieve the main objective of providing insights to businesses.

Major Components:

- Frontend
- Backend (with Web Scraping)
- AI Service
- Databases

Data Flow:

The data flow of the system follows the following steps:

1. **Web Scraping (Backend)**
    - The backend periodically triggers the Web Scraping tasks to collect reviews from the online sources of tenant's choice.
    - Web Scraping modules will collect the related pages over the internet, extract the reviews and related data such as review date, rating, etc. from them and store the raw reviews in a NoSQL database.

2. **Data Processing (Backend)**
    - A data processing job fetches the unprocessed reviews from the NoSQL database and sends them to the AI service.
    - The AI service performs the processing operations such as cleaning, removing noise, transforming, and normalizing the data.
    - Then the AI service sends the processed data back to the backend.
    - The backend service inserts the processed data into the relational database.
    - Once the data is processed, the related data state in the NoSQL database is updated to indicate it is processed.

3. **NLP Analysis (AI Service)**
    - The backend fetches the processed review data and sends them to the AI Service.
    - The AI Service performs sentiment analysis, extracts the most liked and disliked features, and calculates an overall score for the product.
    - After the analysis is done, the results are returned back to the backend and stored in the relational database.

4. **Data Presentation and Visualization (Frontend)**
    - The frontend requests the related data from the backend.
    - The backend queries the data from the relational database and sends it back to the frontend.
    - The frontend then displays the reviews, overall score and charts.

3.2 Subsystem Decomposition

The system is decomposed into modular subsystems to ensure scalability, maintainability, and separation of concerns. Each subsystem has a discrete responsibility and together, they handle the gathering, processing, analyzing, and presenting processes.

1. **Web Scraping Subsystem (Backend Service)**

   Purpose: The Web Scraping subsystem handles the gathering of customer reviews from various online sources.

   Components:

   Scraper Module: A scraper module will handle the gathering of customer reviews from the selected platforms. The scraper module will be flexible and configurable for different platforms. It will be triggered periodically.
   Data Extractor: This module will extract the relevant data from the scraped raw HTML page.

   Interactions:

   Scheduler Triggering: The Web Scraper module will be periodically triggered by the Scheduler Module in the Backend service and start scraping the reviews from the selected platforms.
   Backend Fetch Unprocessed Reviews Triggering: The web Scraping subsystem will write a queue message that will trigger the backend to retrieve newly scraped customer reviews from the NoSQL db.
   Storing the Data: The Web Scraper module will store the raw review data in the NoSQL database after it extracts the relevant information from the raw HTML.

2. **Scheduler Subsystem (Backend Service)**

   Purpose: Scheduler subsystem triggers the tasks that need to be done periodically by writing messages into queue storage systems.

   Components:

   Web Scraper Scheduler: This scheduler component will trigger the Web Scraping subsystem by writing a new message into the queue storage for it to scrape the newest customer reviews from the selected platforms.
   Backend Fetch Unprocessed Reviews Scheduler: This scheduler component will trigger the Backend service to fetch the new customer reviews that the Web Scraper has scraped from the NoSQL database.
   AI Service Scheduler: This scheduler will send the processed data to the AI Service and thus triggering the AI analysis process.

3. **Data Processing Subsystem (AI Service)**

Purpose: To clean, transform, and normalize raw review data in order to transform it into a suitable format for the AI model.

Components:

Data Fetcher: Retrieves the unprocessed, raw reviews from the NoSQL database.
Data Cleaner Removes noise, corrects typos if there are any, and standardizes data formats if necessary.
Data Transformer: Structures data into a format suitable for the relational database schema. For example ensuring the dates are in the same format (e.g., 'YYYY-MM-DD').
Data Sender: Sends the processed data back to the backend.

Interactions:

Receives Data: This subsystem interacts with the backend to receive raw data from the NoSQL database.
Sends Data: This subsystem sends the processed data back to the backend for storage in the relational database.

4. **NLP Analysis Subsystem (AI Service)**

Purpose: To perform sentiment analysis and extract key insights from the processed review data.

Components:

Sentiment Analyzer: Uses NLP techniques to determine the sentiment of each review and categorize the review as positive or negative.
Feature Extractor: It identifies the most liked and disliked features of the product from the customer reviews.
Scoring Module: The subsystem calculates an overall score for the product based on the number of positive and negative reviews.
API Interface: Receives processed customer review data from the backend and returns analysis results.

Interactions:

Receives Data: This subsystem interacts with the Backend to receive processed data for analysis and to send back the analysis results.
Sends Data: Sends the analyzed data back to the backend for storage in the relational database.

5. **Data Management Subsystem (Backend Service)**

   Purpose: To manage the storage and retrieval of raw and processed data.

   Components:

   NoSQL Database: This database stores the raw, unprocessed customer review data after it is scraped.
   Relational Database: It stores processed review data, analysis results, tenant information, and product information.
   Data Access Layer: Manages database connections and queries that interact with the database.

   Interactions: This subsystem interacts with all other subsystems to ensure data is stored and retrieved efficiently and consistently.

6. **Data Presentation and Visualization Subsystem (Frontend)**

   Purpose: To provide an intuitive and interactive user interface for displaying review data and insights.

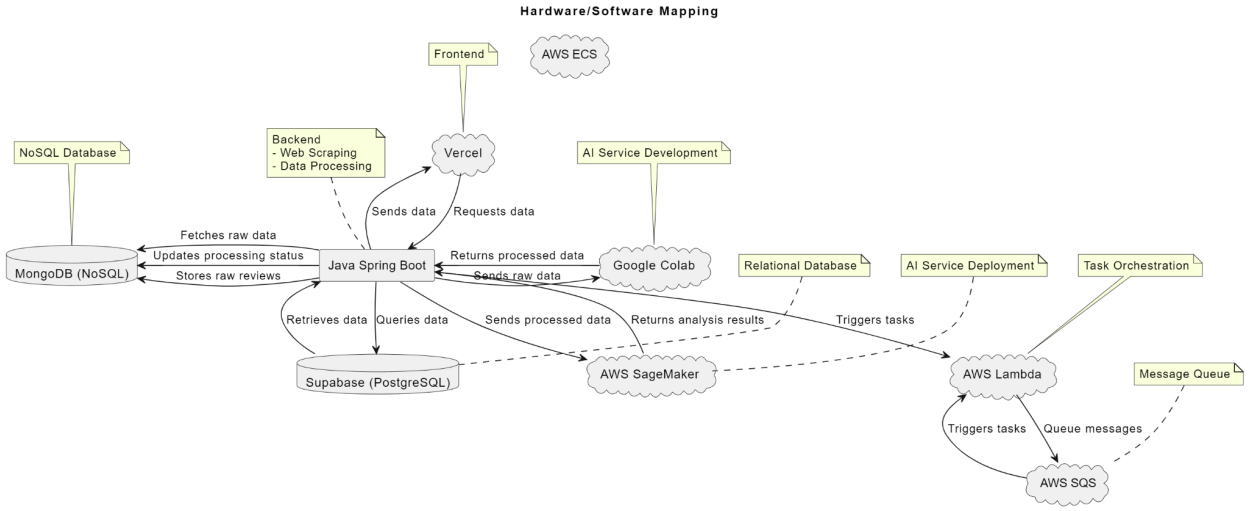
   Components:

   Dashboard Interface: The main user interface for displaying data. The users will be able to see the analysis results, charts, and reports.
   Visualization Tools: Generates charts, graphs, and other visual representations of the data.
   Export Data Buttons: Users will use these buttons to export the data into forms such as xls, pdf.
   Filtering Section: Will provide filtering options such as show only specific platforms, show positive reviews or negative reviews
   API Client: Communicates with the backend to fetch data.

   Interactions: This subsystem interacts with the Backend service to request and receive processed data and analysis results.

3.3 Hardware/Software Mapping

This section outlines the hardware and software resources used to implement the components of the system. The system leverages cloud-based solutions for flexibility and performance.



Deployment Strategy:

Backend (Including Web Scraper):

Hosting Platform: AWS Elastic Container Service (ECS)

Technologies: Java Spring Boot

Purpose: To handle core business logic, data management, web scraping, and communication with other services including the AI service.

Deployment Details:

- **AWS ECS:** The backend services are containerized and deployed on AWS ECS, allowing for easy scaling, management, and orchestration of containers.

Data Storage:

Relational Database: Supabase (PostgreSQL)

NoSQL Database: MongoDB

Purpose: To store processed review data, analysis results, tenant information, product information, and raw unprocessed review data.

Deployment Details:

- **Supabase:** A fully managed PostgreSQL database service that provides real-time capabilities and easy integration with the backend. It will handle structured data, ensuring reliable storage and fast queries for processed data and other relational data.
- **MongoDB:** A NoSQL database used to store raw, unprocessed review data. It offers flexibility in handling semi-structured data and scales well with large volumes of data, making it ideal for storing raw reviews collected from web scraping.

AWS Lambda Functions (Scheduler Subsystem)

Purpose: To create queue-triggered functions for orchestrating data flow and task scheduling.

Technologies: AWS Lambda, AWS SQS (Simple Queue Service)

Deployment Details:

- **AWS Lambda:** They are serverless functions that are triggered by messages in the AWS SQS queue, handling the execution of specific tasks such as triggering web scraping, data processing, and NLP analysis.
- **AWS SQS:** It manages the message queues that trigger the Lambda functions, ensuring reliable task scheduling and execution.

AI Service:

Platforms: Google Colab will be used in the development of the model, and AWS SageMaker for its deployment.

Technologies: Python

Purpose: It will clean, transform, and normalize the raw review data, preparing it for the AI model which will perform sentiment analysis and extract the key features the customer reviews are referring to.

Deployment Details:

- **Google Colab:** It is used for developing and training the initial AI model. It provides a Python notebook with access to powerful GPUs, which makes it an ideal environment for model training.
- **AWS SageMaker:** The trained models are going to be deployed on AWS SageMaker. It provides a managed service for deploying machine learning models. It handles the operations such as running the models, scalability, monitoring, and maintenance.

Frontend:

Hosting Platform: Vercel

Technologies: React.js, TypeScript, Nextjs

Purpose: To provide an intuitive and interactive user interface for displaying review data and insights.

Deployment Details: The frontend is deployed on Vercel, which offers a serverless platform optimized for static site generation and front-end frameworks. This ensures quick deployment, scalability, and efficient delivery of static assets.

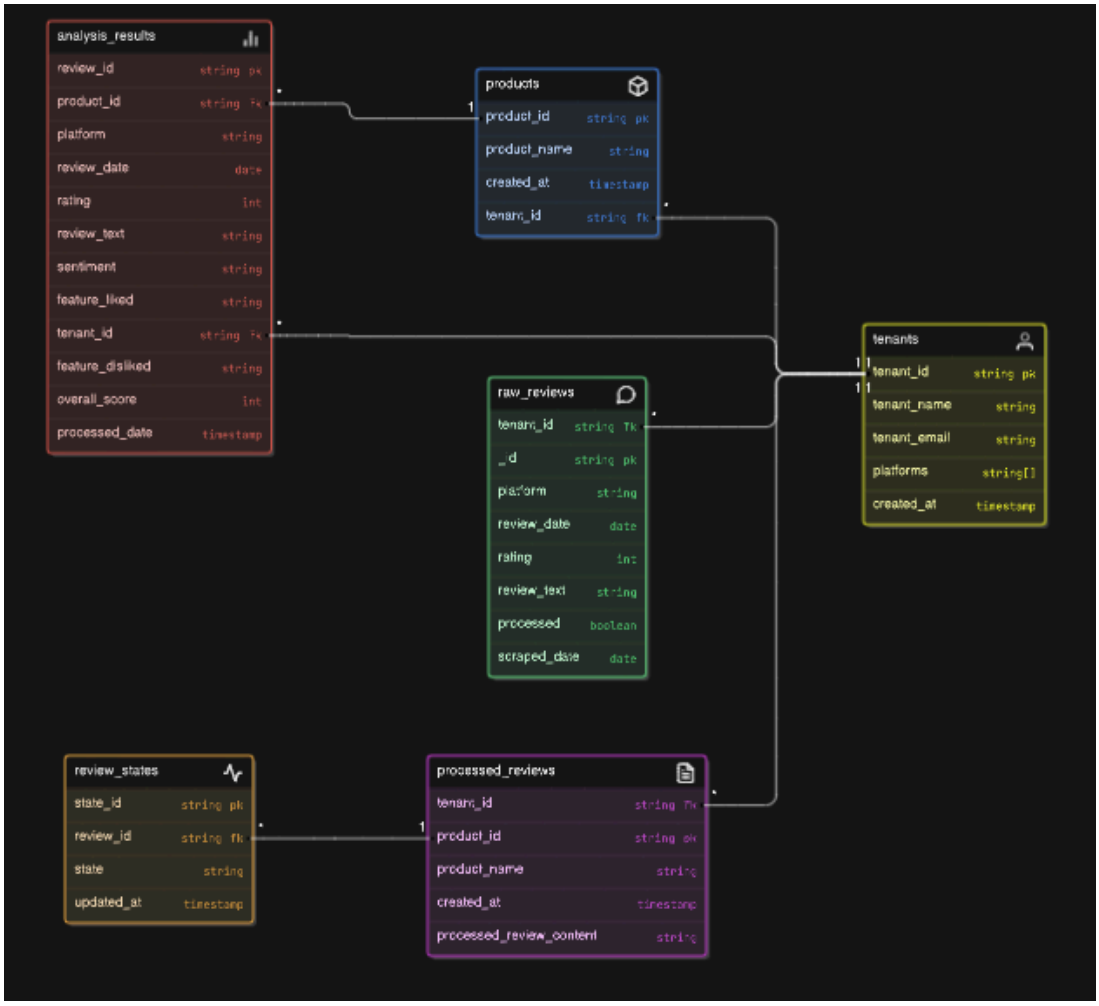
Third-Party Services:

- **NLP Libraries:** Libraries such as NLTK, SpaCy, and Hugging Face Transformers will be utilized for natural language processing tasks.
- **Monitoring and Logging:** Sentry will be used to monitor application performance and log errors or anomalies, providing real-time error tracking and insightful analytics.

3.4 Persistent Data Management

Persistent Data refers to data stored to ensure its availability over time. In our project, persistent data includes customer reviews, results from natural language processing (NLP), and other relevant insights.

Data Management Strategy:



DB Schema:

Table: raw_reviews

- **_id**: Unique identifier for each review (automatically generated, primary key).
- **tenant_id**: Identifier for the tenant (foreign key referencing tenants).
- **platform**: Source of the review (e.g., Amazon, Yelp).
- **review_date**: Date the review was posted.
- **rating**: Numerical rating given by the reviewer.
- **review_text**: Text content of the review.
- **processed**: Flag indicating whether the review has been processed (true/false).
- **scraped_date**: Date the review was scraped.

Relational Database (Supabase PostgreSQL):

Table: tenants

- **tenant_id**: Unique identifier for each tenant (primary key).
- **tenant_name**: Name of the tenant.
- **tenant_email**: Email address of the tenant.
- **platforms**: List of selected platforms to scrape the reviews from.
- **created_at**: Timestamp when the tenant was created.

Table: products

- **product_id**: Unique identifier for each product (primary key).
- **tenant_id**: Identifier for the tenant (foreign key referencing tenants).
- **product_name**: Name of the product.
- **created_at**: Timestamp when the product was created.

Table: analysis_results

- **review_id**: Unique identifier for each processed review (primary key).
- **tenant_id**: Identifier for the tenant (foreign key referencing tenants).
- **product_id**: Identifier for the product (foreign key referencing products).
- **platform**: Source of the review.
- **review_date**: Date the review was posted.
- **rating**: Numerical rating given by the reviewer.
- **review_text**: Text content of the review.
- **sentiment**: Sentiment of the review (e.g., positive, negative).
- **feature_liked**: Features liked by the reviewer.
- **feature_disliked**: Features disliked by the reviewer.
- **overall_score**: Overall score calculated for the product.
- **processed_date**: Timestamp when the review was processed.

Table: processed_reviews

- **product_id**: Unique identifier for each product (primary key).
- **tenant_id**: Identifier for the tenant (foreign key referencing tenants).
- **product_name**: Name of the product.
- **created_at**: Timestamp when the product was created.
- **processed_review_content**: Text content of the processed review

Table: review_states

- **state_id**: Unique identifier for each review state (primary key).
- **review_id**: Identifier for the processed review (foreign key referencing processed_reviews).
- **state**: Current state of the review (e.g., raw, processed).
- **updated_at**: Timestamp when the review state was last updated.

Data Storage Solutions:

- **Primary Storage:** Supabase will be used for relational data storage, to ensure data integrity and ease of access.
- **Secondary Storage:** MongoDB will be used for storing unstructured data, providing flexibility and scalability.

Data Backup and Recovery Plans:

- **Backups:** Automated daily backups of Supabase instances and regular snapshots of MongoDB data.
- **Recovery:** Automated scripts will restore data from backups in case of failure. Regularly testing backup and recovery procedures will ensure data integrity and availability.

Data Processing and Access:

- **Storage:** Raw reviews will initially be stored in the MongoDB database after being scraped. Then, the processed (cleaned and transformed) data and NLP results will be stored in the relational Supabase database for easy querying.
- **Processing:** The processed data will be analyzed using the AI Model, deployed on the AWS SageMaker.
- **Access:** APIs will provide secure access to stored data, allowing for querying and retrieving processed insights. The dashboard will fetch data via these APIs to present real-time insights to users.

3.5 Access Control and Security

Access Control and Security measures ensure that only authorized users can access the system and that data remains secure during storage and transmission.

Authentication and Authorization Mechanisms:

Authentication:

- **Supabase Auth:** This service will handle user authentication, providing secure and scalable authentication mechanisms.
    - **Multi-Factor Authentication (MFA):** MFA will be supported to add an extra layer of security.
    - **Credentials:** Users will sign in using their username and password, which are securely managed and encrypted by Supabase Auth.

Authorization:

- **Role-Based Access Control (RBAC):** Roles such as Admin, Analyst, and Viewer will be defined, each with specific permissions to ensure users have access only to necessary resources.

Data Encryption:

- **At Rest:**
    - **Supabase Storage Encryption:** Data stored in Supabase databases will be encrypted using robust encryption mechanisms.
    - **MongoDB Encryption:** Data stored in MongoDB will be encrypted at rest to ensure data security.
- **In Transit:**
    - **SSL/TLS:** Secure data transmission between clients and servers will be ensured using SSL/TLS protocols.
    - **HTTPS:** All API communications will enforce HTTPS to maintain data integrity and confidentiality during transmission.

Additional Security Measures:

- **Logging and Monitoring:** Sentry will be used to monitor application performance and log errors or anomalies, providing real-time error tracking and analytics.
- **Regular Security Audits:** Periodic security reviews and audits will be conducted to identify and address potential vulnerabilities in the system.
- **Compliance:** Compliance with relevant data protection regulations, such as GDPR, will be applied by implementing appropriate data handling and processing practices. Regular reviews and updates to compliance policies will be carried out to adapt to changing regulations.

3.6 Global Software Control

The global software control for our system is designed to ensure efficient handling of requests and smooth synchronization between subsystems. Here's how it is implemented:

1. **Request Initiation:** Requests originate from the front end when a product owner searches for customer feedback on a specific product. These requests are then sent to the backend server via REST APIs.
2. **Subsystem Synchronization:**
    - **Web Scraping Module:** Upon receiving a request, the backend initiates the web scraping process to gather customer reviews from various online sources. This module runs asynchronously to handle multiple scraping tasks concurrently, leveraging task queues and worker threads to manage load and efficiency.
    - **AI Analysis Module:** Once the data is scraped, it is passed to the AI service for analysis. The AI module uses natural language processing (NLP) techniques to analyze the sentiment of the reviews and identify key features mentioned by customers. The processed data is then stored in the appropriate databases.
    - **Data Storage:** Initial raw data is stored in a NoSQL database due to its unstructured nature, while the analyzed and structured data is stored in a relational database for efficient querying and reporting.

3. **Synchronization and Concurrency Issues:**
    - **Data Consistency:** To maintain data consistency, the system uses transaction management and locking mechanisms, especially when writing analyzed data to the relational database.
    - **Concurrency Control:** The use of task queues and worker threads helps manage concurrent requests. Additionally, rate limiting is applied to the web scraping module to avoid overloading source websites and to comply with their usage policies.
    - **Synchronization:** A message queue system is used to coordinate tasks between the web scraping and AI analysis modules, ensuring that data flows smoothly from scraping to analysis without bottlenecks.

3.7 Boundary Conditions [2]

Boundary conditions describe the startup, shutdown, and error behaviors of the system.[1] These conditions are critical to ensuring the reliability and robustness of the system throughout its lifecycle.

Initialization

1. System Startup:
    - The system initializes by starting all major components sequentially: Backend, AI Service, and Frontend.
    - During the startup, the backend service will connect to both the NoSQL database (for customer reviews) and the relational database (for user data, processed insights, and configurations).

2. Data Access at Startup:
    - At startup, the backend service retrieves configuration data such as scraping schedules, user preferences, and pre-defined NLP models from the relational database.
    - The AI service loads pre-trained NLP models and initial datasets required for sentiment analysis and feature extraction.

3. Service Registration:
    - The backend registers with a service discovery mechanism, allowing other components to locate it.
    - The AI service registers itself with the backend to be used for processing requests.

4. User Interface Initialization:
    - The frontend service initializes by loading essential UI components and fetching initial data like dashboard configurations and user-specific settings.
    - The user interface presents the login screen. Once the user is logged in, users are directed to the main dashboard, which displays a summary of recent analyses and any pending notifications.

Termination

1. Subsystem Termination:
    - Individual subsystems can be terminated independently. For instance, the web scraping module can be stopped for maintenance without affecting the AI analysis or the frontend dashboard.
    - The termination of any subsystem will trigger notifications to other subsystems. For example, if the web scraping service stops, the backend will log this event and notify the frontend to display an appropriate message to the user.

2. Notification on Subsystem Termination:
    - The backend service handles notifications for terminated subsystems and logs these events.
    - The frontend service is updated in real-time about the status of backend services and displays messages or warnings to users accordingly.

3. Database Communication:
    - Local updates during subsystem termination are synchronized with the databases. For example, if the AI service is terminated, it ensures that any intermediate analysis results are saved to the relational database.
    - The backend manages the database connections and ensures data integrity during termination processes.

Failure

1. Node or Communication Link Failure:
    - In the event of a node failure, the system utilizes backup nodes and load balancing to maintain service availability.
    - The system architecture includes redundant communication links to ensure connectivity between the backend, AI service, and frontend.

2. System Recovery from Failure:
    - If the system detects a failure, the system follows a predefined recovery procedure which includes:
        I. Attempting to restart the failed node or service.
        II. Switching to a backup node or service if the restart fails.
        III. Notifying the system administrator of the failure and recovery actions taken.

3. Recovery vs. Initialization:
    - System recovery is similar to initialization but includes additional steps for data integrity checks and synchronization.
    - After recovery, the system verifies that all subsystems are in sync and that no data corruption occurred during the failure.

These boundary conditions ensure the smooth operation of the system, providing a reliable and user-friendly experience for product owners analyzing customer reviews.

5.Glossary

Web Scraping: Web scraping is an automatic method to obtain large amounts of data from websites.[3]
Natural Language Processing: Natural language processing, or NLP, combines computational linguistics—rule-based modeling of human language—with statistical and machine learning models to enable computers and digital devices to recognize, understand and generate text and speech.[4]
Frontend: The frontend of a software program or website is everything with which the user interacts.[5]
Backend: The back end refers to parts of a computer application or a program's code that allow it to operate and that cannot be accessed by a user. Most data and operating syntax are stored and accessed in the back end of a computer system.[6]
Artificial Intelligence (AI): Artificial intelligence, or AI, is technology that enables computers and machines to simulate human intelligence and problem-solving capabilities.[7]
NoSQL: NoSQL databases are non-tabular databases and store data differently than relational tables.[8]
Noise: Noise refers to any content within the review text that is not relevant to the actual review or is unnecessary for the analysis.
Transforming: To ensure that the cleaned data is structured in a way that aligns with the schema of the relational database.
HTML Page: A HTML (Hypertext Markup Language) page is a document designed for display on the World Wide Web.
Queue Storage: Storage is used to decouple and scale microservices, distributed systems, and serverless applications. It allows different components of a system to communicate asynchronously.
Tenant: A group or entity that shares access to the same instance of the software or application
Relational Database: A relational database is a type of database that stores and provides access to data points that are related to one another.[9]
Sentiment Analysis: Sentiment analysis is an application of natural language processing (NLP) technologies that train computer software to understand text in ways similar to humans.[10]
UI (User Interface): UI stands for user interface. It is the point of contact between humans and computers.[11]

Scalable: In software, scalability means a system's ability to handle varying workloads as the needs of your business change.[12]
Separation of Concerns: Separation of concerns is a design principle in software engineering where code is divided into distinct sections, each addressing a separate concern or aspect of functionality.[13]
Stakeholders: Stakeholders are individuals, groups, or entities with a vested interest or concern in a particular project, initiative, or organization.
Cloud-Based Solution: A cloud-based solution provides a service by leveraging the power of the cloud and cloud-based infrastructure to meet business needs, often with greater operational efficiency and cost effectiveness.[14]

6. References

[1]: Patil, D. R., & Rane, N. L. (2023). Customer experience and satisfaction: Importance of customer reviews and customer value on buying preference. *International Research Journal of Modernization in Engineering Technology and Science, 5*(3), 3437. https://www.irjmets.com

[2]: Bruegge, B., & Dutoit, A. (n.d.). System design. Technical University of Munich. Retrieved from https://ase.in.tum.de/tramp.globalse.org/doc/presentations/system_design2.pdf

[3]: GeeksforGeeks. (n.d.). What is web scraping and how to use it? Retrieved from https://www.geeksforgeeks.org/what-is-web-scraping-and-how-to-use-it/

[4]: IBM. (n.d.). Natural language processing (NLP). Retrieved from https://www.ibm.com/topics/natural-language-processing

[5]: Per C. (2020, April 18). Frontend. TechTerms. https://techterms.com/definition/frontend

[6]: Fitzgibbons, L. (2019, May). Front end. TechTarget. https://www.techtarget.com/whatis/definition/front-end

[7]: IBM. (n.d.). Artificial intelligence. Retrieved May 25, 2024, from https://www.ibm.com/topics/artificial-intelligence

[8]: MongoDB. (n.d.). NoSQL explained. Retrieved May 25, 2024, from https://www.mongodb.com/resources/basics/databases/nosql-explained

[9]: Oracle. (n.d.). What is a relational database? Retrieved May 25, 2024, from https://www.oracle.com/database/what-is-a-relational-database/

[10]: Amazon Web Services. (n.d.). What is sentiment analysis? Retrieved May 25, 2024, from https://aws.amazon.com/tr/what-is/sentiment-analysis/#:~:text=Sentiment%20analysis%20is%20an%20application,before%20providing%20the%20final%20result.

[11]: Coursera. (n.d.). UI design. Retrieved May 25, 2024, from https://www.coursera.org/articles/ui-design

[12]: MadAppGang. (n.d.). Scalability in software development: A 101 guide for businesses. Retrieved May 25, 2024, from https://madappgang.com/blog/scalability-in-software-development/#:~:text=In%20software%2C%20scalability%20means%20a,needs%20of%20your%20business%20change.

[13]: Wikipedia contributors. (n.d.). Separation of concerns. In Wikipedia, The Free Encyclopedia. Retrieved May 25, 2024, from https://en.wikipedia.org/wiki/Separation_of_concerns#:~:text=In%20computer%20science%2C%20separation%20of,code%20of%20a%20computer%20program.

[14]: Mitel. (n.d.). What is a cloud-based solution? Retrieved May 25, 2024, from https://www.mitel.com/articles/what-is-a-cloud-based-solution