Title Page

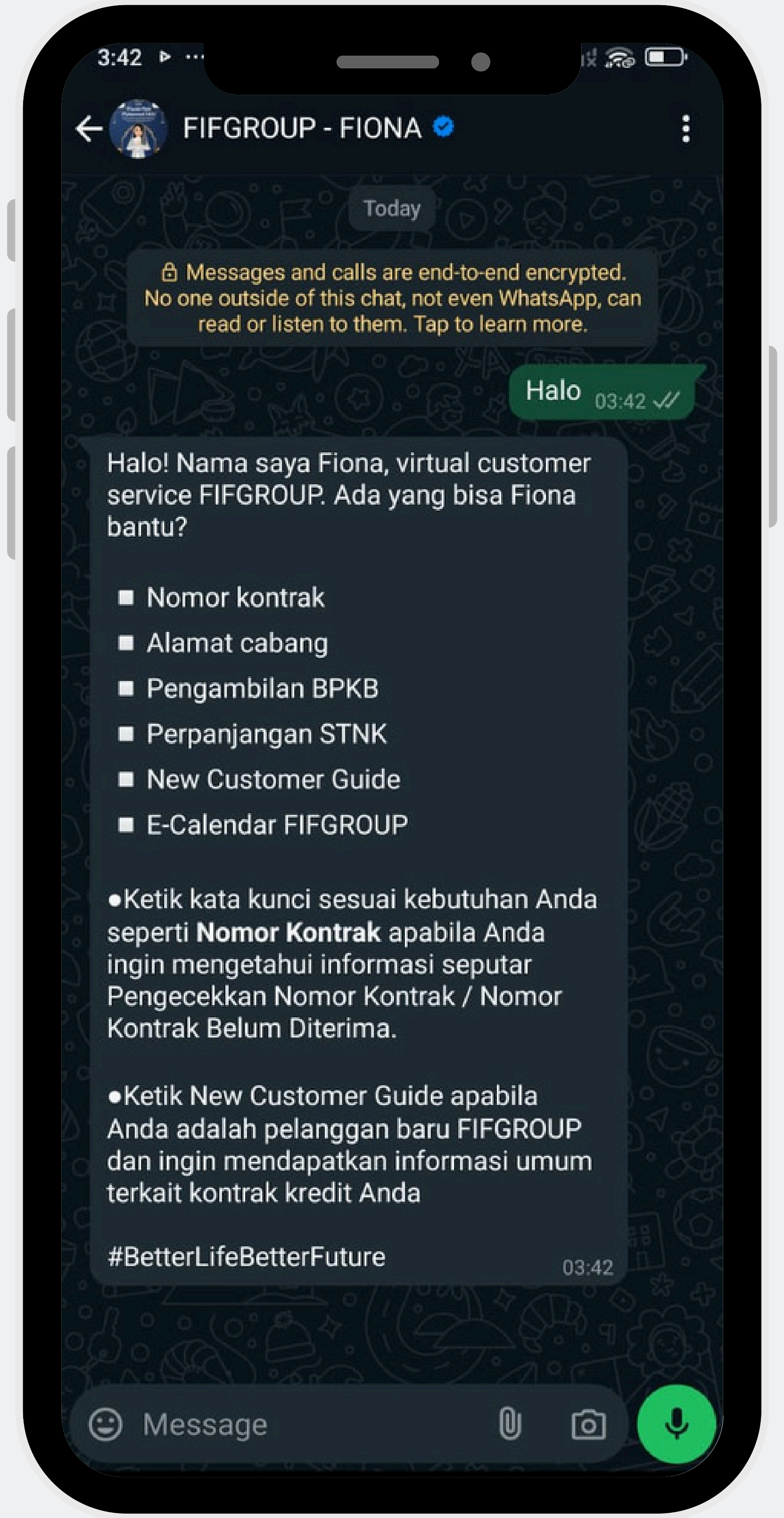
FIFGROUP - PT. Federal International Finance

Fiona 2.0 Project Proposal

Mathew Darren Kusuma - 68772

What is Fiona?

Fiona is a WhatsApp-based customer service chatbot developed by FIFGROUP to assist users with inquiries and provide support through pre-prepared options. Designed to streamline communication, Fiona offers quick responses by presenting customers with multiple-choice menus.



🔒 Messages and calls are end-to-end encrypted. No one outside of this chat, not even WhatsApp, can read or listen to them. Tap to learn more.

Halo 03:42

Halo! Nama saya Fiona, virtual customer service FIFGROUP. Ada yang bisa Fiona bantu?

- Nomor kontrak
- Alamat cabang
- Pengambilan BPKB
- Perpanjangan STNK
- New Customer Guide
- E-Calendar FIFGROUP

●Ketik kata kunci sesuai kebutuhan Anda seperti **Nomor Kontrak** apabila Anda ingin mengetahui informasi seputar Pengecekkan Nomor Kontrak / Nomor Kontrak Belum Diterima.

●Ketik New Customer Guide apabila Anda adalah pelanggan baru FIFGROUP dan ingin mendapatkan informasi umum terkait kontrak kredit Anda

#BetterLifeBetterFuture 03:42

Fiona’s Limitations

- Limited Functionality with Pre-prepared Options
- Lack of Agent Escalation
- Inability to Handle Open-ended Conversations
- Potential Loss of Customer Satisfaction

Title Page Introduction Overview

Insert your topic here

Overview

User Interview

User Stories

Roadmap

Goal: Identify Issues with Limited Functionality and Pre-prepared Options

Questions

1. **How often do you find the pre-prepared options provided by Fiona limiting?**
   - **Answer Example**: "I often feel like none of the options truly address my specific question."
   - **Insight**: Highlights the need for more dynamic and flexible response capabilities.
2. **Can you describe a situation where the available options didn’t meet your needs?**
   - **Answer Example**: "I wanted to know about a specific promotion, but the chatbot only had general information."
   - **Insight**: Identifies specific gaps in the knowledge base that require expansion.
3. **What type of information would you like Fiona to provide beyond the pre-prepared options?**
   - **Answer Example**: "I want to ask more specific questions about my account or services."
   - **Insight**: Reveals user desires for open-ended inquiries and personalized responses.

Goal: Understand the Lack of Agent Escalation in the Current System

Questions

1. **Have you ever felt frustrated by the lack of options to escalate to a human agent?**
   - **Answer Example**: "Yes, when Fiona couldn't answer my question, I had to search for a contact number."
   - **Insight**: Points to the urgency for an effective escalation process.
2. **What would make the escalation process more seamless for you?**
   - **Answer Example**: "I’d prefer a button to directly request a human agent when I’m not getting the help I need."
   - **Insight**: Provides actionable feedback for improving the escalation pathway.
3. **Have you experienced negative outcomes due to the inability to escalate your issues?**
   - **Answer Example**: "Definitely, I ended up abandoning my inquiry out of frustration."
   - **Insight**: Emphasizes the importance of having human support available.

Goal: Assessing User Interactions with Open-Ended Queries

Questions

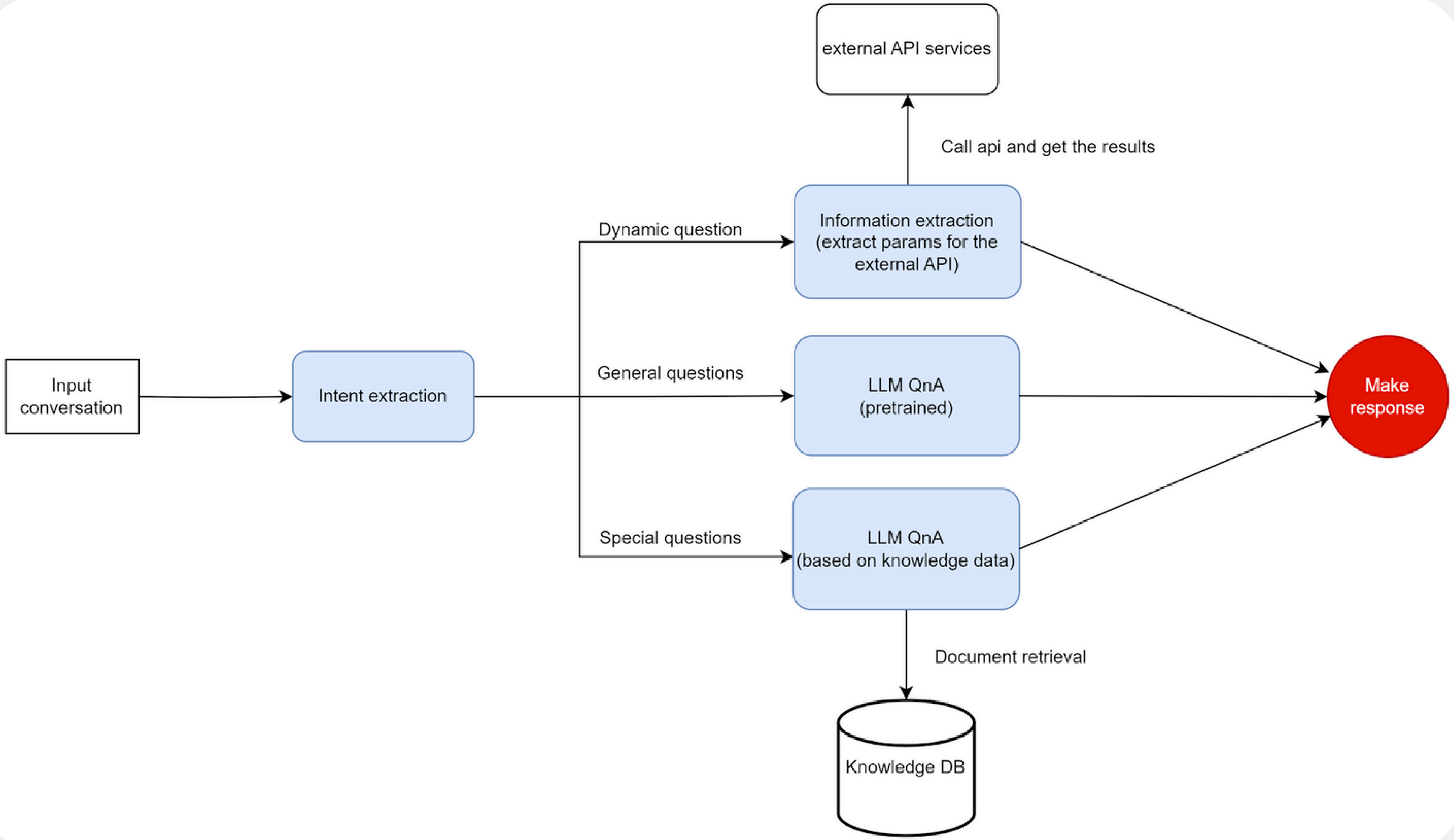
1. **How do you feel when Fiona cannot engage in open-ended conversations?**
   - **Answer Example**: "It feels like I’m talking to a wall; I can’t really explore my questions."
   - **Insight**: Illustrates user frustration with the lack of conversational depth.
2. **Can you share an instance where Fiona's inability to handle an open-ended question affected your experience??**
   - **Answer Example**: "When I asked for advice on choosing a loan, it just gave me a link instead of helping."
   - **Insight**: Identifies specific scenarios where more interactive engagement is needed.
3. **What improvements would you suggest for Fiona to engage in more open-ended dialogues?**
   - **Answer Example**: "Maybe include more follow-up questions or ask if I need further assistance."
   - **Insight**: Offers clear directions for enhancing conversational flow.

Goal: Gauge the Potential Loss of Customer Satisfaction

Questions

1. **How would you rate your overall satisfaction with your interactions with Fiona?**
   - **Answer Example**: "I’d say a 4 out of 10—mostly because it doesn’t meet my needs."
   - **Insight**: Provides a quantifiable measure of user satisfaction.
2. **What specific aspects of Fiona’s performance lead to dissatisfaction?**
   - **Answer Example**: "Limited options and the inability to talk to a real person when needed are frustrating."
   - **Insight**: Clearly identifies areas requiring improvement.
3. **What features or changes would significantly improve your satisfaction with Fiona?**
   - **Answer Example**: "If it could answer follow-up questions better and allow me to escalate to an agent."
   - **Insight**: Offers insights into features that could enhance user experience and satisfaction.

Epic 1: Fine-Tune the LLM



Epic 1: Fine-Tune the LLM

Story 1.1: Fine-tune LLM using FAQs for Commonly Asked Questions

User Story

As a customer service team, I want Fiona to be fine-tuned on our FAQs so the chatbot can provide accurate, immediate answers to common questions without overwhelming the user with too much information.

Backlog Preparation

- Collect existing FAQs and sort them by relevance and frequency.
- Prepare a versioning system to track FAQ updates and re-tune the model periodically.

Tasks

- Gather the most common customer queries from historical FAQ data.
- Fine-tune the LLM by adjusting its weights based on these FAQs to ensure relevant responses.
- Conduct testing rounds with the customer service team to validate the response accuracy for frequent queries.

Done Criteria

- The LLM provides correct answers to 95% of frequently asked questions.
- Customer satisfaction scores improve for simple queries.

Story 1.2: RAG Using Knowledge Documents for Business-Specific Information

User Story

As a customer service team, I want Fiona to access information from business-specific knowledge documents like policy files and product offers, so it can respond to complex questions that FAQs alone cannot cover.

Backlog Preparation

- Collect product policy documents, offers, and other relevant knowledge artifacts.
- Work with IT to ensure data storage in a vector database that supports similarity search.
- Create fallback responses in case relevant documents aren’t found.

Tasks

- Implement a retrieval-augmented generation (RAG) model, where the chatbot retrieves relevant information from external documents before generating a response.
- Use vector embeddings and similarity search to identify the most relevant passages from policies and offers.
- Integrate a vector database to store and manage document embeddings for dynamic retrieval.
- Collaborate with domain experts to tag key sections from documents that customers often inquire about.

Done Criteria

- The chatbot accurately retrieves relevant knowledge in at least 95% of queries that require document-based responses.
- User feedback confirms improved accuracy for more complex inquiries.

Story 1.3: Personalization & Real-Time Responses with Dynamic Data Integration

User Story

As a customer, I want Fiona to provide up-to-date information on product availability, pricing, or promotions so I can make real-time decisions while interacting with the chatbot.

Tasks

- Integrate APIs to pull dynamic data like current inventory levels, product prices, and ongoing promotions.
- Use a hybrid approach: store static product data (e.g., specifications) in a vector database and access dynamic data (e.g., inventory) from APIs in real-time.
- Implement logic to update responses when inventory or pricing changes mid-conversation.
- Test response accuracy under various scenarios, such as high demand periods where stock changes frequently.

Backlog Preparation

- Work with the product team to define which static product attributes will be stored in the vector DB.
- Coordinate with backend developers to ensure seamless API connections.
- Develop fallback logic for when API services are down or unresponsive.

Done Criteria

- API integrations work smoothly, with 100% accuracy in retrieving real-time inventory and pricing data.
- User satisfaction increases as reflected in feedback surveys.

Epic 2: Database Creation and Management

Story 2.1: Database Creation and Management

User Story

As a backend engineer, I want to build a database that stores user interactions and relevant embeddings, so Fiona can efficiently retrieve past conversations and provide personalized responses.

Tasks

- Design a relational or NoSQL schema to store user interactions, embeddings, and relevant metadata.
- Integrate a vector database (e.g., Pinecone, Weaviate) to store ML model embeddings for similarity-based search.
- Set up ETL pipelines to periodically clean, update, and archive stored data.
- Implement indexing mechanisms to optimize query performance for fast retrieval during conversations.

Backlog Preparation

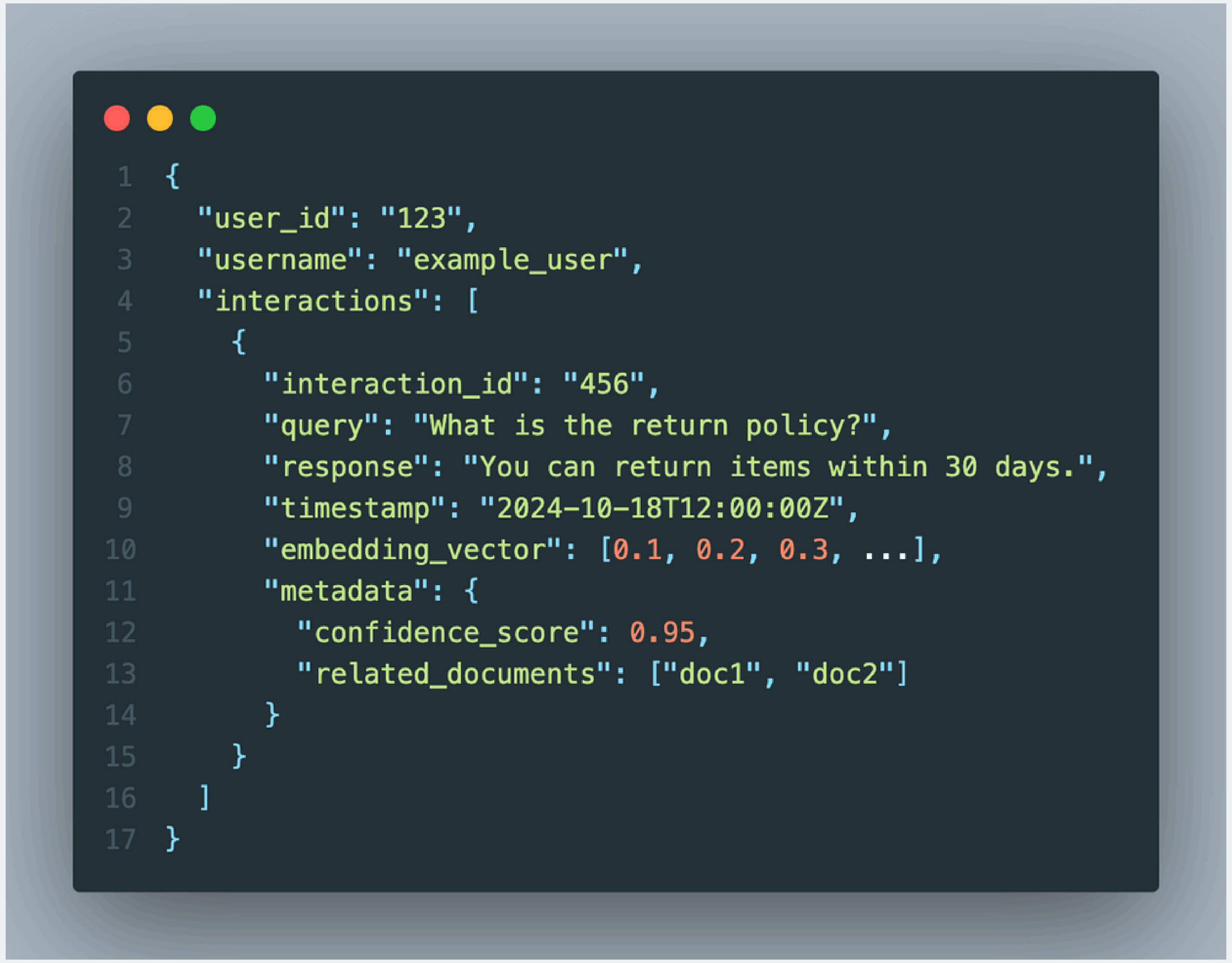
- Collaborate with IT to provision storage resources.
- Ensure that all sensitive data complies with security policies.
- Prepare fallback logic to handle data retrieval failures gracefully.

Done Criteria

- Database responds within sub-second latency for standard queries.
- All essential user data is securely stored with automated backups enabled.

Epic 2: Database Creation and Management

Story 2.1: Database Creation and Management



```
{
  "user_id": "123",
  "username": "example_user",
  "interactions": [
    {
      "interaction_id": "456",
      "query": "What is the return policy?",
      "response": "You can return items within 30 days.",
      "timestamp": "2024-10-18T12:00:00Z",
      "embedding_vector": [0.1, 0.2, 0.3, ...],
      "metadata": {
        "confidence_score": 0.95,
        "related_documents": ["doc1", "doc2"]
      }
    }
  ]
}
```

Epic 3: Model Deployment and API Integration

Story 3.1: Model Deployment and API Integration

User Story

As a machine learning engineer, I want to deploy trained models and integrate them into Fiona, so the system can generate predictions and responses in real-time.

Backlog Preparation

- Prepare CI/CD pipelines for automated model deployment and monitoring.
- Collaborate with domain experts to validate the deployed model's predictions.

Tasks

- Containerize the model using Docker or Kubernetes for seamless deployment.
- Set up a model registry to track versions and updates.
- Create REST or GraphQL APIs to expose the model for inference.
- Integrate the model's API into Fiona's backend to ensure real-time predictions are available during conversations.

Done Criteria

- Models are deployed and accessible via APIs with 99% uptime.
- The model produces accurate responses with an average latency of <500ms.

Epic 4: Monitoring and Continuous Improvement

Story 4.1: Monitor Chatbot Performance Through Analytics and Feedback

User Story

As a customer service team, I want to track Fiona’s performance using analytics tools so I can identify areas of improvement and maintain high user satisfaction.

Tasks

- Set up analytics dashboards to monitor metrics like:
- Collect user feedback within chat sessions (e.g., thumbs up/down).
- Set alerts for abnormal performance metrics (e.g., spikes in escalations).
- Identify conversation patterns that indicate confusing chatbot responses.
- Share monthly performance reports with the customer service team for review.

Backlog Preparation

- Coordinate with the analytics team to identify key metrics and set up a dashboard.
- Collaborate with product owners to plan monthly chatbot performance reviews.

Done Criteria

- Analytics dashboards show accurate metrics with at least 95% uptime.
- At least 2 performance improvements are identified and implemented every month.
- Customer satisfaction scores improve by 10% within the first quarter.

Epic 4: Monitoring and Continuous Improvement

Story 4.2: Periodically Retrain the LLM Using New Data

User Story

As a data scientist, I want to retrain Fiona periodically with recent interaction data to keep it updated and better aligned with evolving customer inquiries.

Tasks

- Extract chat logs and identify trends in questions not answered correctly.
- Add these new trends to the training data and fine-tune the LLM.
- Identify and address recurring knowledge gaps through new FAQs or knowledge documents.
- Conduct validation tests on the updated model before deploying it.

Backlog Preparation

- Set up automated scripts to extract chat logs every month.
- Coordinate with business teams to gather new data (like updated policies or products) in time for retraining.
- Build a sandbox environment to test updated models safely before deployment.

Done Criteria

- Chatbot retraining happens at least once per quarter.
- Test results show at least a 90% improvement in previously problematic queries.

Epic 5: Smooth Handoff to Human Agents

Story 5.1: Escalation to Agent When Customer is Dissatisfied

User Story

As a customer, I want to be able to escalate to a human agent when I’m dissatisfied with Fiona’s response, so I can receive personalized support.

Tasks

- Implement a “Speak to an Agent” option whenever a user expresses dissatisfaction (e.g., negative sentiment).
- Ensure the conversation context is transferred to the agent seamlessly.
- Develop real-time notifications to alert agents of incoming escalations.

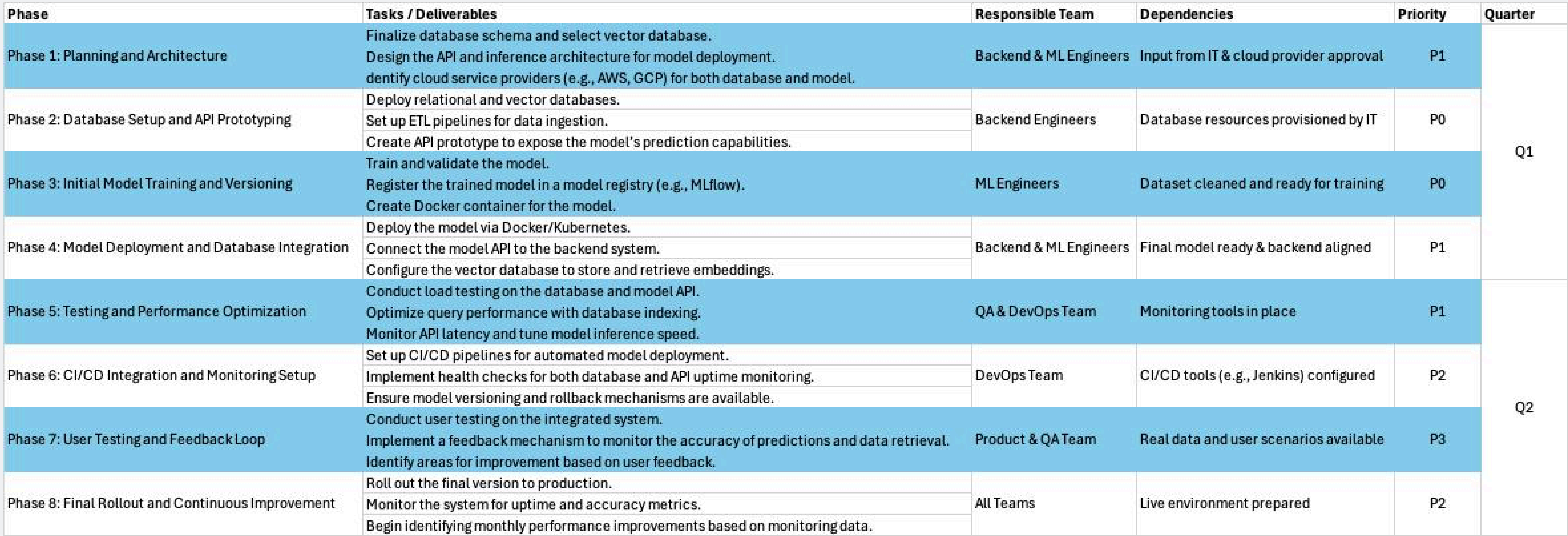
Backlog Preparation

- Backlog Preparation:
- Set up sentiment analysis models to detect dissatisfaction automatically.
- Configure chatbot settings to enable smooth transitions from LLM to agent.
- Prepare agents by training them on how to handle escalated cases efficiently.

Done Criteria

- 100% of escalated cases include complete conversation context for the agent.
- Customer satisfaction scores for escalated cases increase by at least 10%.

Roadmap



Phase	Tasks / Deliverables	Responsible Team	Dependencies	Priority	Quarter
Phase 1: Planning and Architecture	Finalize database schema and select vector database. Design the API and inference architecture for model deployment. dentify cloud service providers (e.g., AWS, GCP) for both database and model.	Backend & ML Engineers	Input from IT & cloud provider approval	P1	Q1
Phase 2: Database Setup and API Prototyping	Deploy relational and vector databases. Set up ETL pipelines for data ingestion. Create API prototype to expose the model's prediction capabilities.	Backend Engineers	Database resources provisioned by IT	P0	
Phase 3: Initial Model Training and Versioning	Train and validate the model. Register the trained model in a model registry (e.g., MLflow). Create Docker container for the model.	ML Engineers	Dataset cleaned and ready for training	P0	
Phase 4: Model Deployment and Database Integration	Deploy the model via Docker/Kubernetes. Connect the model API to the backend system. Configure the vector database to store and retrieve embeddings.	Backend & ML Engineers	Final model ready & backend aligned	P1	
Phase 5: Testing and Performance Optimization	Conduct load testing on the database and model API. Optimize query performance with database indexing. Monitor API latency and tune model inference speed.	QA & DevOps Team	Monitoring tools in place	P1	Q2
Phase 6: CI/CD Integration and Monitoring Setup	Set up CI/CD pipelines for automated model deployment. Implement health checks for both database and API uptime monitoring. Ensure model versioning and rollback mechanisms are available.	DevOps Team	CI/CD tools (e.g., Jenkins) configured	P2	
Phase 7: User Testing and Feedback Loop	Conduct user testing on the integrated system. Implement a feedback mechanism to monitor the accuracy of predictions and data retrieval. Identify areas for improvement based on user feedback.	Product & QA Team	Real data and user scenarios available	P3	
Phase 8: Final Rollout and Continuous Improvement	Roll out the final version to production. Monitor the system for uptime and accuracy metrics. Begin identifying monthly performance improvements based on monitoring data.	All Teams	Live environment prepared	P2	

Title Page | Introduction | Overview | User Interview | User Stories | Roadmap | Thank You

Insert your topic here

print(“Thank You!”)