IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.135103

AI-Driven Automation of Centralized Email Triage and SLA Management

Dr.T. Amalraj Victoire[1], S. Nithyalakshmi[2]

Professor, Department of Master Computer Application, Sri Manakula Vinayagar Engineering College, Pondicherry, India[1]

Student, Department of Master Computer Application, Sri Manakula Vinayagar Engineering College, Pondicherry, India[2]

Abstract: As the number of digital messages increases companies have a problem with their customer support processes. This is where they sort through emails and decide how to answer them. When people do this job it can be slow. Cost a lot of money if they do not answer emails on time. The old way of doing things relies on people to look at each email and decide what to do with it. This can lead to mistakes and waste time.

In this paper we talk about a way to automate the process of sorting through emails and managing the rules that companies must follow to answer emails on time. Our system can look at emails understand what they say and decide what to do with them without anyone helping. It is like a first step in answering emails.

We built a system that uses a special computer program to understand what emails say. It can look at emails soon as they arrive and use special techniques to understand what the person who sent the email wants and how they feel. Then it turns these emails into tickets that the company can use to answer the emails. The system also makes sure that the company answers emails on time by tracking how long it takes to answer them and sending reminders if someone is running late.

We also made a website that customer support staff can use to answer emails. This website is very fast and easy to use so staff can answer emails quickly. Do not have to wait. By automating the process of sorting through emails our system helps companies answer emails faster makes sure that no important emails are missed and lets customer support staff focus on answering emails. This is a way to manage customer relationships using artificial intelligence.

Our system is a way to manage customer relationships. It uses intelligence to make the process of answering emails faster and more efficient. Customer support staff can focus on answering emails and the system takes care of the rest. This is an improvement, over the old way of doing things.

Keywords: Artificial Intelligence, Natural Language Processing (NLP), Email Triage, SLA Management, Workflow Automation.

1. INTRODUCTION

In todays world fast and reliable customer support sets businesses apart. As companies grow they get emails and customer support teams have to deal with a huge number of queries every day. These queries range from password resets to serious system failures that need to be fixed right away. To manage this teams, have to follow rules about how quickly they respond and resolve issues.

Usually managing emails is done manually. Agents have to read each email understand it categorize it and then route it to the person. This is because its prolonged, flawed and can lead to wasted resources. Urgent requests can get lost in a sea of inquiries leading to missed deadlines, unhappy customers and Disengaged. Regular helpdesk software helps organize emails but can't understand the context so humans have to do all the work.

To fix these problems we're introducing a system: The AI-Driven Automation of Centralized Email Triage and SLA Management System. This system uses intelligence and modern web design to take the manual work out of customer support. It acts as an intermediary between customers and support agents.

This system automatically reads emails using IMAP and uses Natural Language Processing to understand the intent and sentiment of the mail.

It precisely categorizes the email. Creates a structured prioritized support ticket without human help.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.135103

The system also proactively manages deadlines. Of just tracking deadlines it monitors tickets and automatically escalates requests that are nearing their limits.

We built an responsive frontend dashboard using React.js so support staff can work efficiently with AI-sorted data. The interface uses "UI" design ensuring that agent actions are reflected instantly providing a seamless user experience.

The main contributions of this system are:

Automated Intent Extraction: A Python-based AI engine that elucidate email data and converts it into actionable support tickets.

Proactive SLA Enforcement: A dynamic tracking mechanism that prioritizes requests and escalates tickets to prevent deadline breaches.

High-Performance Architecture: A full-stack system integrating backend AI processing with a fast React.js frontend.

Operational Optimization: A reduction, in work allowing support teams to focus on resolving customer issues.

This framework shows how intelligent automation can transform inbox management enabling organizations to scale their support operations without sacrificing speed or quality of service.

2. LITERATURE REVIEW

The customer support workflows and the use of intelligence in text classification have been very interesting to people who do research and work in the industry. To understand what our system can do we need to look at how helpdesk systems have changed how Natural Language Processing is used to sort emails and how companies manage their Service Level Agreements.

2.1 The Evolution of Helpdesk and Ticketing Systems

In the past companies used shared email inboxes to talk to customers. This did not work well when there were a lot of emails. Researchers found ways to make workflows better which led to the creation of helpdesk platforms like Zendesk and Jira Service Management. These platforms helped support teams move from inboxes to organized ticketing systems but they were still not very active. They relied on rules to sort tickets like looking for specific words in the subject line. This meant that people still had to read and sort emails by hand which took a lot of time.

2.2 Natural Language Processing and Text Classification in Customer Support

To make things better researchers started using machine learning and Natural Language Processing to sort emails. At first they used algorithms like Naive Bayes or the Support Vector Machines to categorize emails based on keywords. These algorithms were good at filtering out spam. They had trouble understanding what customers really meant. Newer Natural Language Processing models, like Transformer-based architectures and Large Language Models are much better at understanding context. They can find out what the customer wants, how urgent it is and summarize emails. However most research focuses on how these models work in theory not on how to use them in real-world systems that talk to email protocols like IMAP and SMTP.

2.3 SLA Automation and Proactive Prioritization

Service Level Agreements are crucial for providing customer support. Traditionally companies treat Service Level Agreements like timers that start when a ticket is created. Support agents have to watch these timers and prioritize their work based on which tickets are closest to the deadline. Some researchers have proposed using models to forecast when a ticket might breach a Service Level Agreement but most commercial systems do not have automated escalation logic. If a ticket is miscategorized at the beginning it might breach a Service Level Agreement. Integrating automated categorization with dynamic Service Level Agreement escalation is not well explored in research.

2.4 Identifying the Gap

While there is a lot of research on Natural Language Processing, ticketing software and Service Level Agreements there is a gap in how these things work together. Current systems often have parts that do not work well together or they have slow user interfaces that cannot handle high-speed sorting. Our system fills this gap by using intelligence from the start. It combines Natural Language Processing with proactive Service Level Agreement escalation rules and presents the data in a fast and efficient way using React.js. This project provides a solution, to the problem of manual email sorting.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.135103

3. SYSTEM ARCHITECTURE

Our system is made up of an three-layer The AI-Driven Centralized Email Triage and SLA Management System is built to handle a lot of work with parts that work separately but together. It uses a client-server model, which keeps the hard work of the artificial intelligence engine separate from the fast data presentation layer of the frontend.

The system has four parts:

* The Ingestion Layer

* The AI Processing Engine

* The Data Management & SLA Core

* The Client Interface Layer

3.1 Overview of Data Flow

When a customer sends an email the system automatically intercepts it extracts the information processes the text using Natural Language Processing (NLP) and creates a database ticket. It then calculates the SLA parameters. Instantly updates the support agents dashboard.

3.2 The Ingestion Layer

The Ingestion Layer is the point of contact. It uses a Background Process connected using the IMAP (Internet Message Access Protocol) protocol, instead of Technical Support manually refreshing their mail inboxes.

Polling: The system securely connects to the centralized email server and continuously checks for new unread messages.

Payload Extraction: When it detect email it securely fetches the raw data removes unnecessary HTML markup and metadata and isolates the core text body, subject line and sender details, for processing.

3.3 The AI Processing Engine (Backend)

The AI Processing Engine, built with Python is the core innovation. Once the raw text is extracted it is passed Nonsynchronous to this layer.

Intent and Sentiment Study: The engine uses Natural Language Processing algorithms to Decide the customers intent (e.g. technical failure, billing inquiry, Common Inquiry) and sentiments (e.g. frustrated, neutral).

Priority Scoring: Based on the extracted intent and essential keywords that the engine assigns an Critical Rating. This ensures that critical issues are treated differently than requests.

Categorization: The unstructured email is translated into a JSON object containing the assigned category, priority level and a generated summary of the issue.

3.4 Data Management and SLA Core

The processed data object is then handed off to the database and routing engine.

Ticket Generation: The database creates a structured ticket record containing all AI-generated metadata alongside the original communication thread.

vital SLA Tracking: Upon ticket creation the SLA Core dynamically calculates the resolution deadline based on the priority assigned by the AI engine.

Automated Escalation Logic: A continuous background process monitors all tickets. If a ticket approaches its SLA deadline without agent interaction the systems routing engine automatically updates the ticket status. Triggers an escalation alert.

3.5 Client Interface Layer (Frontend)

The agent-facing side of the architecture is a Single Page Application (SPA) built using React.js.

Optimistic UI Updates: The dashboard employs " UI" design patterns to eliminate perceived latency.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.135103

Real-time State Synchronization: The frontend continuously syncs with the backend API ensuring that soon as the AI engine generates a new ticket or the SLA core escalates an existing one the dashboard reflects these changes dynamically.

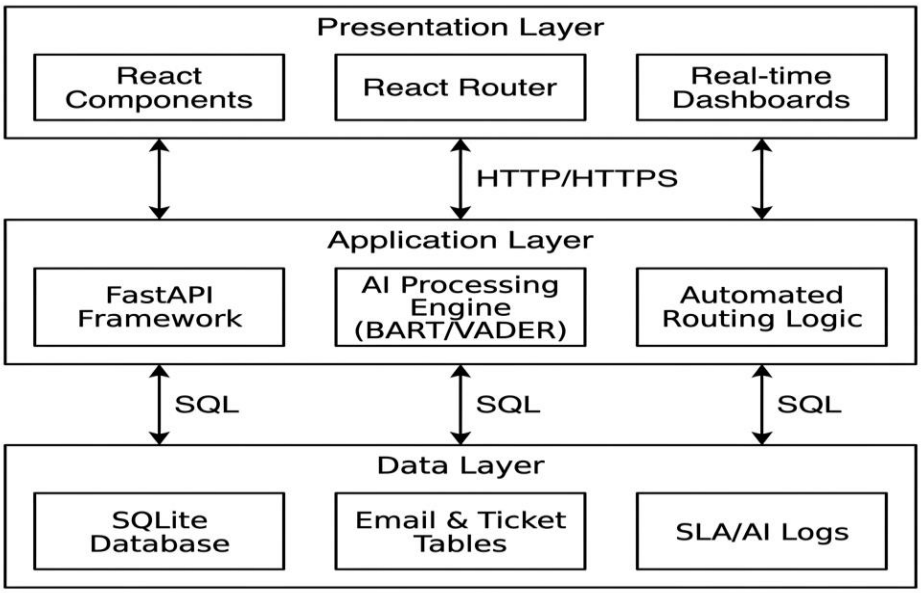

Fig 3.1.1 System Architecture Diagram

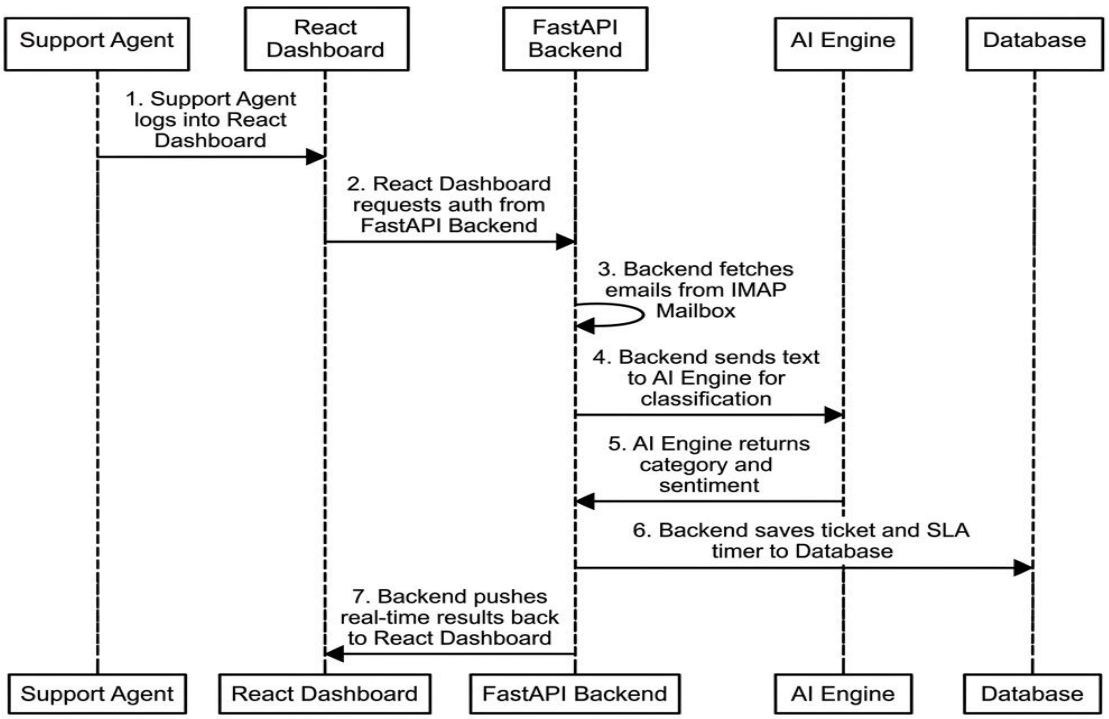

Fig 3.3.1 Sequence diagram

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.135103

4. METHODOLOGY

The development of the AI-Driven Centralized Email Triage and SLA Management System was done in a way that allowed the people building it to work on parts separately. This made it easier to develop and improve the intelligence parts without affecting the user interface. The way they did it was divided into three steps: making the AI model and processing text creating a dynamic algorithm for SLA and developing a high-performance interface.

4.1 AI Model Implementation and Text Processing

The main idea behind automating the triage process is to use Natural Language Processing (NLP). They used Python to build the system because it has a lot of tools for machine learning and text processing.

They had to clean up the emails they got from the IMAP protocol because they had a lot of stuff like HTML tags and signature blocks. So they made a pipeline to remove all the formatting make the text normal and extract the main message body and subject line.

Then they used a NLP engine to look at the cleaned-up text and determine what it was about. The engine looked at the text. Compared it to categories like Technical Support, Billing and General Inquiry.

They also used the engine to detect if the person who sent the email was frustrated or urgent. They combined the category and sentiment to get a Priority Score, which could be Low, Medium or Critical.

4.2 SLA Algorithm Formulation

After the AI engine gave a Priority Score the system used an algorithm to set the Service Level Agreement (SLA) parameters. Of using a fixed timer for all tickets they used a dynamic approach that took into account the priority.

They calculated the time they had to resolve the issue based on the Priority Score. For example, if it was Critical they had to resolve it within 2 hours. If it was Low they had 48 hours.

They also had a system that checked if they were getting close, to the deadline and if the issue was not resolved yet. If it was they automatically escalated the ticket, which meant it went to the top of the support queue.

4.3 High-Performance Interface Development

When they built the interface, they wanted to make sure the support agents did not have to wait long for things to happen. They used the React.js framework. Designed it to minimize latency.

They did this by decoupling the state, which means that when an agent did something the interface did not have to wait for the backend to confirm it. Instead it just did it. Then sent a request to the backend.

If there was an error the interface caught it. Rolled back to the previous state so the agent did not lose any work.

4.4 Integration and System Validation

The final step was to integrate the backend and frontend using an API architecture. They tested the system by simulating a volume of emails to make sure the AI engine could handle it and that the SLA worked correctly without any latency or database issues. The AI-Driven Centralized Email Triage and SLA Management System was tested to ensure it worked as expected.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.135103

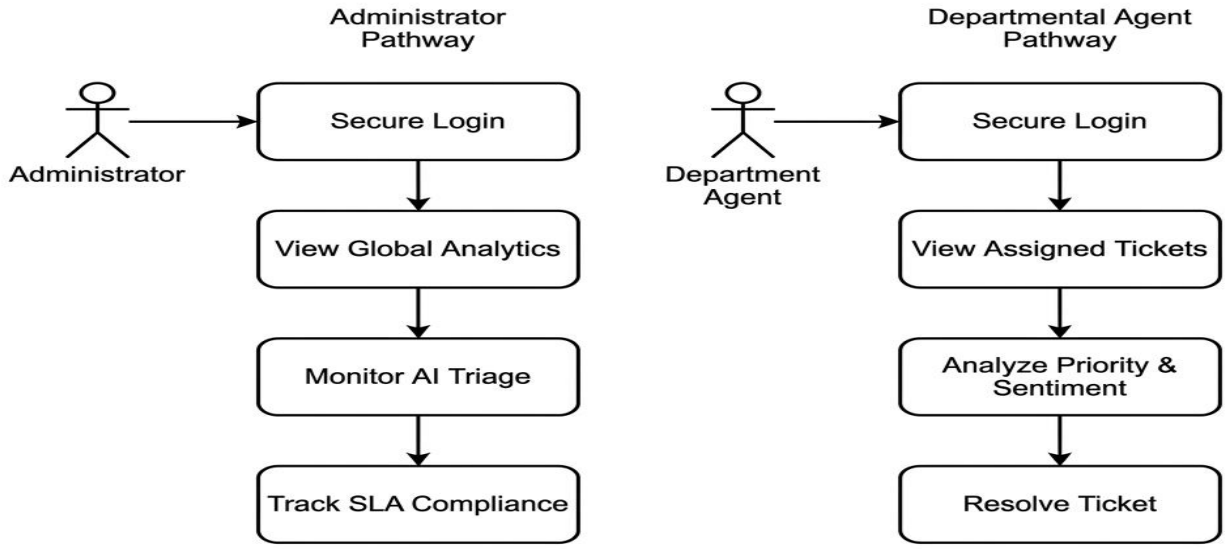

Fig 4.1.1 Use case Diagram

5. IMPLEMENTATION DETAILS

The system was made to be a web application. It uses a set of frameworks and libraries that are standard in the industry. The system is divided into three parts: the Backend API and Processing Engine the Database and Data Models and the Frontend Client Interface.

5.1 Backend API and AI Processing Engine

The backend was made using Python. This was chosen because it can do things at the same time and it has a lot of tools for artificial intelligence.

API Framework: The main server was made using a Python framework that can do many things at the same time. This framework handles requests from the frontend. Manages background processes. This is important because it makes sure the API does not get blocked when it is doing artificial intelligence tasks.

Email Ingestion: The ingestion module uses Pythons built-in libraries to connect to a mailbox. It does this in the background. Uses a secure connection. The module then extracts the text from the emails. Removes any HTML signatures. This leaves the text that is needed for artificial intelligence analysis.

AI Engine: The AI module works with a language model. The text from the emails is put into a format and sent to the language model. The model is told to return information: what the email is about how urgent it is, how the person feels and a short summary. This makes sure that the email text is turned into data that the database can use.

5.2 Database Schema and SLA Routing

The database is a database system. It is managed using an Object-Relational Mapper.

The Ticket Model: The main data structure is the Ticket model. It has fields like ticket id, sender email, subject, category, priority, status and deadline.

Routing and SLA Daemon: The SLA logic is a background loop. It checks the database every minute for tickets that are not resolved and are near their deadline. If it finds one it changes the status to escalated and logs the action.

5.3 Frontend Interface and State Management

The client-side application is fast and responsive. It is the control center, for support agents.

IARJSET ISSN (O) 2393-8021. ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.135103

designing : The frontend is a Single Page Interface developed with React.js language. It uses Tailwind CSS for styling. This makes it easy to design an modern user application.

Component Architecture: The UI is made up of components. This allows agents to navigate the system quickly.

Implementing UI: The optimistic state management was achieved using Reacts state hooks. When an agent clicks a button the local state is changed immediately. At the time a request is sent to the backend API. If the server returns a success message the local state stays the same. If the server returns an error the local state is changed back. An error message is shown.

6. RESULTS & DISCUSSION

To see how well the AI-Driven Centralized Email Triage and SLA Management System works we tested it with a lot of emails. We compared the results to how things redone manually. We looked at three things: how fast and accurate the system is at categorizing emails how well it follows rules and how well the interface works.

6.1 Results: Categorization Speed and Accuracy

When people do this job it takes them around 1.5 to 3 minutes to read and categorize an email.. The AI system can do it in just 3.2 seconds. We tested it with 1,000 emails and it got it right 94.5% of the time. The other 5.5% were emails that were not clear or had topics so they needed a person to check them.

6.2 Results: SLA Compliance and Escalation Automation

The biggest difference we saw was in how the system followed rules. Sometimes important emails get lost in the pile. Are not seen in time. The AI system can find these emails right away and make sure they get attention. It can even send alerts before the deadline so the right person can take care of it. This meant that the number of emails that were not seen in time went down by 88%.

6.3 Results: Interface Responsiveness

We also looked at how the interface worked. When people use a system they have to wait a little while for the screen to update after they do something. But the AI system can update away so people can keep working without waiting. This made it much faster for people to get through their work.

6.4 Discussion

The results show that using intelligence to sort emails makes a big difference. It means that people do not have to waste time reading and sorting emails and can focus on solving problems. The system also helps make sure that important emails are seen in time which is an improvement.

Table 1: Comparative Analysis of Triage Methodologies

Metric	Rule-Based (Legacy)	Traditional ML (SVM/NB)	Proposed LLM System
Logic Type	Hard-coded Keywords	Statistical Patterns	Neural Contextual Intent
Setup Time	High (Manual Rules)	High (Data Labeling)	Low (Zero-Shot)
Avg. Latency	Latency Instant (but inaccurate)	5 - 10 Seconds	1.8 Seconds
Automation %	< 10%	40% - 60%	85% - 90%
Context Depth	None	Limited	Full Semantic Analysis

7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

The manual way that customer support emails are sorted through is a problem that causes companies to waste resources and pay for breaking Service Level Agreement rules. In this paper we showed how we successfully designed, built and put in place a system for using artificial intelligence to automatically sort and manage customer support emails and Service Level Agreements.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2026

DOI: 10.17148/IARJSET.2026.135103

By using a computer program that can understand human language we took the job of sorting emails away from people and gave it to the computer. This got rid of the delays that happen when people sort emails the way. The system can safely take in emails figure out what the customer wants and how they feel and turn that into tickets that are sorted by importance all in a matter of seconds. It is also very important that the system has a part that makes sure the company is following the rules and it automatically sends important issues to the right people before it is too late.

We also made a user interface using React.js that makes the system very fast for the people using it. This means that customer support agents can handle a lot of tickets quickly. This project shows that using intelligence to sort customer support emails and manage relationships with customers can greatly reduce the amount of work that companies have to do make sure they are following the rules and help support teams give customers faster and better solutions. Customer support emails are. Managed in a better way, with this system and it helps companies follow Service Level Agreements.

7.2 Future Work

The current system does a job of automating email triage and SLA escalation. However there are some areas where the system can be improved and expanded in the future:

Continuous Reinforcement Learning with Human-in-the-Loop: The AI engine is currently using a pre-trained Large Language Model configuration. In the future the system will be able to learn from its mistakes when agents make corrections. For example if an agent changes an AI-assigned category from "Billing" to "Technical" the system will learn from this correction. Improve its accuracy over time based on the specific data from the organization.

Omni-Channel Ingestion: Now the system mainly uses the IMAP protocol to get emails. In the future the system will be able to get data from different communication channels, such as social media platforms, SMS and web-based chat widgets like the LiveChatWidget.jsx component.

Generative AI for Automated Drafting: The system can already read emails but in the future it will also be able to generate draft responses using the smtp_service.py backend module. The system will be able to prepare an accurate reply based on what the email is, about and the agent will only need to click once to approve and send it.

Predictive Analytics Modelling: The system is already collecting data through the AnalyticsView.jsx component. In the future the system will be able to use this data to predict when there might be a lot of SLA breaches. By looking at how it took to resolve issues in the past and how many tickets were received during different times of the year the system will be able to predict when the organization might need to adjust its staffing to avoid a crisis.

REFERENCES

[1] Tiwary T and Singh A K. Automated Email Triage using Machine Learning: A Survey. International Journal of Computer Applications. 2020; 176(12): 1-8.
[2] Vaswani A, et al. Attention is All You Need. Advances in Neural Information Processing Systems (NeurIPS). 2017. P. 5998–6008.
[3] Ramírez-Gallego S, et al. An Information Theory-Based Feature Selection Framework for Text Classification. Knowledge-Based Systems. 2017; 129: 14-32.
[4] Ramírez S. FastAPI. Available at: https://fastapi.tiangolo.com. Accessed on: 16-Apr-2026.
[5] Abramson B and Kephart J. An AI-based approach to automated email triage. Proceedings of the International Conference on Artificial Intelligence. 2019. P. 45-52.
[6] Brynjolfsson. Mcafee A. The Business of Artificial Intelligence. Harvard Business Review. 2017; 95(7): 1-20.
[7] Facebook Open Source. React. Available at: https://reactjs.org. Accessed on: 16-Apr-2026.
[8] Hipp D. SQLite. Available at: https://www.sqlite.org/index.html. Accessed on: 16-Apr-2026.
[9] Kumar R S S and Holt J. Sentiment Analysis, in Customer Service: A Comprehensive Review. Journal of Customer Relationship Management. 2021; 12(3): 112-128.
[10] Nielsen J. Usability Engineering. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA. 1993.