**Technical Explanation Document**

**AI-Powered Support Copilot**

**Project Objective**

The goal is to create an AI-powered Support Copilot that assists customer support executives with real-time issue analysis, response recommendations, conversation summarization, and monitoring unattended issues. The system is designed as RESTful APIs for easy integration with existing support portals.

**Technology Stack Justification**

|  |  |  |
| --- | --- | --- |
| **Component** | **Choice** | **Reason** |
| API Framework | FastAPI | High-performance, asynchronous, built-in OpenAPI docs, Python-based. |
| Hosting (Prototype) | Google Colab + Ngrok | Quick, serverless testing and public API exposure without infrastructure setup. |
| Deployment (Prod) | AWS (EC2 / Lambda) | Scalable, cloud-native, with high availability and security features. |
| API Documentation | Swagger / OpenAPI | Industry-standard, interactive API documentation for developers. |
| Database | MySQL / Amazon RDS | Structured data storage, easy scaling, relational management of issues/customers. |
| AI Models | Optional: LLM APIs (OpenAI / AWS Bedrock) | Future enhancement for NLP tasks like summarization and template generation. |

**System Architecture**

* **User Layer:** Support Executives via Support Portal
* **API Layer:** REST APIs built using FastAPI
* **Processing Layer:** NLP/LLM services (planned), Business Logic
* **Database Layer:** MySQL / Amazon RDS (planned for full deployment)
* **Security Layer:** OAuth 2.0 or JWT (to be integrated before production)

(Architecture diagram attached separately.)

**Design Choices Explained**

**a. RESTful APIs**

* APIs allow easy integration with existing support systems.
* REST ensures scalability, modularity, and statelessness.

**b. FastAPI**

* Simplifies building APIs using Python.
* Built-in OpenAPI generation for documentation.
* Async support ensures fast responses.

**c. Cloud Deployment (AWS Recommended)**

* Enables scaling to handle increasing issue volumes.
* Offers managed database services (RDS) and security options (IAM roles, VPC).

**d. Data Privacy & Security**

* Future versions will include authentication (OAuth 2.0 / JWT).
* HTTPS ensures secure data transmission.
* Storage to comply with GDPR and other regulations.

**API Functional Flow**

1. Support Portal raises new issue →
2. API analyzes issue (severity, similar past issues) →
3. System recommends responses and templates →
4. Conversation automatically summarized →
5. Monitoring APIs track unattended critical issues.

**Future Enhancements**

* Replace dummy responses with real database connections.
* Integrate LLM APIs for generating message templates and summaries.
* Migrate APIs from Colab to AWS Lambda / EC2 for production.
* Implement proper authentication and user access control.
* Extend monitoring APIs for automated SLA enforcement.

**Conclusion**

This modular API-based design ensures scalability, real-time performance, and easy integration into existing support ecosystems, making it a future-proof solution for enhancing customer support operations using AI.