



GSoC Proposal for BeagleBoard.org



Table of contents

1	Introduction	1
1.1	Summary links	1
1.2	Status	1
1.3	Proposal	1
1.4	About	1
2	Project	2
2.1	Description	2
2.1.1	Technical Implementation	2
2.2	LLM Fine-tuning Architecture	2
2.3	RAG Integration Pipeline	2
2.4	Hosting Infrastructure	3
2.5	Deployment Targets	3
2.6	Evaluation Framework	3
2.7	Software	4
2.8	Hardware	4
3	Architecture and Diagrams	5
4	Timeline	8
4.1	Detailed Timeline	8
4.1.1	Community Bonding (May 9 - May 26)	8
4.1.2	Milestone 1: Foundation (June 3)	8
4.1.3	Milestone 2: Data Preparation (June 17)	9
4.1.4	Milestone 3: Model Training (July 1)	9
4.1.5	Midterm Evaluation (July 8)	9
4.1.6	Milestone 4: Agentic Evaluation (July 22)	9
4.1.7	Milestone 5: Web Interface (Aug 5)	10
4.1.8	Final Submission (Aug 19)	10
4.2	Benefit	10
5	Experience and Approach	11
5.1	Personal Background	11
5.2	Experience	11
5.3	Contingency	11
5.4	Misc	12
5.5	References	12

Chapter 1

Introduction

1.1 Summary links

- **Contributor:** [Fayez Zouari](#)
- **Mentors:** [Jason Kridner](#), [Aryan Nanda](#) , [Kumar Abhishek](#)
- **Code:** [BeagleMind](#)
- **Documentation:** [BeagleMind Forum Thread](#)
- **GSoC:** [Project Description on GSoC](#)

1.2 Status

This project is currently just a proposal.

1.3 Proposal

- Created accounts across [OpenBeagle](#) and [Beagle Forum](#)
- The PR Request for Cross Compilation: [#197](#)
- Created a project proposal using the [proposed template](#).

1.4 About

- Forum: [FAYEZ_ZOUARI](#)
- OpenBeagle: [fayezzouari](#)
- Discord ID: [.kageyamo](#)
- GitHub: [fayezzouari](#)
- School: INSAT (National Institute of Applied Science and Technology)
- Country: Tunisia
- Typical work hours: 9:00 AM - 6:00 PM (UTC+1)
- Previous GSoC participation: No

Chapter 2

Project

Project name: BeagleMind - Documentation Assistant with Fine-tuned LLM and RAG

2.1 Description

BeagleMind combines fine-tuned LLMs with RAG to create an accurate documentation assistant that:

1. Uses PEFT/LoRA fine-tuning on BeagleBoard documentation
2. Implements RAG for fact-based responses and to reduce LLM hallucination
3. Accessed using a HF inference endpoint
4. Deploys via: - CLI tool for local usage - Web interface with websockets
5. Includes agentic evaluation framework

2.1.1 Technical Implementation

2.2 LLM Fine-tuning Architecture

The system will employ the selected LLM as its base model, utilizing Parameter-Efficient Fine-Tuning (PEFT) with LoRA adapters to specialize the model for BeagleBoard documentation. The training pipeline processes OpenBeagle resources through:

- Semantic segmentation of technical documentation
- Generation of instruction-response pairs
- Dynamic masking of code samples for focused learning

Evaluation will combine:

- Perplexity measurements on held-out documentation
- Task-specific accuracy on BeagleBoard API questions
- Human review of generated troubleshooting steps

2.3 RAG Integration Pipeline

The retrieval-augmented generation system implements a three-stage accuracy enforcement:

1. Document Processing:

- Hierarchical chunking preserving code-sample context
- Metadata enrichment with section headers
- Cross-document relationship mapping

2. Vector Retrieval:

- Hybrid dense-sparse retrieval using BAAI embeddings
- Query-adaptive reranking
- Confidence-based fallback mechanisms

3. Response Generation:

- Contextual grounding with retrieved passages
- Automatic citation injection
- Confidence thresholding for uncertain responses

2.4 Hosting Infrastructure

The production deployment features:

Table 1: Hosting Specifications

Component	Implementation
Inference Endpoint	Hugging Face TGI with 4-bit quantization
Load Balancing	Round-robin with health checks
Monitoring	Prometheus metrics for: - Token generation latency - Retrieval hit rate - Hallucination alerts

2.5 Deployment Targets

Multi-platform accessibility through:

1. Web Interface:

- React.js frontend with response streaming
- Interactive citation visualization
- Session-based query history

2. CLI Tool:

- Access to the hosted LLM through an Api Key
- Configurable verbosity levels
- Automated test script integration

2.6 Evaluation Framework

The agentic evaluation system employs three specialized test agents:

1. Fact-Verification Agent:

- Cross-references answers with source docs
- Flags unsupported technical claims
- Maintains accuracy heatmaps

2. Completeness Auditor:

- Scores answer depth on:
 - API reference coverage
 - Troubleshooting steps
 - Example code relevance

3. Stress-Test Bot:

- Generates adversarial queries
- Measures failure modes
- Identifies documentation gaps

2.7 Software

- **Programming Languages:** Python
- **ML Tools:** PEFT, LoRA, Quantization
- **Frameworks:** FastAPI, Hugging Face Transformers
- **Database:** ChromaDB/Weaviate/Qdrant
- **Frontend:** React
- **Deployment:** Docker, Nginx, PYPI, Hugging Face Spaces
- **Version Control:** Git, GitHub/GitLab

2.8 Hardware

- **Development Boards:** - BeagleBone AI-64 - BeagleY-AI
- **Cloud Services:** - Hugging Face Spaces / Inference Endpoints - Vercel

Chapter 3

Architecture and Diagrams

These diagrams represent the workflow of the methods mentioned earlier.

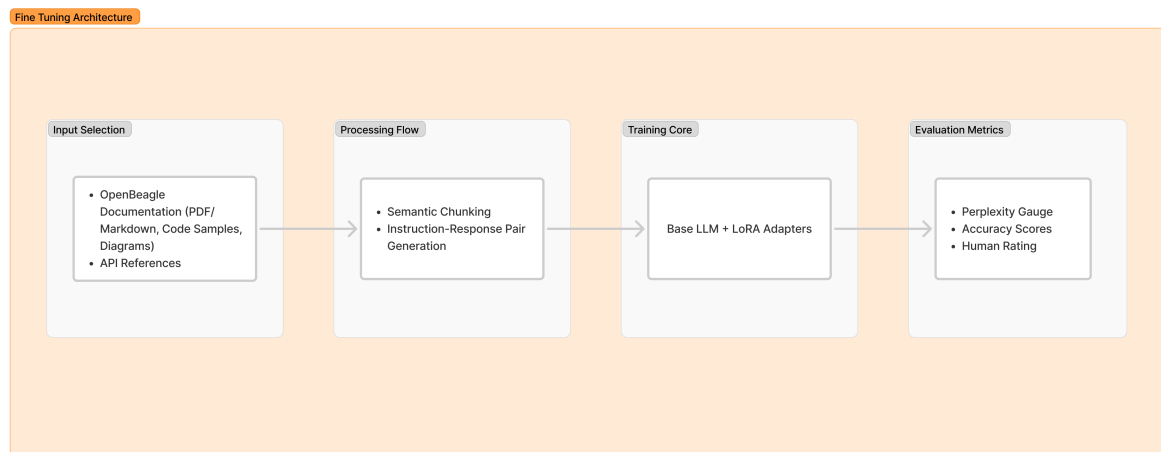


Fig. 1: Fine-Tuning Architecture

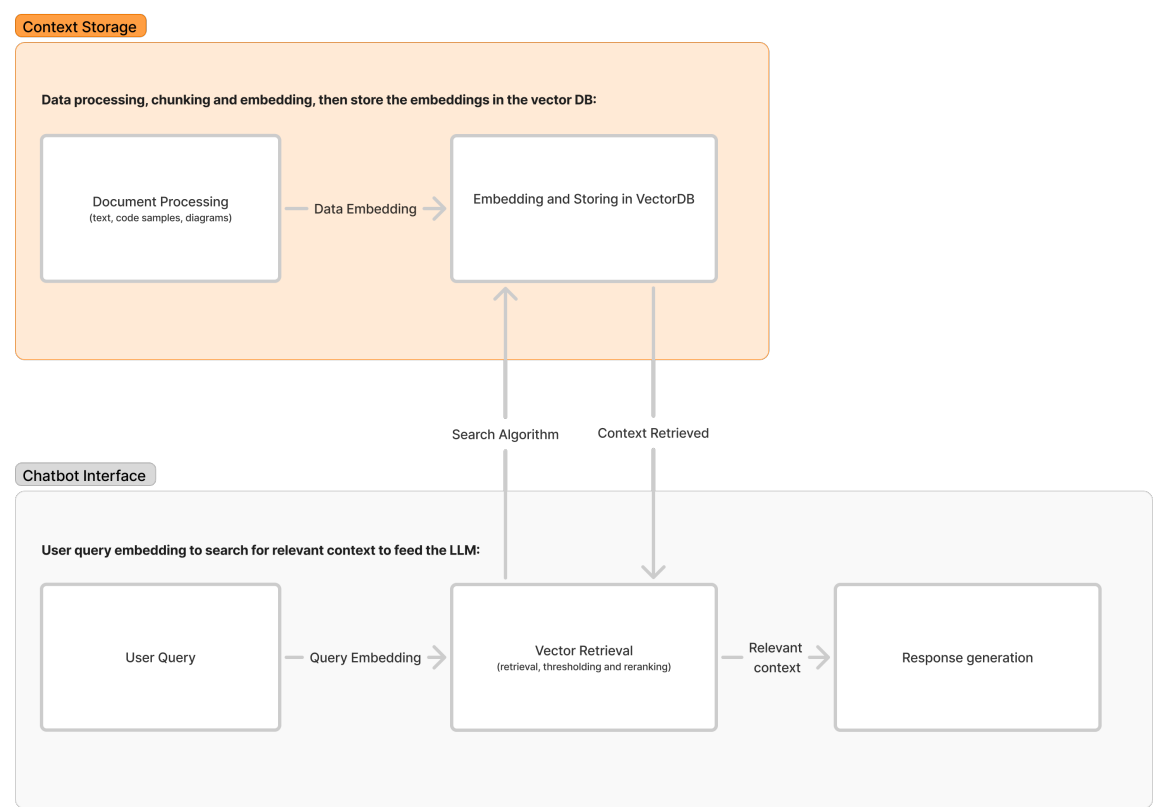


Fig. 2: RAG Integration Pipeline

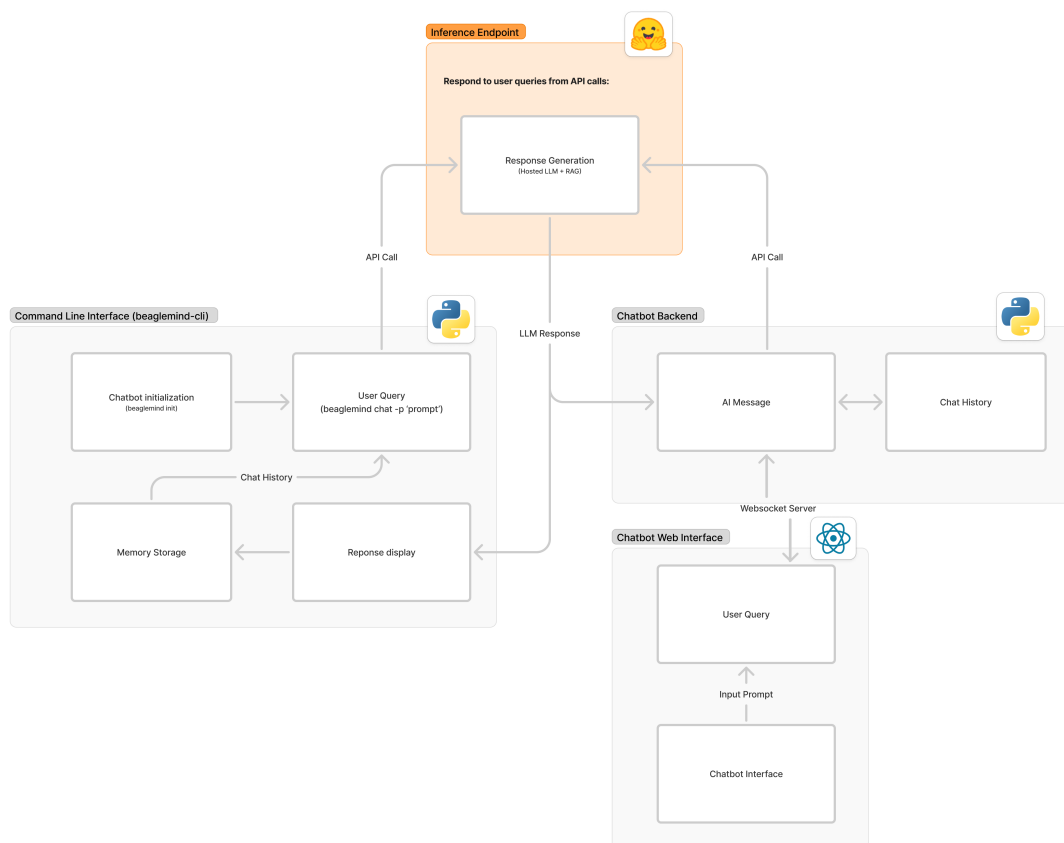


Fig. 3: Deployment Structure

Chapter 4

Timeline

Deadline	Milestone	Deliverables
May 27	Coding Begins	Finalize architecture diagrams
June 3	M1: Foundation	CLI prototype, Fine-tuning strategy doc
June 17	M2: Data Preparation	Curated dataset, Vector DB ready
July 1	M3: Model Training	Fine-tuned model on HF, Initial benchmarks
July 8	Midterm Evaluation	Working CLI with local inference
July 22	M4: Agentic Evaluation	Test agents implemented, Accuracy reports
Aug 5	M5: Web Interface	Websocket server, React frontend
Aug 19	Final Submission	Full documentation, Demo video

4.1 Detailed Timeline

4.1.1 Community Bonding (May 9 - May 26)

- Develop workflow diagrams:
 - Data collection pipeline
 - Fine-tuning process
 - RAG integration flow
- Finalize model selection criteria
- Establish evaluation metrics with mentor

4.1.2 Milestone 1: Foundation (June 3)

1. CLI Prototype:

- Basic question-answering interface
- Chatbot using only RAG just to present the PoC
- Provide helpful parameters like -h for help, -p for prompt and -l to refer to a log file
- Simple evaluation script

2. Video demonstration:

- Provide video demonstration
- Present a proof of concept
- Highlight that the actual solution will feature a hosted fine-tuned LLM and RAG to reduce hallucination

3. Fine-tuning Prep:

- Document preprocessing scripts
- Training environment setup

4.1.3 Milestone 2: Data Preparation (June 17)**1. Document Processing:**

- Data formatting
- Generate synthetic Q&A pairs
- Convert all docs to clean Markdown
- Extract code samples, diagrams, circuit schemas and any resource that could help in the troubleshooting

2. Vector Database:

- Implement chunking strategy
- Test retrieval accuracy
- Optimize embedding selection

4.1.4 Milestone 3: Model Training (July 1)**1. Fine-tuning:**

- Training runs with different parameters
- Loss/accuracy tracking
- Quantization tests

2. Deployment:

- HF Inference Endpoint setup
- Performance benchmarks
- Hallucination tests

4.1.5 Midterm Evaluation (July 8)

- Functional CLI with:
 - Model inference
 - Basic RAG integration
 - Accuracy metrics
- Video demonstration
- Mentor review session

4.1.6 Milestone 4: Agentic Evaluation (July 22)**1. Evaluation Agents:**

- Fact-checking agent
- Completeness evaluator
- Hallucination detector

2. **Automated Testing:**

- 100-question test suite
- Continuous integration setup
- Performance dashboard

4.1.7 Milestone 5: Web Interface (Aug 5)

1. **Backend:**

- FastAPI websocket server
- Dockerize the server
- Async model loading
- Rate limiting

2. **Frontend:**

- React-based chat UI
- Response visualization
- Mobile responsiveness

4.1.8 Final Submission (Aug 19)

- Comprehensive documentation:
 - Installation guides
 - API references
 - Training methodology
- 5-minute demo video
- Performance report

4.2 Benefit

BeagleMind will provide:

- 24/7 documentation assistance
- Reduced maintainer workload
- Visualized technical answers
- Accelerated debugging
- Offline documentation access
- Improved onboarding experience

Chapter 5

Experience and Approach

5.1 Personal Background

As an Embedded Systems Engineering student with a passion for AI and robotics, I find the BeagleMind project perfectly aligns with my academic specialization and technical interests. My coursework in embedded systems, combined with self-study in Large Language Models (LLMs) and Retrieval-Augmented Generation (RAG), has prepared me to bridge the gap between hardware documentation and AI-powered assistance.

5.2 Experience

As an Embedded Systems Engineering student with AI specialization, I bring:

1. **LENS Platform:**

- RAG Chatbot with Citations: Developed a retrieval-augmented chatbot that provides answers with detailed references, URL, page number, and File Name.

2. **Chatautomation Platform:**

- Built multimodal data loaders (PDFs, images, audio)
- Implemented voice interaction system (STT + LLM + TTS)
- Developed WhatsApp/Instagram chatbot integrations

3. **Orange Digital Center Internship:**

- Created MEPS monitoring system
- Developed biogas forecast mode
- Implemented agentic workflows for production reports

4. **x2x Modality Project:**

- Hexastack Hackathon 1st place (Open source contribution)
- Speech to Text for effortless communication
- Text to Speech for improved accessibility
- Image and Document Processing into text for smoother integration

5.3 Contingency

If blockers occur:

1. Research documentation and source code
2. Seek community support (Discord/Forum)
3. Implement alternative approaches
4. Escalate to the mentor if unresolved

5.4 Misc

- Will comply with all GSoC requirements
- Merge request will be submitted to BeagleBoard GitHub
- Current demo available at bb-gsoc.fayez-zouari.tn | [CLI GitHub Repo](#)

5.5 References

1. [Hugging Face Transformers](#)
2. [ChromaDB Documentation](#)
3. [BeagleBoard Documentation](#)
4. [PEFT Fine-tuning](#)