

Product Requirements Document



AI Agent Assistant

CONTENTS

Team Structure	3
Overview	3
Problem	3
Expert's Insight	3
Market Consideration	4
Objectives	4
Constraints	4
User Persona and ICP	5
Features And Technical Req.	6
User Journey Map	11
Model Requirement	13
Assumptions & Dependencies	14
KPIs And OKRs	14
Resources	15

Team structure:

Product Team	Yuvansh Sharma, Adhvet Sharma
Development Team	S Akash, Adhvet Sharma
Design Team	Aditya Bajar
Team Name	BajarBoys

Overview:

Development of an AI-powered ticket resolution system that leverages machine learning for **rapid analysis and resolution of incoming tickets, utilising natural language processing (NLP)** to extract relevant information. The system will employ **recommendation engines to suggest solutions based on historical data** and dynamically **learn from interactions using reinforcement learning**. By integrating with existing knowledge base repositories and employing automated response mechanisms, the system aims to **mitigate ticket escalation, optimise support workflows, and empower agents with real-time insights for more efficient problem resolution**.

Problem:

Agents struggle with ticket management, spending too much time prioritising, categorising, and finding solutions. They lack tools to quickly sort tickets by importance, mood, and type or pass them to the right team. Finding helpful articles is hard, and editing them is clunky. There's no way to measure solution effectiveness, slowing down improvement. This manual process leads to slow responses, annoyed customers, and more workload on the agents.

Expert's Insight:

"As a former product manager in yellow.ai, I've witnessed its evolution firsthand. The ticket resolution market's rapid growth is fueled by its ability to revolutionise customer support, offering cost-effective solutions with 24/7 availability. Key trends include personalised interactions and emotional intelligence integration. Despite challenges like regulatory compliance and language barriers, the future holds immense potential, with AI chatbots poised to become even more integral to businesses across sectors."

-Sriram Pingali, Product Manager Yellow.ai

Market Considerations:

PARAMETER	S1	S2	S3
Global generative AI in customer services market	Valued at USD 308.4 million in 2022	CAGR of 25.11% from 2023 to 2032	Expected to surpass USD 2,897.57 million by 2032
Business Impact and Efficiency	AI successfully automates about 70% of customer requests	Companies with AI-powered customer service grow revenue 4-8% faster	Companies lose \$75 billion to \$2 trillion annually due to low-quality customer support
Deployment and Application Trends	Chatbot segment expected to grow highest, nearly \$5 billion by 2032	Asia-Pacific: Expected fastest CAGR from 2023 to 2032	North America: >48% revenue share in 2022
Customer Behaviour and Expectations	Positive experience with AI elevates buyer satisfaction by up to 20%	Immediate response expected by 90% of customers	73% of shoppers believe AI support can positively impact their experience

Objectives:

Streamlining of ticket management to enhance team collaboration, and deliver accurate solutions, aiming to boost support efficiency and elevate customer satisfaction.

1. Deploy a classification model to automate ticket prioritisation, categorization, and sentiment analysis for ticket management
2. Implement a system for routing tickets to specialised teams based on categories, to increase efficiency and suggest estimated resolution times
3. Integrate a recommendation engine to suggest relevant articles and solutions from the knowledge base, enhancing support efficiency
4. Enable NLP driven solutions to extract and auto-generate editable responses from articles, streamlining agent workflows
5. Establish a feedback-driven scoring mechanism to assess the effectiveness of auto-generated solutions for feedback loop

Constraints:

1. Quality, quantity and bias of the data
 - a. Acceptable training time and resource requirements for model training
 - b. Minimum performance metrics (precision, recall, F1-score) for ticket classification
 - c. Ability to handle class imbalance and skewed data distributions

2. NLP and machine learning
 - a. Support for multiple languages
 - b. Ability to handle domain-specific terminology and jargon
 - c. Minimum accuracy thresholds for sentiment analysis, named entity recognition, etc.
 - d. Context retention
 - e. Computation power (GPU heavy)
 - f. Cost of computation
3. Integration with existing ticketing softwares
 - a. Integration with existing ticketing systems
 - b. Support for various document formats
 - c. Ability to handle large knowledge bases

User Personas and ICP:

Agent Persona:

Sarah, a senior customer support agent, handles around 50-60 tickets per day across various product categories. Her key responsibilities include triaging tickets, assigning priorities, routing them to relevant teams, and providing solutions. However, she faces challenges in manually categorising and prioritising tickets, which is time-consuming and error-prone. Finding relevant articles in the knowledge base is tedious and slow. Additionally, keeping up with the high ticket volume and meeting SLA targets is increasingly difficult. Sarah lacks visibility into the effectiveness of provided solutions and areas for improvement.

Product manager persona:

David is a product manager responsible for the company's customer support offering. His primary goal is to ensure high customer satisfaction and a positive support experience. He monitors customer satisfaction metrics (CSAT, FRT, NPS) and identifies areas for improvement. David aims to optimise support workflows and processes to enhance efficiency and reduce resolution times. However, he lacks real-time insights into ticket management and resolution processes. Measuring the effectiveness of support solutions and identifying knowledge gaps is difficult. David struggles to quickly adapt and improve support offerings based on customer feedback and evolving needs.

Element	Attributes	Description
Customer Profile	Company Size	Companies with 500 or more employees.
	Customer Support Team Size	Organisations with customer support teams consisting of 20 or more agents.
	Ticket Volume	Companies receiving 1,000 or more customer support tickets or inquiries per month.
	Existing Knowledge Base	Organisations with a knowledge base containing at least 5,000 articles, documentation, or solutions.

Element	Attributes	Description
	Customer Base	Companies with a customer base of 50,000 or more active users or customers.
	Historical Data	Organisations with a minimum of 2 years' worth of historical ticket data, customer interactions, and support documentation.
	Customer Satisfaction Metrics	Organisations with a current customer satisfaction (CSAT) score below 80% or a first-response time (FRT) above 24 hours, indicating room for improvement in support efficiency and customer experience.
	Multilingual Support	Companies providing customer support in 3 or more languages.
Business needs and pain-points	High ticket volume overwhelming support teams	
	Manual ticket prioritisation and categorization is slow and inefficient	
	Difficulty in quickly finding relevant knowledge base articles/solutions	
	No automated mechanism to route tickets to specialised teams	
	Lack of insight into solution effectiveness and areas for improvement	
	Slow response times leading to poor customer satisfaction	

Features and Technical requirements:

- Ticket Analysis and Categorization:

Scope:

Implement a machine learning model to automatically classify tickets based on priority, mood, and category to reduce manual categorization.

Use case:

A customer submits a support ticket regarding a technical issue. The AI classifies it as high priority, technical category, and frustrated mood, allowing the support team to prioritise and address it promptly.

Features	Flow
Preprocessing	<ul style="list-style-type: none"> Text cleaning and normalisation (e.g., removing noise, formatting from "My recent bill seems incorrect. I don't understand the charges for X service.") Tokenization (e.g., splitting the sentence into words/tokens like "bill", "seems", "incorrect", etc.) Word embedding (e.g., converting tokens to numerical vectors using techniques like Word2Vec or GloVe)
Encoder Layer	<ul style="list-style-type: none"> Multiple stacked encoder layers Self-Attention sublayer <ul style="list-style-type: none"> Analyses relationships between words in the sentence (e.g., "bill" is related to "incorrect" and "charges" in the given sentence) Considers context and influence of words on each other's meaning (e.g., "bill" becomes more relevant when considering "incorrect" and "charges") Feed Forward Network sublayer <ul style="list-style-type: none"> Adds non-linearity to the model Captures complex relationships in the data (e.g., learning deeper connections between "bill", "incorrect", "charges", and "service") Refines understanding from the Self-Attention sublayer
Sentence Understanding	<ul style="list-style-type: none"> Sentence travels through multiple encoder layers (e.g., the given sentence goes through multiple layers) Each layer builds upon the understanding from the previous layer Model learns intricate relationships between words (e.g., progressing from understanding individual words to comprehending the overall intent of a billing issue) Progresses from basic word understanding to comprehensive sentence interpretation
Output Vector	<ul style="list-style-type: none"> Final encoder layer generates an output vector representing the processed sentence Captures the overall meaning and sentiment (e.g., the output vector represents the customer's confusion about an incorrect bill and charges for a specific service)
Downstream Applications	<ul style="list-style-type: none"> Classification <ul style="list-style-type: none"> Output vector fed into a classifier Predicts the ticket category (e.g., classifier may categorise the email as a "Billing Issue" based on the output vector) Question Answering <ul style="list-style-type: none"> Model used to answer specific questions (e.g., "What is the disputed charge?" - the model could identify the charge for "X service" based on the sentence understanding)

- **Specialised Team Forwarding:**

Scope:

Ensure tickets are handled by the most suitable team, improving resolution times.

Use case:

A ticket related to billing queries is automatically forwarded to the finance team, ensuring it is addressed by experts in that domain.

Features	Flow
Ticket Content Analysis	<ul style="list-style-type: none"> ● Integrate LLaMA language model for ticket content processing ● Generate output vector from LLaMA model's final encoder layer representing sentence meaning and sentiment
Clustering and Team Assignment	<ul style="list-style-type: none"> ● Implement clustering algorithms (e.g., Gaussian Mixture Models, Spectral Clustering) ● Process LLaMA model's output vector through clustering algorithms ● Determine most suitable team or cluster for handling the ticket based on clustering results
Rule-based Routing	<ul style="list-style-type: none"> ● Develop rule definition and management system ● Define complex rules and conditions for ticket routing ● Support dynamic routing based on real-time conditions ● Evaluate rules based on ticket content, clustering results, and real-time conditions
Ticketing System Integration	<ul style="list-style-type: none"> ● Integrate with ticketing system APIs/SDKs ● Automatically forward tickets to assigned teams based on clustering and rule evaluation results

- **Knowledge Base Recommendation Engine:**

Scope:

Providing agents with quick access to relevant knowledge base articles can significantly expedite ticket resolution.

Use case:

An agent receives a ticket about product troubleshooting. The system recommends relevant articles and guides to assist the agent in resolving the issue efficiently.

Features	Flow
Text Extraction Module	<ul style="list-style-type: none"> Retrieves content from customer tickets, including problem descriptions and inquiries. Extracts relevant text from the organisation's knowledge base articles in various formats (HTML, PDF, Word, etc.).
Text Preprocessing	<ul style="list-style-type: none"> Ensures uniformity and consistency in language and format of the extracted text data.
Text Indexing And Search Engine Integration	<ul style="list-style-type: none"> Indexes the preprocessed text data using a search engine like Elasticsearch. Enables rapid retrieval of relevant articles based on search queries or ticket content.
Recommendation Engine	<ul style="list-style-type: none"> Retrieves pertinent articles from the indexed knowledge base based on the ticket content. Ranks the articles using advanced relevance ranking algorithms like BM25. Prioritises the most suitable resources for addressing customer concerns.
Multilingual Support	<ul style="list-style-type: none"> Identifies the language of the customer ticket. Retrieves knowledge base articles in the corresponding language for agents. Provides multilingual support for handling customer inquiries.
Integration with Llama Model	<ul style="list-style-type: none"> Utilises the final output vector from the Llama Model to determine the content of the ticket. Enables accurate retrieval of relevant articles based on the model's understanding of the ticket content.

- **Solution Generation:**

Scope:

Implement natural language processing (NLP) to extract solutions from articles and auto-generate editable responses.

Use case:

An agent selects a suggested article, the system extracts relevant information and generates a response, which the agent can edit before sending to the customer.

Features	Flow
Analyse passages from top-ranked documents	<ul style="list-style-type: none"> Extract specific answers to the user's question

Utilise NLP techniques	<ul style="list-style-type: none"> • Named entity recognition (NER) • Dependency parsing • Identify entities and relationships within text • Aid in answer extraction
Present extracted answers	<ul style="list-style-type: none"> • Readable format for user • Include relevant context from original documents
User review and access	<ul style="list-style-type: none"> • User can review presented answers • Access source documents for additional information if needed

- **Feedback Score:**

Scope:

Establish a feedback mechanism to evaluate the effectiveness of auto-generated solutions.

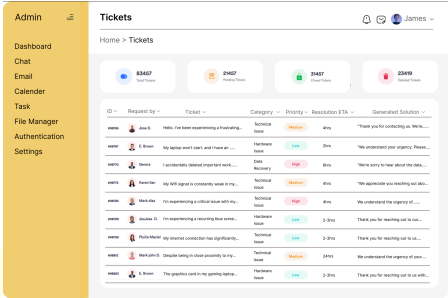
Use case:

After resolving a ticket, agents provide feedback on the usefulness of the suggested solutions and knowledge base articles, which is then used to improve the underlying models and enhance future recommendations.

Features	Flow
Answer Accuracy Evaluation	<ul style="list-style-type: none"> • Compare extracted answers with ground truth • Initial human evaluation, then automate
Answer Relevance Assessment	<ul style="list-style-type: none"> • Analyze match between answers and user query • Semantic similarity, keyword matching, contextual understanding
Response Completeness Checking	<ul style="list-style-type: none"> • Decompose user query into sub-queries • Verify all sub-queries are answered
Timeliness Measurement	<ul style="list-style-type: none"> • Track speed of response delivery
User Satisfaction Feedback	<ul style="list-style-type: none"> • Gather feedback via surveys, ratings • Incorporate feedback into agent learning
Hallucination Detection	<ul style="list-style-type: none"> • Identify unsupported or inaccurate information • Compare against input data and knowledge base

User Journey Map:

SECTION SUB-SECTION USER STORY & EXPECTED BEHAVIOURS SCREENS

<p>Dashboard</p>	<p>All tickets view</p>	<p>As a new user, I face a technical issue with the product and raise a ticket for the same</p> <p>Expected behaviour:</p> <ul style="list-style-type: none"> • User should be able to submit a ticket via existing CRM • The software automatically ingests the new ticket, extracting relevant information such as customer details, subject line, description, and any attachments • Tickets are received and automatically classified by the AI based on priority, mood, and category • The new ticket entry is displayed in the dashboard, with a brief summary including customer name, issue category, priority and mood 	
<p>Ticket Details</p>	<p>Ticket details with relevant articles</p>	<p>As an agent, I want to open the ticket details for self resolution or directing to relevant department</p> <p>Expected behaviour:</p> <ul style="list-style-type: none"> • The agent selects the ticket from the dashboard, opening the detailed ticket view. • The ticket details view displays the full ticket information, including customer information, ticket content, suggested knowledge base articles, recommended solutions • Agents can click on recommended articles within the UI to view their content and relevance to the ticket 	

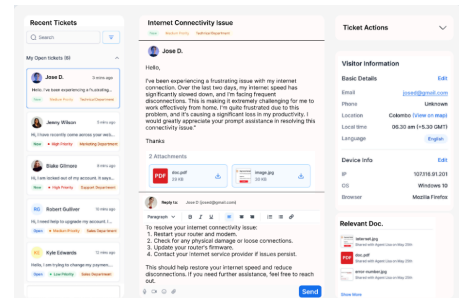
SECTION	SUB-SECTION	USER STORY & EXPECTED BEHAVIOURS	SCREENS
---------	-------------	----------------------------------	---------

Solution generation from article (response editor)

After finding a relevant article, I now want to compose a solution based on the article

Expected behaviour:

- Agent can extract pertinent information from selected articles and auto-generate editable responses within the UI
- The response editor allows the agent to format the response, insert knowledge base article links or attachments, and utilise predefined response templates if applicable
- The agent can now submit the response after editing the recommended solution



Feedback collection

I am prompted with the feedback form

Expected behaviour:

- A feedback mechanism, such as a rating system or comment box, is presented to the agent, enabling them to provide feedback on the AI-driven assistance provided
- The model ingests the rating and adjusts the weights accordingly

Model Requirements:

The LLAMA2 model demonstrates impressive performance metrics across various dimensions, making it an optimal choice for classification tasks. With a precision of 0.89 and a recall of 0.55, LLAMA2's safety model effectively filters out unsafe responses, ensuring that generated outputs meet stringent safety standards. Additionally, LLAMA2 achieves remarkable gains in both median and maximum performance metrics, indicating consistent improvement in response quality. Through iterative fine-tuning techniques, LLAMA2 achieves significant enhancements, with each Proximal Policy Optimization (PPO) iteration leveraging a batch size of 512 and a PPO clip threshold of 0.2. These iterative refinements contribute to LLAMA2's superior performance, with its Ghost Attention method further enriching context and enhancing response coherence. Overall, LLAMA2's robust architecture and iterative refinement processes culminate in

outstanding performance metrics, making it the preferred choice for classification tasks demanding precision, recall, and contextual relevance.

Feature	Spec	Justification
Context Window	128K+	Large data volume handling required.
Modalities	Text, Vision	We want users to be able to upload images and converse with them.
Fine-Tuning Capability	Required	Fine-tuning the model with proprietary data is essential for achieving optimal performance.
Latency	High Priority	Given the real-time nature of the product, speed is incredibly important. We are willing to spend more for a faster model as we believe it will lend itself to a positive ROI.
Precision	0.89	LLAMA2 demonstrates impressive precision in filtering unsafe responses.
Recall	0.55	LLAMA2's recall metric ensures effective identification of safe responses.
Fine-Tuning Technique	Iterative PPO	Iterative fine-tuning with PPO (batch size: 512, clip threshold: 0.2) enhances performance.

Assumptions & Dependencies:

- Open-source NLP libraries like Hugging Face Transformers, LlamaCPP, Kaleido, Elastic search , Rank-BM25.
- Access to cloud services and infrastructure for deploying and scaling the solution.
- Data cleaning and preprocessing processes to handle inconsistencies or errors
- Access to ticket data from the existing ticketing system or other sources
- The product can be integrated with the organisation's existing ticketing system and other relevant systems
- Sufficient computing resources are available for training and deploying machine learning models, and running other components.

KPIs and OKRs:

- Average Ticket Resolution Time: Achieve a 20% reduction in average ticket resolution time within the first 6 months of implementation.
- Reduce overall support costs by 10% within the first 18 months of implementation.
- Increase agent productivity by 25% within the first year of implementation, as measured by the number of tickets resolved per agent.
- Increase agent productivity by 25% within the first year of implementation, as measured by the number of tickets resolved per agent.
- First Contact Resolution Rate: Track the percentage of tickets that are resolved on the first interaction with the customer, without the need for follow-up or escalation. Increase the first contact resolution rate by 15% within the first year of implementation.
- Customer Satisfaction (CSAT) Score: Maintain an average CSAT score of 4.5 or higher (on a scale of 1-5) during the first year of implementation.
- Knowledge Base Utilisation: Achieve a 50% knowledge base utilisation rate within the first 6 months of implementation.
- Agent Feedback Score: Achieve an average agent feedback score of 4 or higher (on a scale of 1-5) within the first year of implementation.

Resources:

- Github repository:
<https://github.com/akash1764591/product-a-thon>
- Training Dataset:
https://github.com/akash1764591/product-a-thon/blob/main/Ticket_data.csv
- Prototype:
<https://drive.google.com/file/d/1wiL7n11VQ9XY-oEPxgH3BpTZICGCTHIX/view?usp=sharing>
- Architectures and models:
<https://www.sciencedirect.com/science/article/pii/S0957417422010776>
Other reference articles:
<https://blog.stackademic.com/revolutionizing-crm-with-gpt-and-llm-a-comprehensive-overview-14e236dcad68>
- <https://masterofcode.com/blog/ai-in-customer-service-statistics>
- <https://www.tidio.com/blog/ai-customer-service-statistics/>
- <https://www.precedenceresearch.com/call-center-ai-market>
- https://huggingface.co/spaces/akash88/predibase-customer_support