



**VCU** College of Engineering

CS 25-349:

AI-Powered Email Response System  
Using Fine-Tuned LLMs for Customer  
Service in React

**Preliminary Design Report**

Prepared for  
Keroles Hakem, CoStar Group

By  
Cameron Clyde, Emma Smith,  
Angela Harris, Sohil Marreddi

Under the supervision of  
Preetam Ghosh

12/9/2024

## **Executive Summary**

As CoStar Group continues to grow, the demands on its customer service teams increase significantly. CoStar Group's internal customer relationship management system is known as Case Management. Case Management employees are responsible for handling complex cases and responding to customer emails. Balancing these responsibilities often results in delayed responses and inconsistent communication quality, as employees struggle to maintain both professionalism and personalization in their emails. Furthermore, replying to emails takes away time employees have to deal with their cases.

These challenges highlight the need for a solution that streamlines the email process, allowing employees to respond quickly and efficiently without compromising the quality of service. This project will leverage Large Language Models (LLMs) to assist in generating quality email responses for Case Management Employees. The overall aim of this project is to create a web application with a fine-tuned LLM on specific customer service data to generate relevant and context-aware email responses. The project goals include implementing a feedback loop to continuously improve the model's performance, allowing employees to review and edit generated drafts before sending them to customers and reducing the time required to respond to customer emails.

The design specifications and constraints outline a system that facilitates the generation of professional email responses for open customer service cases, integrating the React library for frontend development and utilizing FastAPI and PostgreSQL for backend functionality. The key requirements that will be implemented in this design include a feedback loop for AI suggestions, JWT-based authentication, AI response templates that will be customizable for the user, storage management for data related to case assignment logic, AI-generated responses, and representative feedback, and continuous integration and deployment pipelines for automated testing and updates. A successful design will effectively reduce the workload for customer service representatives, handle an increasing number of customer service cases and interactions, minimize customer service response times, and enhance customer satisfaction. In the creation of this design, we will adhere to a budget of \$1000, comply with data privacy regulations, and take necessary security measures against vulnerabilities that can present themselves.

In order to meet the necessary design requirements, it's important that we are always applying a set of codes and standards that will ensure quality, reliability, and safety. In this document, a number of codes and standards relating to security, privacy, software development, quality, ethics, design, etc. are given that can be directly applied to our design. Two codes relating to security and privacy that we will be adhering to in our design are the GDPR (General Data Protection Regulation) and CCPA (California Consumer Privacy Act). We will also be applying multiple standards, including ISO/IEC 27701, relating to information security management, ISO/IEC 25010, which outlines the software quality requirements and evaluation, OpenAPI, the standard for API documentation, ISO/IEC TR 24027, the guidelines for evaluating AI-based systems, Material Design Guidelines, which describes the best practices for UI design using Material UI components, etc. With the usage of these codes and standards, we can be sure to create the most efficient and secure design.

The scope of the project is clearly defined to ensure all objectives are met on time and within budget. The web application, CASEflow, will be developed using an Agile methodology with sprints spanning 2-3 weeks each. Deliverables include a fully functional web application that allows employees to log in, access customer emails, and generate AI-assisted responses, fine-tuned on CoStar's customer service data. Key features include an interface for employees to review and edit AI-generated responses, a feedback loop to improve the AI's performance over time, and detailed metrics for response times and model accuracy improvements. Academic deliverables include the project proposal, reports, and presentations required for the Capstone EXPO.

The project will utilize resources such as GitHub for version control, React for frontend development, FastAPI for backend operations, and PostgreSQL for database management. The large language model (LLM) will be fine-tuned using AWS SageMaker, which will be provided by CoStar. The team will focus on continuous integration to ensure a smooth development process, with testing and feedback incorporated at each sprint to avoid scope creep and ensure all deliverables are met. The application aims to streamline email response times, reduce employee workload, and improve the quality and consistency of customer interactions.

## **Table of Contents**

Section A. Problem Statement	5
Section B. Engineering Design Requirements	6
B.1 Project Goals (i.e. Client Needs)	6
B.2 Design Objectives	6
B.3 Design Specifications and Constraints	7
B.4 Codes and Standards	8
Section C. Scope of Work	10
C.1 Deliverables	10
C.2 Milestones	11
C.3 Resources	12
Section D. Concept Generation	13
Section E. Concept Evaluation and Selection	17
Section F. Design Methodology	19
F.1 Computational Methods (e.g. FEA or CFD Modeling, example sub-section)	19
F.2 Experimental Methods (example subsection)	20
F.3 Validation Procedure	20
Section G. Results and Design Details	22
G.1 Modeling Results	22
G.2 Experimental and Testing Results	24
G.3. Final Design Details/Specifications	25
Section H. Societal Impacts of Design	30
H.1 Public Health, Safety, and Welfare	30
H.2 Societal Impacts	31
H.4. Economic Impacts	31
H.7. Ethical Considerations	31
Section I. Cost Analysis	33
Section J. Conclusions and Recommendations	34
Appendix 1: Project Timeline	36
Appendix 2: Team Contract (i.e. Team Organization)	37
References	41

## Section A. Problem Statement

CoStar Group is the global leader in the digitalization of commercial real estate. Since their start in 1986, they have expanded into the world of residential real estate and become a leader in this industry. Their mission is to digitize the world's real estate and make it easier for people and companies to discover properties. They provide real estate information, analytics, and online marketplaces (CoStar Group, 2024). In their mission to digitize real estate around the world, CoStar Group continues to grow every year. CoStar Group acquires new brands and companies to help achieve this mission. CoStar Group has over 25 brands, including Homes.com and Apartments.com.

Merging these businesses into CoStar Group can be difficult. Every company that CoStar Group acquires does their business differently. In particular, most companies have their own subscriptions with different Customer Relationship Management (CRM) systems. CRM systems are a strategic tool that integrate technology, customer knowledge and relationships to enhance business efficiency, foster customer loyalty, and drive economic growth (Gil-Gomez et al., 2020). It is evident that CRMs are essential, but CoStar Group does not want to deal with multiple different CRMs and pay fees for each CRM. To solve this, they decided to run it all internally and develop their own CRM, known as Case Management.

In CoStar's existing customer service response process, representatives begin by reading the incoming email along with the associated email history to fully understand the customer's issue. After reviewing the context, they select an appropriate response template from a set of predefined options that best aligns with the customer's inquiry or concern. The representative then customizes the chosen template by filling in relevant details specific to the case, such as the customer's name, case number, and any additional information that addresses the issue.

As the amount of cases increases, the current process begins to show limitations. This process becomes time consuming, labor intensive, and prone to human error as representatives are required to manually fill each template (Mesquita et al., 2022). The reliance on templates can result in repetitive and impersonal responses, which may not fully address each customer's issue. This adds additional time and effort as representatives must write their own reply. While technologies have been proposed to automatically query and select customer data to fill response templates, they are not completely accurate and still fail to address customer issues that are not covered by templates (Malik et al., 2007). As case volumes rise, it becomes increasingly difficult for employees to manage the workload effectively, leading to delayed responses, inconsistencies in communication, and reduced overall quality of customer service (Sheth et al., 2024). This issue creates the need for a more scalable solution that can maintain personalization without sacrificing speed or accuracy.

Our project aims to improve and streamline this process by developing an email response system that utilizes a fine-tuned large language model (LLM) to generate personalized responses. This system will analyze the incoming email and its history to understand the customer's issue, then automatically create a response without relying on premade templates. The LLM will use the context of the email history to create responses more tailored to the specific case, while automatically filling in relevant customer data, which increases both the speed and quality of customer service interactions.

## **Section B. Engineering Design Requirements**

### **B.1 Project Goals (i.e. Client Needs)**

This project aims to assist customer service employees at CoStar Group with the high-volume of emails received. The demands for customer service employees grow as CoStar Group acquires new businesses. It can be difficult for employees to manage their existing work on top of the many emails they receive. Responding to emails in a personalized and professional way is time consuming. When trying to balance work on existing complex cases, writing these emails can lead to inconsistencies. With this in mind, the project goals are as follows:

- To design an app that reduces the time required to respond to customer emails
- To enable employees to focus on high priority work by automating the generation of strong email replies
- To improve the personalization and professionalism of email responses from customer service employees
- To allow employees to review and edit generated drafts, ensuring accuracy and appropriateness
- To enhance the AI model through employee feedback to continuously improve the model's performance

### **B.2 Design Objectives**

The following are key objectives of the design:

- The design will reduce email response times for employees by generating draft replies. This will be measurable through employee feedback and satisfaction, with a target of reducing response time by 30%.
- The design will create a web app, with an authentication system to ensure only CoStar Group Case representatives can access this information. This structure will be completed halfway through the fall semester, with testing to ensure functionality as expected.
- The design will fine-tune a Large Language Model (LLM) using customer service data to generate relevant, context-aware email responses. The speed and accuracy will be measured before fine tuning. After fine-tuning, the speed and accuracy should improve by 20%.
- The design will implement a feedback loop where employees can rate the email responses produced by the model to improve the model. This will be measured by the model's ability to learn from feedback.
- The design will include a user interface allowing employees to review and edit responses for accuracy and appropriateness. This will be measurable by employee usage and edit rates.

### **B.3 Design Specifications and Constraints**

#### **Design Specifications**

##### *Functional Requirements*

- The system must be powered by fine-tuned LLMs
- Design must facilitate the generation of professional email responses for open customer service cases
- The system must integrate the React library for frontend development
- Design must integrate with existing customer service platforms
- Design must implement a feedback loop, allowing employees to rate AI-generated suggestions
- Design must integrate FastAPI and PostgreSQL for backend functionality
- Design must implement JWT-based authentication for user login and signup on the backend
- Design must implement login and signup pages on the frontend
- Design must be able to store data related to case assignment logic, AI-generated responses, and representative feedback on those responses
- Design must allow for customization of the AI response templates
- Design must follow *IEEE P7003 Algorithmic Bias Considerations* to ensure that AI-generated responses are free from bias and are transparent to both employees and customers
- The design must include unit tests, integration tests, and automated testing pipelines to follow ISO/IEC/IEEE 29119 software testing standards
- Design must implement continuous integration and deployment pipelines for automated testing and seamless updates (CI/CD best practices)

#### *Non-Functional Requirements*

- Design must effectively reduce the workload for customer service representatives
- The system should be designed to handle an increasing number of customer service cases and employee interactions
- The AI model must generate responses quickly to ensure minimal delays in customer service workflow
- The design must follow *ISO/IEC 25010* to ensure software quality, including functionality, usability, and maintainability
- The design must adhere to *WCAG 2.1 guidelines* to ensure that the interface is accessible to users with disabilities
- AI model must undergo regular evaluation to ensure that its suggestions are accurate, relevant, and helpful
- Design should aim to minimize response customer service response times
- Design must enhance customer satisfaction

#### **Design Constraints**

##### *Cost*

- Design must adhere to a budget of \$1000

##### *Security and Data Privacy*

- All user data must comply with data privacy regulations

- Design must implement data retention policies that comply with *GDPR/CCPA* standards to ensure user data is only stored for the required period
- Design must comply with *GDPR* guidelines, ensuring that user data is protected and the necessary opt-ins and transparency measures are in place
- Design must ensure protections against the top 10 web application vulnerabilities, including secure authentication, access control, and input validation (OWASP Top Ten)
- The system must follow standards for managing personally identifiable information (ISO/IEC 27701 Compliance)
- Design must securely store JWTs within the system

#### *Quality Assurance and Testing*

- Design must ensure that tests cover security vulnerabilities and performance bottlenecks, which limit the system's performance (ISTQB guidelines)
- Design must include robust error handling

### **B.4 Codes and Standards**

#### **Codes:**

##### *Security and Privacy*

- [GDPR](#) (General Data Protection Regulation): Data protection and privacy law for EU residents
- [CCPA](#) (California Consumer Privacy Act): Privacy law for California residents, focusing on data transparency and user control over personal data

#### **Standards:**

##### *Security*

- [OWASP Top Ten](#): Guidelines for mitigating common web vulnerabilities
- [NIST Cybersecurity Framework](#): Best practices for managing cybersecurity risks
- [ISO/IEC 27701](#): Information security management systems standard
- [JWT Best Practices](#): Guidelines for secure handling of JSON Web Tokens
- [CIS Benchmarks](#) (Center for Internet Security): Best practices for cloud infrastructure security

##### *Privacy*

- [ISO/IEC 27701](#): Privacy extension to ISO 27001, focusing on managing personal data (PII)

##### *Software Development Standards*

- [ISO/IEC 25010](#) (Software Quality Requirements and Evaluation): Software quality metrics, such as usability and reliability
- [IEEE 12207](#): Framework for software development lifecycle (SDLC)
- [SOLID Principles](#): Object-oriented design principles for maintainability



- [Agile Development Framework](#): Practices for iterative, adaptive development

#### *API and Web Standards*

- [RESTful API Best Practices](#): Guidelines for designing RESTful APIs (proper use of HTTP methods, statelessness)
- [OpenAPI](#): Standard for API documentation
- [W3C Web Standards](#): Accessibility, responsive design, and cross-browser compatibility
- [WCAG 2.1](#) (Web Content Accessibility Guidelines): Guidelines for designing accessible web content

#### *AI Ethics and Guidelines*

- [ISO/IEC TR 24027](#): Guidelines for evaluating AI-based systems
- [IEEE P7003 Algorithmic Bias Considerations](#): Recommendations for avoiding bias in AI systems

#### *Quality Assurance Standards*

- [ISTQB](#) (International Software Testing Qualifications Board): Best practices for software testing
- [ISO/IEC/IEEE 29119](#): Software testing standards for comprehensive testing
- [CI/CD Best Practices](#): Guidelines for automate testing, deployment, and version control

#### *Frontend UI/UX Design Standards*

- [Material Design Guidelines](#): Best practices for UI design using Material UI components
- [User-Centered Design](#) (UCD): Focus on user needs during the design process ([ISO 9241-210](#))

#### *Database Standards*

- [GDPR/CCPA Data Retention Policies](#): Clear policies for data storage and retention, especially regarding personal and sensitive data

## Section C. Scope of Work

### C.1 Deliverables

1. A Web Application named CASEflow, a secure web application which allows users to login to access protected data(emails), and get an AI generated email response which improves on the Large Language Model we Finetune.
  - 1.1 Given real company data (data cleaned for privacy), we are to provide Fine Tuning to the existing Large Language Model to improve speed and performance of the AI response.
  - 1.2 A Feature for employees to rate AI-generated suggestions, improving the LLM over time.
  - 1.3 Functionality for employees to review, edit, and send the AI-generated email responses using the application.
  - 1.4 Metrics and reports on response times and AI accuracy improvements using statistics from user feedback(this would be done through testing as we Fine-Tune the LLM).
2. Provide technical documentation, a user manual, and build steps for running the application CASEflow.
3. Academic Deliverables
  - 3.1 Team Contract
  - 3.2 Project Proposal
  - 3.3 Preliminary Design Report
  - 3.4 Fall Poster and presentation
  - 3.5 Final design report
  - 3.6 Capstone EXPO poster and presentation

Some important issues to discuss with the design team, sponsor, and faculty advisor include the following:

The only third party vendor we might be unable to get access to is Amazon Web Services accounts. In our current plan of action we need an AWS account to work on the large language model which is a core feature of our application CASEflow.

Method of deployment, issues with our deployment, whether that our application is failing to build, or not enough testing being done because everything is being run on our local environment until deployment.

All deliverables can be worked on remotely; however, due to a large volume of remote work the group has multiple means of communication including Discord, Slack, email, and Zoom.

Meeting Minutes and other information is all organized in shared google drives for effective remote work.

## C.2 Milestones

Our team will be utilizing Agile Development with a focus on Continuous Integration. The project will be divided into distinct phases, or 'sprints,' each spanning 2-3 weeks, depending on the complexity of the tasks. The length and content of each sprint will be collaboratively decided by the student team, project mentor, and faculty advisor. The deliverables from each sprint will serve as milestones to track progress and ensure that the final project objectives are completed on time.

### Sprints:

Sprint 1: Set up basic project structure Date: 9/9/24 - 9/23/24

- Set up folder structure for Frontend(react), Backend(FastAPI), and connect database(PostgreSQL).

Sprint 2: Authentication Date: 9/23/24 - 10/7/24

- Complete Sign in and Log in page UI(react)
- Create JWT tokens to enable secure logins, Connect frontend and backend for testing.

Sprint 3: Case Management & Representative Schema Date: 10/7/24 - 10/21/24

- Create routes for Different pages
- Create Casetable(frontend)
- Create database models for Casetable(backend)
- Create API Endpoints for retrieval and entry of the CaseTable information.
- Create Form for users to submit new cases, and a dashboard to view assigned cases(frontend)

### Milestone: Completion of basic project design

**Date: 10/21/24**

Sprint 4 (Weeks 8-10): Connecting LLM

Date: 10/21/24 - 11/4/24

- Connect LLM to backend VIA AWS Sagemaker (pending aws account approval)
- Test all components of application before fine tuning next sprint

Sprint 5 - 8: LLM Fine-Tuning

Date: 11/4/24 - 12/16/24

### Milestone: Completion of LLM Fine Tuning

**Date: 12/16/24**

Sprint 9 - 10: LLM Feedback

Date: 1/13/25 - 2/10/25

- LLM Feedback and metrics report testing.

Sprint 11 (Weeks 19-20): Testing

Date: 2/10/25 - 2/24/25

- Testing and Documentation write up

**Milestone: Completion of testing**

**Date: 2/24/25**

Sprint 12: Bug Fixes

Date: 2/24/25 - 3/10/25

Sprint 13: Deployment

Date: 3/10/25 - 3/24/25

**Milestone: Completion of project**

**Date: 3/24/25**

**Academic Milestones:**

**CMSC 451 Deliverables**

Team Contract

Date: 9/4/24

Project Proposal

Date: 10/11/24

Preliminary Design Report

Date: 11/15/24

Fall Design Poster and Presentation

Date: 12/12/24

**CMSC 452 Deliverables**

Final Design Report

Date: 2/23/25

Capstone EXPO Abstract

Date: 3/5/25

Capstone EXPO Poster

Date: 4/18/25

Capstone EXPO Presentation

Date: 4/25/25

### **C.3 Resources**

#### **Paid Resources**

Access to Amazon web service account, needed to utilize AWS SageMaker for LLM fine-tuning and machine learning aspect. These accounts will be provided by Project Sponsor (CoStar Group).

#### **Free Resources**

We will be using GitHub for our Version Control System, which is accessible to all VCU students. Programming language, frameworks and technologies are all available to open source download. We will be using React, FastAPI, PostgreSQL, SwaggerUI and Visual Studio Code. For Communication we are using Slack, Discord, Zoom, and Email.

## **Section D. Concept Generation**

When finalizing our design concept, we focused on the needs of the customer, which are the Case Management Employees at CoStar Group. When drafting design concepts, we thought it was beneficial to start simple then build complexity. We started with a basic AI-powered email generator and progressively introduced additional functionalities, such as case management and advanced AI capabilities.

### **Design Concept 1: Simple AI-Powered Email Generator Application**

The first design concept starts simple, but still aims to meet all of the necessary customer requirements. This design has a simple email generation system. It incorporates a lightweight, easy to use, LLM to generate email responses based on predefined templates. Once a draft is produced, representatives have the option to edit the draft with a simple text editor.

The primary technologies for this design include React for the frontend, providing a simple interface, and FastAPI for the backend to facilitate API calls to the AI model. OpenAI's GPT-3.5 would handle email generation, because it is easy to use and does not require extensive fine-tuning. To protect sensitive data, there would be a simple user authentication system. User authentication would rely on a basic username-password structure.

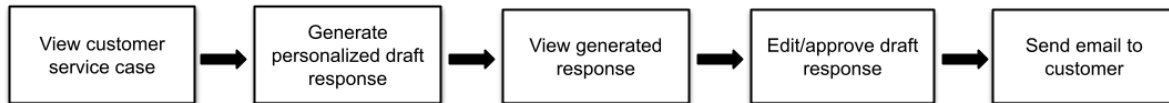
The advantages of this design are its simplicity, ease of development, and cost-effectiveness. The simple features mean faster deployment and lower development costs. It is also user-friendly, as the straightforward interface allows representatives to quickly learn and use the tool. However, it has disadvantages due to its simplicity. The AI model is not robust enough to address multiple types of emails. It uses predefined templates, which restricts its flexibility for generating responses to a variety of emails. In addition, there is no rich text editor for representatives to use to edit the email drafts. The basic feedback loop and use of a simple LLM is not robust enough for the AI to continuously learn and improve.

This concept addresses the design problem by providing a simple yet effective tool AI-assisted email response generation. Representatives benefit from a system that generates appropriate responses while remaining easy to use. However, more advanced email generation and case management is necessary to make the application useful.

### **Design Concept 2: Simple AI-Powered Email Generator Application with Case Management System**

This design builds on Design Concept 1 by introducing a case dashboard to the application. This allows users to view and manage existing cases more effectively. The dashboard provides an overview of case details such as their status and assignment. Like the first concept, this system includes basic login functionality for authentication. This concept could use React for the frontend along with a component-based design to implement the case dashboard and email generator pages. For the backend, FastAPI would also be utilized to manage case data and communicate with the AI email generator. The email generation engine would also use OpenAI's GPT-3.5 due to its ease of use.

The image below shows the workflow of this design concept. Users can view their assigned cases through the dashboard, generate personalized email drafts based on GPT-3.5's suggestions, and edit the drafts as necessary before sending them to customers.



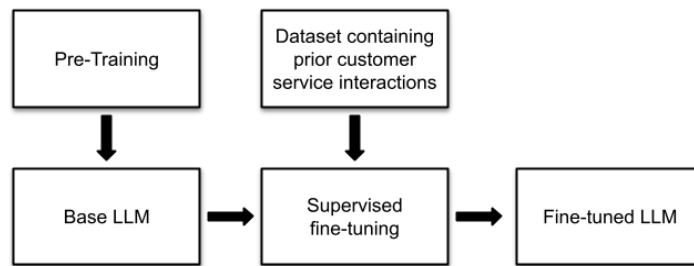
This design offers several key advantages. First, the introduction of a case dashboard enhances case management, providing users with an overview of cases. This allows representatives to manage and prioritize cases more effectively. Additionally, the integration of OpenAI's GPT-3.5 for email generation ensures high-quality, contextually relevant drafts that can be customized with ease. The interface remains simple and user-friendly. However, this design also comes with some drawbacks. One key limitation is the lack of a rich text editor, which still restricts the ability to fully personalize email responses. Representatives may struggle with editing emails for specific customer needs, limiting the flexibility of the system. Additionally, the lack of a robust feedback loop means that the AI-generated email responses cannot be continuously improved based on user feedback. The basic authentication system may become a security concern as the system grows and handles more sensitive data.

By introducing a case dashboard and improving the email generation technology, this design addresses some of the limitations of the first concept. However, it still lacks a rich text editor and a robust feedback loop, which could limit the personalization and adaptability of email responses. This design is more sophisticated but remains relatively simple, making it a practical next step for enhancing user experience without adding significant development complexity.

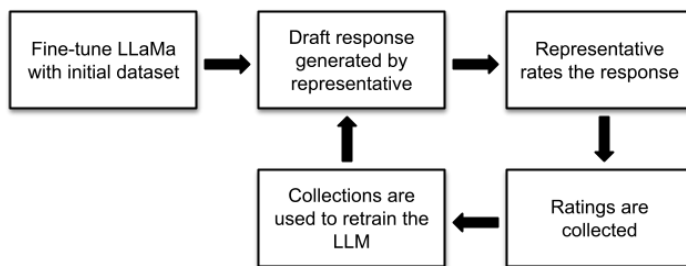
### **Design Concept 3 (Final): Advanced Case Management and Email Generation System**

The final design concept builds on the previous designs, adding advanced features to meet more complex user needs. This design will use LLaMA 3.1, running on Amazon SageMaker, to generate high-quality email responses, improving the AI's capabilities. This implementation will have a more robust case management system, allowing users to add, delete, and edit existing cases. On the case dashboard, representatives can sort the case by status, who it is assigned to, or creation date. Furthermore, each case will have details and a timeline page that representatives can use to have a holistic view of the case. The backend, built using FastAPI, will store this critical case information. The backend will also store draft email responses, email response rates, and real customer emails. This design also incorporates a rich text editor, powered by Lexical, to allow representatives to format and customize email responses. Secure login functionality is implemented using JWT tokens to ensure user security.

The workflow for Design Concept 3 incorporates continuously improving the AI model, specifically LLaMA 3.1, to generate effective email responses. The process begins with the pre-training phase, where the base LLM is trained on a large, general dataset. Next, the model will be fine-tuned using a dataset of prior customer service interactions. The image below shows the cycle that the LLM would follow to generate emails:



A feedback loop is implemented to continuously enhance the model's performance. As representatives use the AI to draft emails, they can rate the quality of the generated responses. These ratings, along with the finalized email drafts, are fed back into the system as additional training data. Over time, this retraining process helps refine the model, allowing it to produce increasingly accurate and context-aware responses. The image below shows the workflow for the feedback loop.



There are several advantages to this design concept, which is ultimately why we decided on it. The integration of LLaMA 3.1 enables advanced email generation, ensuring that responses are contextually accurate and professional. This ability enhances customer interactions. The rich text editor offers representatives flexibility when editing email drafts. The advanced case management dashboard allows users to track, edit, and manage cases efficiently, with additional features like a timeline view providing a detailed history of case progression. This helps representatives with their work by storing everything they need in one application. Secure user authentication, implemented using JWT tokens, ensures that user data and system access are protected. Furthermore, the modular architecture of the system, supported by FastAPI and SageMaker, allows for scalability as user demands grow. The system's capability to store user feedback allows for continuous improvement of AI-generated responses over time.

However, this robust design comes with certain disadvantages. The advanced features like AI-driven email generation, rich text editing, and robust security measures increase the complexity of development. This may affect deployment times, as these features will take longer to implement. The complexity of this system could also impact system performance. The initial costs associated with implementing Amazon SageMaker and maintaining a robust backend infrastructure are higher compared to simpler systems. The system's effectiveness depends on the

quality of the AI-generated responses, so inaccuracies from poor training data or limited feedback could affect performance.

This concept offers a robust solution by addressing the need for detailed case management, rich email editing, and secure user authentication. However, the complexity of integrating these advanced features increases the development effort and potential risks, such as performance concerns or a steep learning curve for users. Despite these challenges, this concept is the most scalable and adaptable, making it well-suited for long-term use as the system evolves.



## Section E. Concept Evaluation and Selection

In our initial planning phases, we researched several methods to approach the project. Through our research and conversations with our sponsor, we identified key objectives and constraints that our design concepts must satisfy to ensure project success. In this section, we will evaluate the design concepts we researched and develop a systematic method to compare them against the established criteria. Based on this evaluation, we will select the most suitable options for the project.

### Criteria

The primary goal of our project is to enhance the efficiency of customer service representatives when responding to clients. This goal can be achieved by designing essential criteria. To satisfy these requirements, we need a robust and efficient application, while also addressing practical constraints such as budget, development time, and tool complexity.

The criteria we used to evaluate our design concepts are:

- **Ease of Use:** How much of an adjustment would the representative go through to use our application.
- **Performance:** How well the application meets the functional needs.
- **Reliability:** The robustness of the application and its capacity to handle expected workloads.
- **Cost-effectiveness:** Whether the application offers a good balance between cost and functionality.
- **Security:** How secure is the application's handling of sensitive data.
- **Response Quality:** Does the model's generated responses meet the client's requirements.

### Design Concepts

In section D of the report we discussed our three iterations of design concepts. In concept one, we came up with a simple application that is only able to generate responses based on a prompt that the customer service representative supplies. The representative is then able to copy and paste the generated response into their own program.

For concept two, we introduced the addition of a case dashboard into our program. This dashboard gives basic information on each case such as status, assigned representative, and communication history. This extra information can be fed into the AI model when generating responses for a more context-aware and personalized response.

Last, concept 3 adds our fine-tuned model with a more fully fledged-out case management system. This design iteration will allow users to add, delete, and edit existing cases, and will feature a more informative dashboard. This dashboard will contain a full timeline of the case, and a Lexical text editor for responses with the AI response generator integrated into the system.

In addition to that we also created a user authentication system using industry standards such as JWT authentication and OAuth 2.0 scheme.

### Scoring Methodology

We will score how well each concept fits each criterion based on discussions with our advisors. For each criteria we will rank design concepts from best fit to worst fit. For example, if we had five options in a concept, the best fit would be ranked (5) and the worst fit would be ranked (1). These numbers will then be summed for a total score and for each criteria and the concept with the highest score will be chosen.

### Decision Matrix

Criteria	Design Concept 1	Design Concept 2	Design Concept 3
Ease of Use	3	2	1
Performance	1	2	3
Reliability	1	2	3
Cost-effectiveness	3	2	1
Security	1	2	3
Response Quality	1	2	3
<b>Total Weighted Score</b>	<b>10</b>	<b>12</b>	<b>14</b>

### Results

After our evaluation, we decided to proceed with design concept three for our application. This concept is far more complex than the other concepts with many moving parts. While it has a lower ease of use due to its higher complexity, it makes up for it in other areas. This concept will have the highest response quality due to its access to the full communication history between the representatives and the customers. Since we are using an open source model, LLaMa, we are able to keep customer service communication data within the company, increasing security. LLaMa 3 8B is also far cheaper to use for this application than other models, as we only pay for the time it is being run. Overall, design concept three is the most difficult to implement, however, it offers the best balance between cost-effectiveness, security, and response quality, making it the best choice for our application.

## **Section F. Design Methodology**

The success of this project relies on a robust and iterative design methodology that guarantees the final product meets all specified objectives and client requirements. This section outlines the methods and processes that we created to help with evaluation, improvement, and validation of the design. We will discuss our computational, architectural, and validation methods.

### **F.1 Computational Methods**

The primary computational method used in this project involves fine-tuning a Large Language Model (LLM) on real customer service data. This process will be conducted using the LLaMa 3 8B model, with fine-tuning being conducted in Amazon SageMaker. Key steps include:

- 1. Data Preprocessing**

Before we can feed this data into our model for fine-tuning we need to clean it. First, we must wipe all Personally Identifiable Information (PII).

- 2. Fine-Tuning**

With our preprocessed text we can begin to fine-tune the model. Our goal is to create a context-aware model that writes similarly to a CoStar customer service representative. We plan on fine-tuning our model by designing a prompt that shows an incoming email with its corresponding outgoing email.

- 3. Model Evaluation**

To evaluate our model, we will test its output against real customer service representatives' responses. We can look at measurements such as word similarity between the model and real representatives before and after the fine-tuning to see if our model made any improvements.

- 4. Continuous Improvement**

We will implement a feedback loop in production that allows representatives to rate model responses. The feedback will further fine-tune the model to create continuously enhanced responses.

### **F.2 Experimental Methods**

We developed experimental methods to validate the various parts of our application and their usability for customer service representatives. Key aspects include:

- 1. Testing Equipment:**

We can conduct testing with tools such as Postman to simulate API interactions between

the frontend (React) and backend (FastAPI).

2. **Test Setup:** Representative email drafts will be generated for various customer service cases. Test cases will evaluate accuracy, response relevance, grammatical correctness, and overall professional tone.
3. **Data Acquisition and Instrumentation:** Automated logs will record response generation times, error rates, and user interactions during testing. Observational data will be collected during usability sessions with representatives.
4. **Testing Procedures:**
  - **Baseline Testing:** Evaluate the default performance of the fine-tuned LLM against standard customer service cases.
  - **Usability Testing:** Collect qualitative feedback from employees regarding the quality and efficiency of AI-generated responses.

### **F.3 Architecture/High-level Design**

**The high-level architecture consists of the following components:**

1. **Frontend:**

A React-based web application will provide an interface for representatives to review and edit AI-generated drafts. Core features include a dashboard, case management interface, and response editor.
2. **Backend:**

FastAPI will handle authentication, response generation requests, and database management. JWT-based authentication ensures secure access to the application.
3. **Database:**

PostgreSQL will store case data, response templates, representative feedback, and system logs. Data will be organized to support efficient retrieval and management.
4. **Continuous Integration/Deployment (CI/CD):**

CI/CD pipelines will automate testing and deployment processes to ensure reliable updates. GitHub Actions and AWS CodePipeline will facilitate this workflow.

### **Validation Procedure**

Validation is important to ensure that the system meets the client's requirements and performs as intended. The following steps outline the validation plan:

**Performance Metrics:** The final design will be evaluated based on:

- **Response Accuracy:**  
After we conduct fine-tuning we will give the model real-world incoming customer emails and compare the model's responses to real representative responses. We can measure accuracy based on metrics such as word similarity between the model's and real representatives' responses.
- **Efficiency Gains:**  
We can measure how long on average it would take for someone to generate a response and respond to customers using our model, and compare that to the current average customer service response time.
- **User Satisfaction:**  
With our integrated AI generated response rating system, we can gauge how well our model is performing based on the feedback given in the ratings.

### **Client Demonstration**

Our team will schedule a demonstration with CoStar in the spring semester to present a working prototype. During the demonstration, the system's features, including AI-generated email drafts, case management functionality, and feedback integration, will be showcased. After this demonstration we can take feedback to make improvements to our application.

This design methodology ensures a robust and systematic approach to achieving a validated solution that ensures the client needs are met in our final iteration.

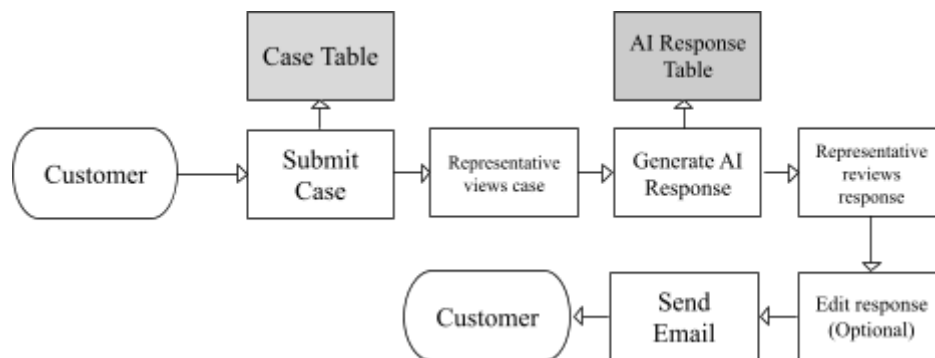
## Section G. Results and Design Details

We have successfully implemented key components of the web application designed to streamline customer service email responses through AI-powered assistance. By focusing on both backend and frontend development, we've made significant progress toward a fully functional prototype. This includes efficient case management, intuitive user interfaces, and the start of AI integration for generating email drafts. Together, these advancements have brought us much closer to achieving our objectives of enhanced response times, while maintaining professional communication standards, and reducing the workload on customer service teams.

### G.1 Modeling Results

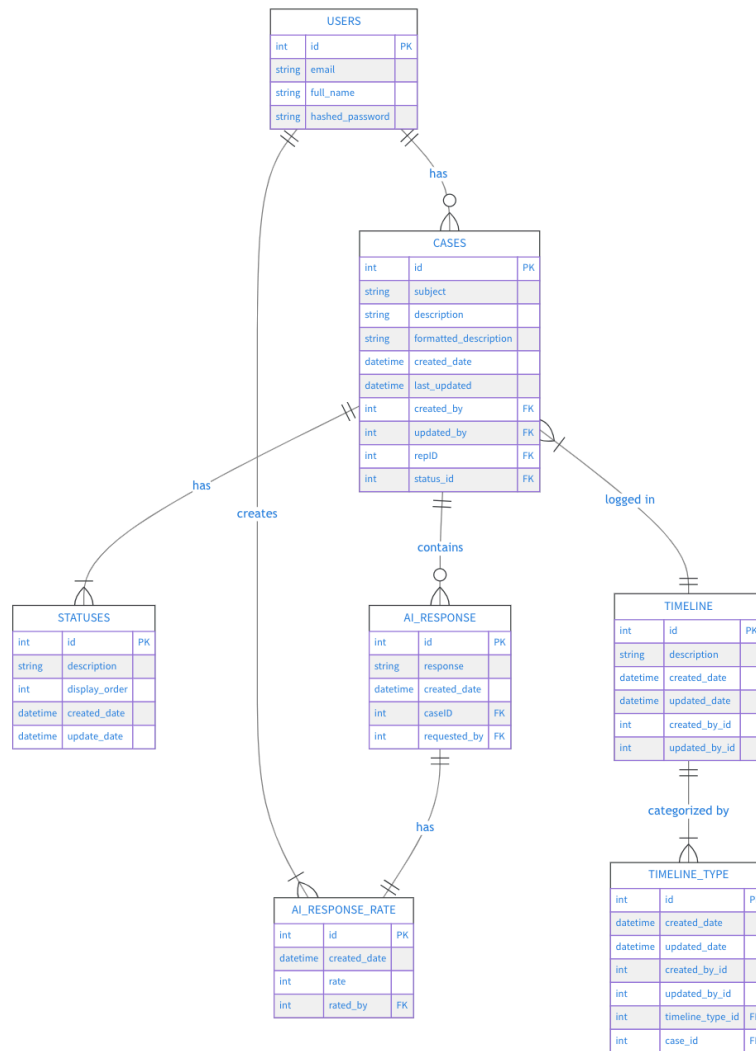
In this subsection, we present the key models and diagrams that were essential in shaping the design and functionality of our AI-powered customer service web application. These models were pivotal in making decisions, streamlining processes, and ensuring that the system aligns with the project's goals.

The following model is a general representation of the flow of processes in our system. It illustrates the sequence of events, from case submission to AI-generated email response, human review, and final delivery to the customer. As processes advance from one step to the next, we can also see how certain data, such as cases and AI responses are stored.



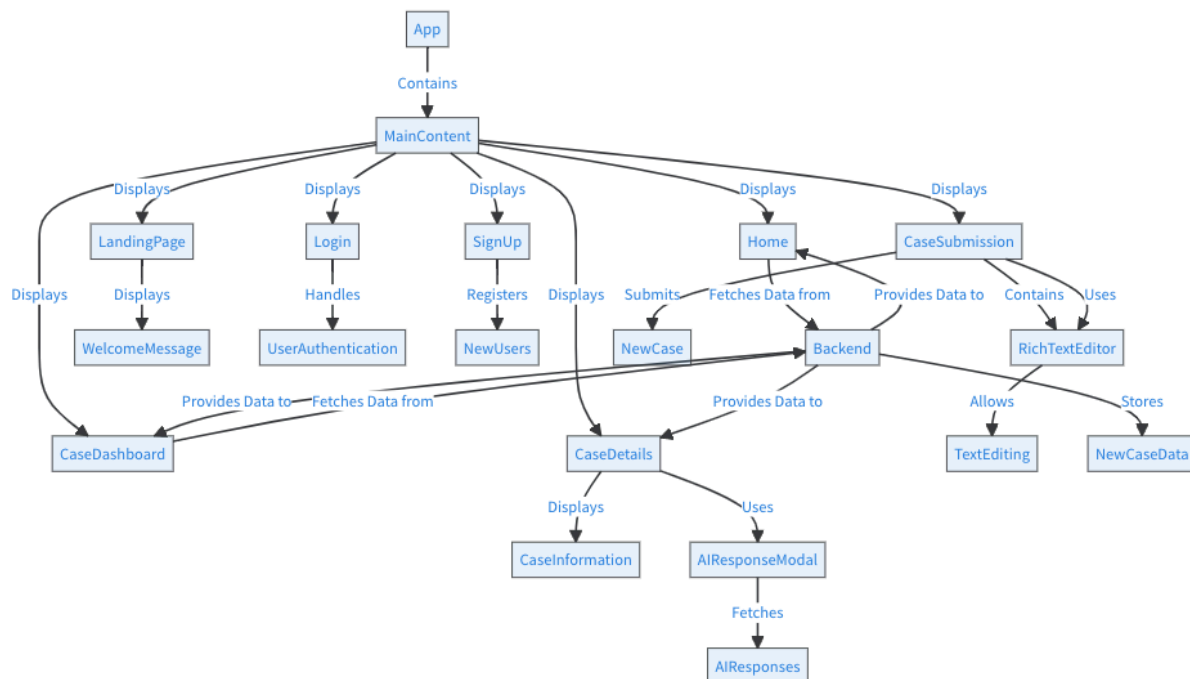
By mapping out these processes, we ensured a smooth workflow that balances automation with human oversight, preventing errors and maintaining a high standard of communication.

Using FastAPI, we stored our data in a database with the following tables: Users, Cases, Roles, Statuses, AIResponse, AIResponseRate, TimelineType, and Timeline, which are connected to our frontend to support the functionality of our application. Multiple of our tables in our database depend on each other and pull information from each other to then give to our frontend to use when producing our user interface using various CRUD functions and endpoints to, for example, submit cases, create user logins, etc.. The database schema for our project has been meticulously designed to ensure optimal performance and maintainability. Each table is tailored to store specific types of data, facilitating efficient data management retrieval. The relationships between these tables are illustrated in an Entity-Relationship diagram, which provides a clear visualization of how entities interact.



The Users table, for example, is linked to the Cases table and AIResponseRate table, highlighting that users can create multiple cases and rates. Similarly, the Cases table has associations with Statuses, AIResponse, and Timeline, indicating various stages and actions related to a case. These relationships help streamline the process of tracking and managing customer service inquiries. The AIResponse Table is particularly significant as it stores the AI generated responses, which are then rated and reviewed by employees. This setup ensures a continuous feedback loop, allowing the AI model to improve over time based on user ratings stored in the AIResponseRate table. By implementing this database design, we will have achieved significant improvements in managing customer service cases. The structured approach to storing data ensures consistency and reliability, while the comprehensive schema supports complex queries and data analysis. The ER diagram serves as an essential tool for understanding the data flow and interactions, making it easier for developers to navigate and modify the database as needed. Overall, creating this model has enabled us to deliver a robust and scalable solution that enhances the efficiency and effectiveness of customer service operations.

In addition to the backend database schema, our frontend design is crucial for delivering a user-friendly interface. Modern web technologies were used to create an intuitive and responsive React-based application. We created a frontend component diagram to illustrate the structure and functionality of our application. This diagram provides a detailed overview of the various components used in the frontend and how they interact to deliver a seamless user experience.



Within our app, we implemented various components, including LandingPage, Login, SignUp, Home, CaseSubmission, AIResponseModel, CaseDashboard, CaseDetails, and RichTextEditor. The diagram shows Home, LandingPage, Login, SignUp, CaseDashboard, CaseDetails, and CaseSubmission grouped together under Main Content to show everything that works together to dynamically display different views based on user interactions. The AIResponseModel component enhances the case management process by providing AI-generated responses that are reviewed and rated by employees. The RichTextEditor component allows users to craft detailed and formatted case descriptions, improving the quality of information submitted. By implementing this structured approach, we ensure that our frontend components work together harmoniously, facilitating efficient data management and user interaction. This design not only meets the project's objectives but also enhances the overall user experience, making the application intuitive and easy to navigate. Overall, the frontend component diagram is a crucial element of our project, showcasing how we achieved a solution that aligns with our goal of improving customer service operations.

## G.2 Experimental and Testing Results

The experimental phase involves rigorous testing of various components and algorithms to ensure that our AI model meets the desired performance standards. Further testing will be done

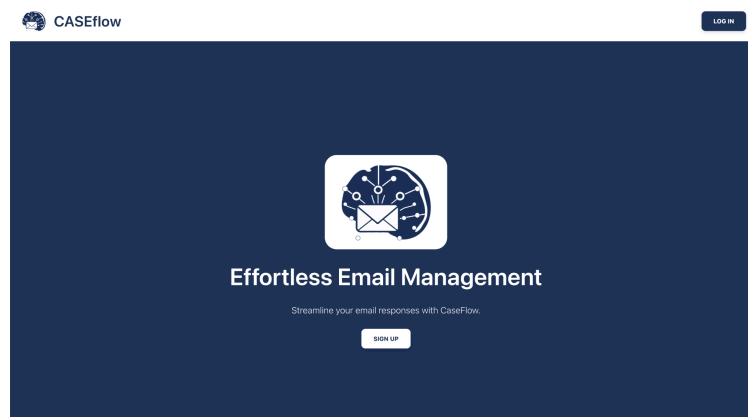


to fine-tune the LLM, using a diverse dataset of customer service interactions. The dataset will be cleaned and preprocessed to remove noise and irrelevant information. The LLM is to be trained on this dataset, so that the model will consistently generate high-quality responses with a high degree of contextual relevance. Our desired results will show significant improvements in response times and consistency when compared to human-generated responses alone. Aside from fine-tuning our LLM, we developed a prototype of our web application and conducted extensive testing to validate its functionality and user experience. Our prototype includes key features such as case submission, user authentication, a case details page, and a case timeline. The frontend was developed using React to create a user-friendly interface. We conducted functional testing to ensure that all components of the application worked as intended. This included testing the case submission process and making sure that cases were able to be viewed once they were submitted. There was also a lot of testing done to ensure that user authentication was working properly. We used JWT tokens for authentication and successfully created a functional login page. On the backend, we used FastAPI to test various endpoints to ensure their functionality. For each table we tested all CRUD operations and confirmed they were working as intended. Other endpoints, such as login and submit case endpoints, were tested through the frontend with a successful login and case submission. As we continue to develop our prototype, further testing will be done to ensure that our application meets all of our objectives and requirements.

### **G.3. Final Design Details/Specifications**

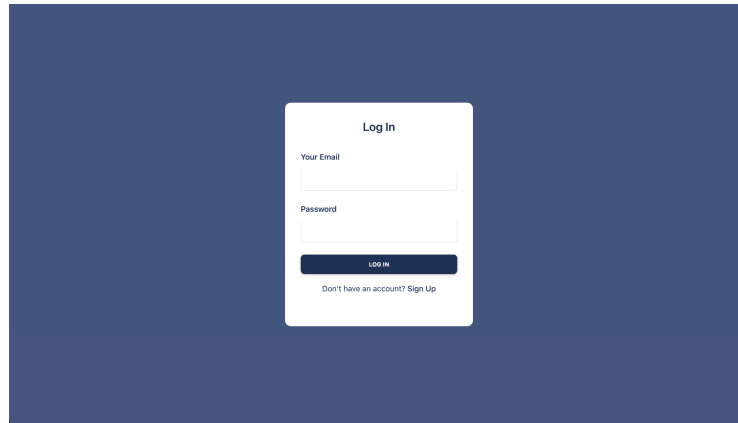
In this section, we provide an in-depth look at the final design of our AI-powered customer service web application. The following storyboard showcases the user interface and key components of the application. Each screenshot is accompanied by a detailed description, highlighting the design choices, functionality, and user experience considerations.

#### *Landing Page*



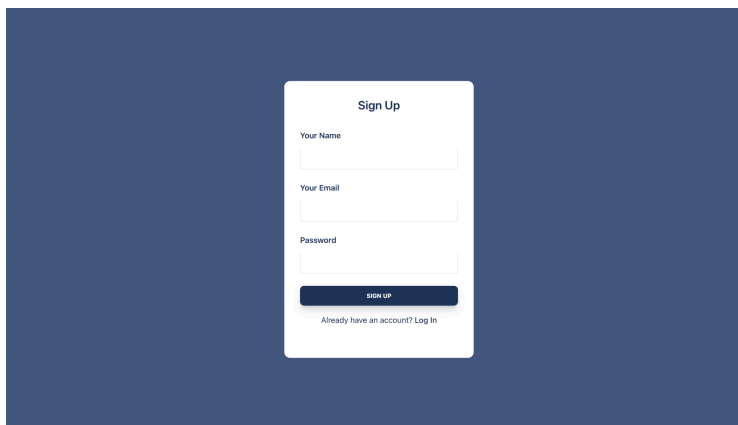
The landing page serves as the entry point for users visiting our application. The design is clean and simple, with clear navigation options for new and returning users. Key elements include quick access to login and sign-up pages, ensuring an intuitive and seamless experience.

### *Login Page*

The login page features a dark blue background. Centered on the page is a white rectangular form with rounded corners. At the top of the form, the text "Log In" is displayed. Below this, there are two input fields: "Your Email" and "Password". A dark blue button with the text "LOG IN" in white is positioned below the password field. At the bottom of the form, a link reads "Don't have an account? Sign Up".

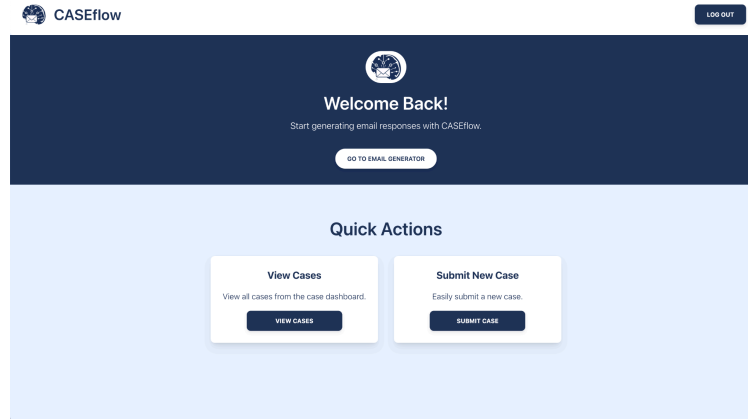
The login page is designed to provide secure access to the application. Users can enter their credentials to log in, or if you are a new user, there is an option provided to reroute them to the sign-up page. The layout is simple and user-friendly, focusing on functionality and security. This page is crucial for ensuring that only authorized users can access the system.

### *Sign-up Page*

The sign-up page features a dark blue background. Centered on the page is a white rectangular form with rounded corners. At the top of the form, the text "Sign Up" is displayed. Below this, there are three input fields: "Your Name", "Your Email", and "Password". A dark blue button with the text "SIGN UP" in white is positioned below the password field. At the bottom of the form, a link reads "Already have an account? Log In".

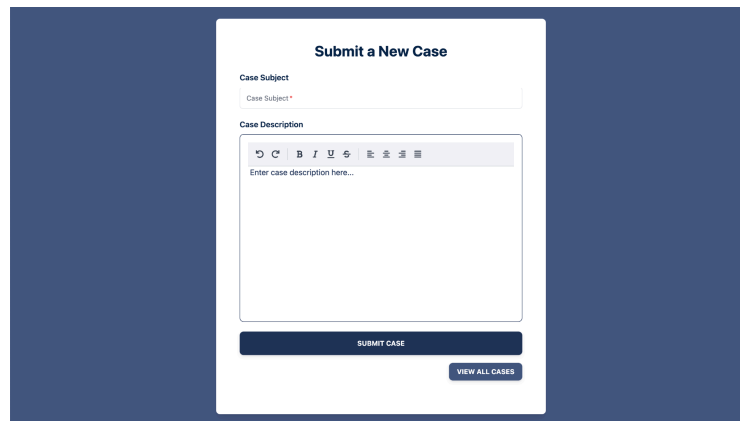
The sign-up page allows new users to create an account. The form collects essential information, such as name, email, and password. If the user already created an account, there is an option provided to reroute them to the login page. The design emphasizes ease of use, with clear instructions and validation to guide users through the registration process. This ensures a smooth onboarding experience for new users.

## Home Page



The home page serves as the central hub for users after logging in. It provides quick links to the main features, such as the case submission page and case details page. The design is very user-friendly and leaves little room for misinterpretation. Once a user is finished using the application, there is a provided logout button to securely close out of the application.

## Case Submission Form

The screenshot shows the 'Submit a New Case' form. The form is white and centered on a dark blue background. It has a title 'Submit a New Case'. Below the title is a 'Case Subject' field with a red asterisk indicating it is required. Below that is a 'Case Description' field with a rich text editor toolbar (bold, italic, underline, link, unlink, list, indent, outdent) and a placeholder text 'Enter case description here...'. At the bottom of the form are two buttons: 'SUBMIT CASE' and 'VIEW ALL CASES'.

The case submission form is designed to allow users to submit new customer service cases with detailed descriptions. It includes a rich text editor, providing users with the tools to format their text as needed. The form also includes validation to ensure that all necessary fields are completed before submission, such as the case subject field. Once a case has been submitted, there is also the option to go to the case dashboard and view all cases. The design of the case submission form is simple and provides a familiar appearance to other email applications providing comfort to the user in usability.

Case Dashboard

← RETURN TO HOME

Case Dashboard

Assigned To Me

Filter by Status

Sort By

ID	Description	Status	Assigned To	Actions
1	This is a test case		Rep null	View
2	This is another test case. Testing styling		Rep null	View
3	This is another test case. Testing styling		Rep null	View
4	This is another test case. Testing styling		Rep null	View
5	test case		Rep null	View
6	test case		Rep null	View
7	test case		Rep null	View
10	Test description		Rep null	View
11	This is the styled tes		Rep null	View
12	This is the description.		Rep null	View
13	submit		Rep null	View

The case dashboard displays a list of all customer service cases. It provides filters and sort options to help users quickly find specific cases. Each case includes key details such as case ID, description, status, who it was assigned to, and actions taken. The design ensures that users can efficiently manage and navigate through a high volume of cases, enhancing productivity.

Case Details Page

← RETURN TO CASE DASHBOARD

Case Details

Timeline

No description

Created by User ID: 1 on 11/14/2024, 12:00:00 AM

Case #1

Created By

1

Subject

Test Case #1

Description

This is a **test** case

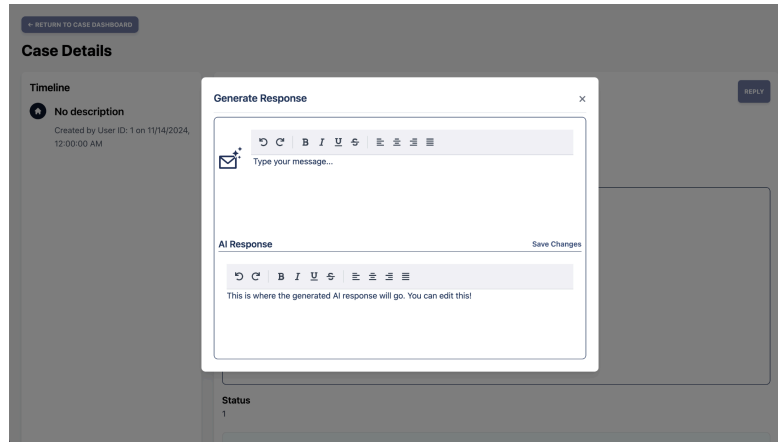
Status

1

REPLY

The case details page provides comprehensive information about a selected case. This includes the case history/timeline, who the case was created by, the case subject, the description of the case, and the status. There is also the option to reply to the case. The layout is designed to present information clearly and concisely, ensuring that users have all the necessary details to handle complex cases effectively.

## *AI Response Modal*



The AI response page allows users to view and interact with AI-generated email responses. Users can review the response with the option of making any necessary edits before sending them. The model also includes a rich text editor for styling purposes meant to improve user experience. The design focuses on usability and accessibility, ensuring that users can easily review and customize their responses as they see fit.

## **Section H. Societal Impacts of Design**

### **H.1 Public Health, Safety, and Welfare**

In the development of the AI-powered email generation and case management system for CoStar Group, the design team placed significant emphasis on public health, safety, and welfare. Recognizing that contemporary engineering must account for broader societal impacts, the team integrated several safety features that not only enhance the functionality of the product but also prioritize the well-being of users and customers alike. A foundational aspect of this design is secure authentication through JSON Web Tokens (JWT). This security measure ensures that only authorized personnel can access sensitive customer information, thereby protecting both employees and clients from potential data breaches. As organizations increasingly handle personal data, maintaining robust security protocols is essential to uphold public trust and comply with regulatory standards.

The system also incorporates data protection mechanisms that securely store critical information such as case details, draft email responses, and customer emails in a robust backend infrastructure. This approach minimizes risks associated with data loss or unauthorized exposure, addressing vital concerns about privacy and compliance with data protection regulations like GDPR. By safeguarding personal information, the design contributes to public welfare by ensuring that customer data remains confidential and secure. Moreover, the use of LLaMA 3.1 for generating contextually accurate email responses significantly enhances communication quality. Miscommunication in customer service can lead to frustration or even harm when important information is misunderstood or omitted. By leveraging advanced AI capabilities to ensure precise and appropriate responses, the system reduces the likelihood of errors that could adversely affect customer experiences.

The inclusion of a case management dashboard further supports public welfare by enabling representatives to efficiently track and manage cases. This feature allows for a holistic view of case statuses, which can lead to quicker resolutions for customer issues. When representatives have access to organized information about their cases, they can respond more effectively to customer needs, ultimately improving service quality. These safety features collectively ensure that the design promotes public health, safety, and welfare by addressing data security concerns, improving communication accuracy, and enhancing operational workflows. The consideration of these factors during the engineering design process not only meets technical requirements but also aligns with broader societal values. By prioritizing user safety and data protection from the outset, the team has established a product that is not only functional but also responsible for its impact on public welfare.

In summary, as contemporary engineers navigate complex design challenges, integrating considerations of public health, safety, and welfare into their projects is crucial. The thoughtful incorporation of these elements into the AI-powered email generation system exemplifies how engineering solutions can positively influence society while addressing user needs effectively..

## H.2 Societal Impacts

The implementation of this AI-powered system could have several societal impacts:

**Changing Work Dynamics:** The system may alter how customer service representatives interact with their work, potentially reducing repetitive tasks and allowing for more focus on complex customer issues.

**Customer Experience:** Faster, more accurate responses could lead to improved customer satisfaction and trust in businesses using such systems.

**AI Literacy:** As more companies adopt AI-powered tools, there may be an increased societal need for understanding and interacting with AI systems.

**Job Market Shifts:** While the system aims to assist rather than replace human workers, it may lead to changes in skill requirements for customer service roles, emphasizing AI interaction and oversight.

## H.4 Economic Impacts

The design has several potential economic implications:

**Operational Efficiency:** By streamlining email responses and case management, the system could significantly reduce operational costs for businesses.

**Market Competitiveness:** Companies adopting such advanced AI systems may gain a competitive edge in customer service, potentially influencing market dynamics<sup>1</sup>.

**Investment in AI Technology:** The success of such systems could drive increased investment in AI research and development, stimulating economic growth in the tech sector<sup>1</sup>.

**Workforce Adaptation:** There may be economic impacts related to workforce training and adaptation as employees learn to work alongside AI systems.

## H.7 Ethical Considerations

Several ethical considerations arise from the implementation of this AI-powered system:

**Data Privacy:** The system's reliance on customer data for training and improvement raises questions about data ownership and usage rights.

**AI Decision-Making:** As the AI generates email responses, there's a need to ensure transparency in how these decisions are made and to what extent they are relied upon.

**Human Oversight:** Maintaining appropriate human oversight and intervention capabilities is crucial to prevent over-reliance on AI-generated responses.

**Bias Mitigation:** Continuous monitoring and adjustment of the AI model are necessary to prevent the perpetuation or amplification of biases in customer interactions.

**Job Displacement Concerns:** While the system is designed to assist rather than replace human workers, ethical considerations around potential long-term impacts on employment should be addressed.

**Informed Consent:** Customers should be made aware when they are interacting with AI-generated responses, raising questions about disclosure and transparency; however, the AI only provides a template that is still edited by a customer service representative.

By considering these impacts throughout the design process, the team can make informed decisions to maximize benefits while mitigating potential negative consequences. This approach ensures that the AI-powered email generation and case management system not only meets technical requirements but also aligns with broader societal values and ethical standards.



## **Section I. Cost Analysis**

Our project will be incorporating AWS SageMaker and LLaMa to clean our data and fine-tune our LLM. AWS offers a pay-as-you-go pricing model, where training instances is \$0.27 per hour, data processing is \$0.92 per hour, and inference is \$0.40 per hour. The cost of fine-tuning LLaMa models depends on the model size and training duration. We will be fine-tuning an 8 billion parameter model, which may cost around \$2000 for the entire process. Based on these components, the estimated total cost for fine-tuning the LLaMa model and cleaning the data will be approximately \$2000, more or less. This includes costs for training instances, data processing, and real-time inference. Fine-tuning the LLaMa model will take up the overall expense. These costs ensure that our AI-powered customer service web application operates efficiently and effectively, providing high-quality, automated email responses.

## **Section J. Conclusions and Recommendations**

The design team's journey toward the final design of the AI-powered email generation and case management system for CoStar Group reflects a thorough application of the engineering design process. This evolution involved iterative development, user feedback, and a commitment to addressing the specific needs of Case Management Employees.

### **Evolution of the Design**

The primary goals were to create an efficient email generation tool that could assist Case Management Employees in handling customer communications while also incorporating a robust case management system. The objectives included:

- Developing a user-friendly interface.
- Ensuring high-quality, contextually relevant email responses.
- Implementing secure user authentication.
- Allowing for continuous improvement of the AI-generated content.

### **Progression Through Design Concepts**

#### **Design Concept 1: Simple AI-Powered Email Generator**

Challenges: The initial design utilized a basic language model (LLM) for generating emails from predefined templates. While it was cost-effective and easy to deploy, it lacked flexibility and advanced editing capabilities.

Lessons Learned: The team recognized the limitations of a simplistic approach, particularly in terms of personalization and adaptability.

#### **Design Concept 2: Enhanced Email Generator with Case Management**

Improvements: This iteration introduced a case management dashboard, allowing users to view and manage cases more effectively while still generating email drafts using GPT-3.5.

Obstacles Overcome: The integration of case management features improved functionality, but the absence of a rich text editor remained a significant limitation.

#### **Design Concept 3: Advanced Case Management and Email Generation System (Final Design)**

Final Triumphs: The final design incorporated LLaMA 3.1 for superior email generation, a comprehensive case management system, and a rich text editor for enhanced customization.

Secure login using JWT tokens addressed data security concerns.

Continuous Improvement: A feedback loop was established to refine the AI model based on user interactions, ensuring that the system evolves with usage.

## Summary of Final Design Features

The final design is characterized by several key features:

**Advanced Email Generation:** Utilizing LLaMA 3.1 enables contextually accurate and professional responses.

**Robust Case Management Dashboard:** Users can add, delete, and edit cases while tracking their status through various views.

**Rich Text Editor:** This feature allows representatives to format and personalize email drafts effectively.

**Secure Authentication:** Implemented through JWT tokens to protect sensitive data.

**Feedback Loop for Continuous Improvement:** Ratings from users help refine the AI's performance over time.

### Future Considerations

While the final design meets the project's goals effectively, there are opportunities for further advancement:

**Scalability Enhancements:** As usage grows, additional features could be integrated to support larger datasets or more complex case scenarios.

**AI Model Updates:** Continuous training with diverse datasets could enhance the AI's contextual understanding further.

**User Training Programs:** To mitigate the learning curve associated with complex features, structured training sessions could be beneficial for users.

### Unanswered Questions

How will user feedback be systematically collected and analyzed to ensure ongoing improvements?

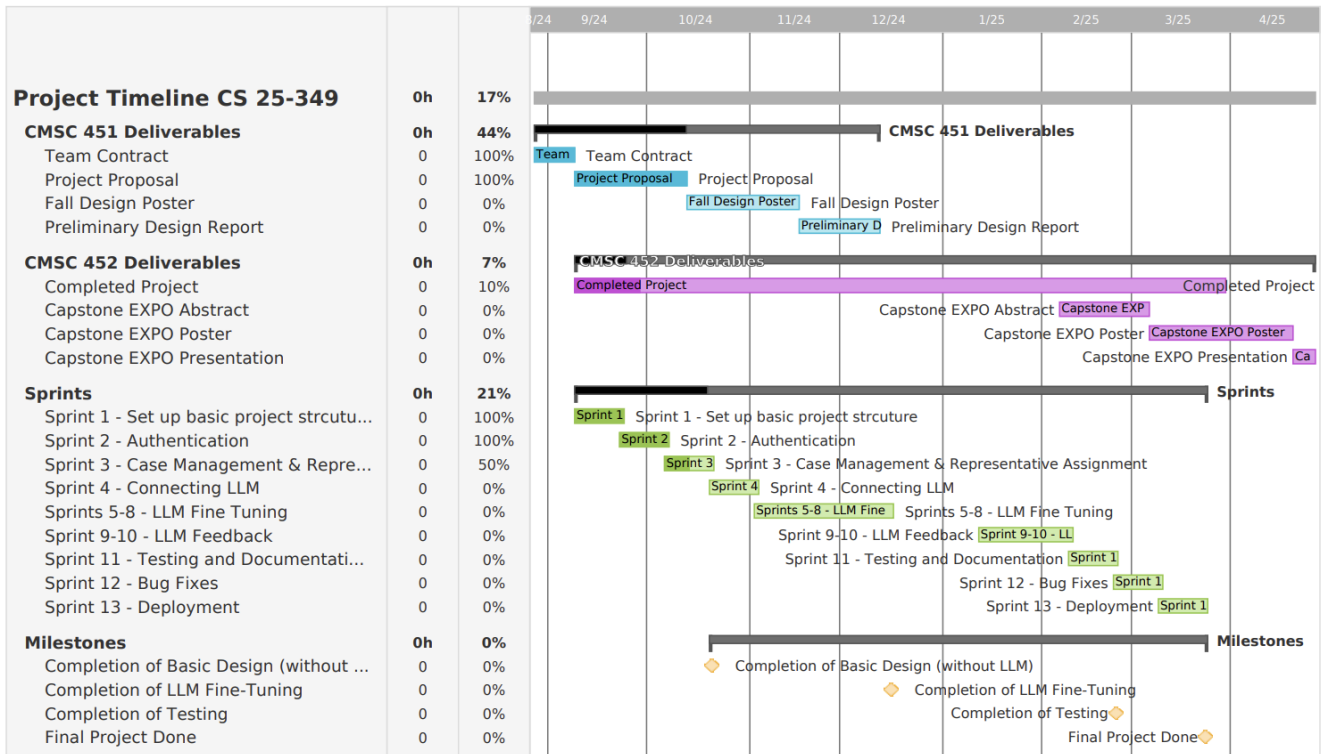
What measures can be implemented to ensure system performance remains optimal as complexity increases?

### Conclusion

In conclusion, the design team's iterative approach led to a sophisticated solution that effectively addresses the needs of CoStar Group's Case Management Employees. By starting with simple concepts and progressively adding complexity based on user feedback, they developed a comprehensive system that enhances communication efficiency while ensuring security and adaptability. Future researchers looking to build on this project will find comprehensive documentation and resources available, including testing plans and relevant schematics stored securely for easy access. This foundation will facilitate seamless advancement in enhancing this vital tool for customer service excellence.

## Appendix 1: Project Timeline

The figure below illustrates the timeline of our project in Gantt Chart form. The categories of tasks are as follows: CMSC 451 Deliverables, CMSC 452 Deliverables, Sprints, and Project Milestones. The sprints section will change as the team is assigned sprints.



## Appendix 2: Team Contract (i.e. Team Organization)

### Step 1: Get to Know One Another. Gather Basic Information.

<i>Team Member Name</i>	<i>Strengths each member bring to the group</i>	<i>Other Info</i>	<i>Contact Info</i>
Angela Harris	Organization, quick learner, communication, flexibility	Strong believer in the idea that there are no “dumb” questions	harrisam2@vcu.edu
Emma Smith	Problem-solving, React experience, leadership	I am passionate about learning new things and enjoy collaborating with others.	smith3@vcu.edu
Sohil Marreddi	Eager to learn from others, previous project experience	Intrigued to build real world products and create that product from scratch.	marreddiss@vcu.edu
Cameron Clyde	Adaptable, team-work, quick learner, front-end experience.	I'm always looking for ways to improve my knowledge and skills.	clydec@vcu.edu

<i>Other Stakeholders</i>	<i>Notes</i>	<i>Contact Info</i>
Faculty Advisor: Preetam Ghosh, VCU Engineering		pghosh@vcu.edu
Sponsor: Keroles Hakem, CoStar Group		khakem@costar.com

**Step 2: Team Culture. Clarify the Group's Purpose and Culture Goals.**

<b><i>Culture Goals</i></b>	<b><i>Actions</i></b>	<b><i>Warning Signs</i></b>
Attend every meeting	<ul style="list-style-type: none"><li>- Set up meetings in shared calendar</li><li>- Send reminder message in group chat day of meeting</li><li>- Discuss when there are schedule conflicts/reschedule as needed</li></ul>	<ul style="list-style-type: none"><li>- Student misses meeting without communication</li></ul>
Holding each other accountable to stay on top of work	<ul style="list-style-type: none"><li>- Work on project weekly</li><li>- Discuss goals for the week and add them to meeting notes</li><li>- Set reasonable deadlines for goals</li></ul>	<ul style="list-style-type: none"><li>- Student shows up for weekly meeting with no considerable work done</li></ul>
Creating a safe environment with open communication	<ul style="list-style-type: none"><li>- Talking through problems/conflicts as they arise</li><li>- Communicate schedule</li><li>- Open to new ideas</li></ul>	<ul style="list-style-type: none"><li>- Shutting down other teammates ideas</li><li>- Passive aggression</li></ul>

### Step 3: Time Commitments, Meeting Structure, and Communication

<i>Meeting Participants</i>	<i>Frequency Dates and Times / Locations</i>	<i>Meeting Goals Responsible Party</i>
<i>Students Only</i>	<i>As Needed, On Discord Voice Channel or Zoom</i>	<i>Discuss updates on progress and challenges Work through challenges as group and discuss potential solutions (Angela will record notes and upload them to shared Google Drive)</i>
<i>Students Only</i>	<i>Every Monday 7-9pm in library room</i>	<i>Actively work on project, discuss any difficulties Review previous weeks progress and discuss goals for upcoming week (Emma will take notes and upload them to shared Google Drive)</i>
<i>Students + Faculty advisor</i>	<i>As Needed</i>	<i>Update faculty advisor and get answers to our questions (Sohil will take notes on shared Google Docs; Angela will organize and lead meeting)</i>
<i>Students + Project Sponsor + Faculty advisor</i>	<i>Thursday at 6PM on Zoom</i>	<i>Update project sponsor to ensure we are on the right track Ask questions and discuss challenges (Cameron will take notes; Emma will organize and lead meeting; Sohil will demo prototype so far and give updates)</i>

#### Step 4: Determine Individual Roles and Responsibilities

<b><i>Team Member</i></b>	<b><i>Role(s)</i></b>	<b><i>Responsibilities</i></b>
Angela Harris	Logistics Manager	<ul style="list-style-type: none"><li>- Primary contact for communication with faculty advisor</li><li>- Documenting meeting times</li><li>- Obtaining information for the team</li><li>- Following up on communication of commitments</li></ul>
Emma Smith	Project Manager	<ul style="list-style-type: none"><li>- Primary contact for communication with sponsor</li><li>- Create a welcoming environment at meetings</li><li>- Document and organize team goals for week</li><li>- Schedule weekly meetings with sponsor</li><li>- Book library rooms for student meetings</li></ul>
Cameron Clyde	Test Engineer/Financial Manager	<ul style="list-style-type: none"><li>- Monitors team budget.</li><li>- Oversees experimental design, test plan, procedures and data analysis</li><li>- Leads presentation of experimental finding and resulting recommendations</li></ul>
Sohil Marreddi	Systems Engineer	<ul style="list-style-type: none"><li>- Takes notes of client/sponsor requirements to put together solutions</li><li>- Works with team members to design architecture of the software</li></ul>

#### Step 5: Agree to the above team contract

*Team Member:* Angela Harris      *Signature:* Angela Harris

*Team Member:* Sohil Marreddi      *Signature:* Sohil Marreddi

*Team Member:* Cameron Clyde      *Signature:* Cameron Clyde

*Team Member:* Emma Smith      *Signature:* Emma Smith



## References

- [1] A global leader in the digital transformation of the \$300+ trillion real estate industry. Home | CoStar Group. (2024). <https://www.costargroup.com/>
- [2] Gil-Gomez, H., Guerola-Navarro, V., Oltra-Badenes, R., & Lozano-Quilis, J. A. (2020). Customer relationship management: Digital Transformation and Sustainable Business Model Innovation. *Economic Research-Ekonomska Istraživanja*, 33(1), 2733–2750. <https://doi.org/10.1080/1331677x.2019.1676283>
- [3] Mesquita, T., Martins, B., & Almeida, M. (2022, October). Dense template retrieval for customer support. In Proceedings of the 29th International Conference on Computational Linguistics (pp. 1106-1115). <https://aclanthology.org/2022.coling-1.94/>
- [4] Sheth, J. N., Jain, V., & Ambika, A. (2024). Designing an empathetic user-centric customer support organization: Practitioners' perspectives. *European Journal of Marketing*, 58(4), 845–868. <https://doi.org/10.1108/ejm-05-2022-0350>
- [5] Malik, R., Subramaniam, L. V., & Kaushik, S. (2007, January). Automatically selecting answer templates to respond to customer emails. In IJCAI (Vol. 7, No. 1659, p. 3015). [https://www.researchgate.net/profile/Lv-Subramaniam/publication/220815935\\_Automatically\\_Selecting\\_Answer\\_Templates\\_to\\_Respond\\_to\\_Customer\\_Emails/links/55b6580e08aec0e5f436fe0e/Automatically-Selecting-Answer-Templates-to-Respond-to-Customer-Emails.pdf](https://www.researchgate.net/profile/Lv-Subramaniam/publication/220815935_Automatically_Selecting_Answer_Templates_to_Respond_to_Customer_Emails/links/55b6580e08aec0e5f436fe0e/Automatically-Selecting-Answer-Templates-to-Respond-to-Customer-Emails.pdf)