



GSoC Proposal for BeagleBoard.org

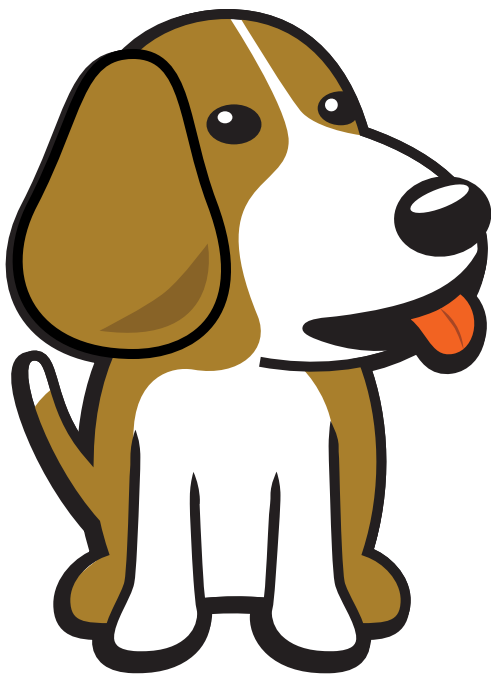


Table of contents

1	Introduction	1
1.1	Summary links	1
1.2	Status	1
1.3	Proposal	1
1.4	About	1
2	About the Project	3
2.1	Description	3
2.2	Objectives	3
2.3	Goals	3
3	Methods	5
3.1	Collecting training Dataset and Preprocessing	5
3.2	Retrieval-Augmented Generation (RAG) Implementation	5
3.3	LLM Fine-Tuning Strategy	5
3.4	Model Deployment and Web Interface Development	6
3.5	Full-Stack Implementation	6
3.6	Performance Evaluation Metrics	6
3.7	Hardware Testing and Optimization	6
3.8	Project Workflow	7
3.9	Software	7
3.10	Hardware	7
3.11	Timeline summary	8
3.12	Timeline detailed	8
3.12.1	Community Bonding Period (May 8th - June 1st)	8
3.12.2	Milestone #1, Releasing Introductory Video and Data Preparation & Preprocessing (June 15)	8
3.12.3	Milestone #2, RAG Pipeline Development (June 29)	8
3.12.4	Milestone #3, Fine-Tuning LLM (July 7)	9
3.12.5	Milestone #4 - Building the Chatbot (July 13)	9
3.12.6	Submit midterm evaluations (July 18th)	9
3.12.7	Milestone #5 - Model Optimization & Deployment (July 31)	9
3.12.8	Milestone #6 - Web Interface Development (August 10)	9
3.12.9	Milestone #7 - Full-Stack Features & Final Testing (July 29)	9
3.12.10	Submit final project video, documentation and final work to GSoC site and complete final mentor evaluation (August 24)	10
3.12.11	Final Submission (August 25)	10
3.12.12	Initial results (September 1)	10
4	Experience and approach	11
4.1	Contingency	11
4.2	Benefit	12
4.3	Misc	12

Chapter 1

Introduction

The BeagleBone® AI-64 from the BeagleBoard.org Foundation is a complete system for developing artificial intelligence (AI) and machine-learning solutions with the convenience and expandability of the BeagleBone platform and onboard peripherals to start learning and building applications. To enhance user experience, this project will implement an AI-powered assistant on the BeagleBone AI-64 to help users troubleshoot errors and streamline the onboarding process for new contributors. Leveraging the board's powerful processing units, the assistant will provide real-time, efficient support, offering solutions to technical issues and guiding newcomers through BeagleBoard's ecosystem.

1.1 Summary links

- **Contributor:** [Ashish Upadhyay](#)
- **Mentors:** [Aryan Nanda](#)
- **Repository:** [Main Code Repository on Gitlab](#), [Mirror of Code Repository on Github](#)
- **Documentation:** [Ashish Upadhyay / docs.beagleboard.io](#) · [GitLab](#)


1.2 Status





This project is currently just a proposal.

1.3 Proposal

- Created accounts across [OpenBeagle](#), [Discord](#) and [Beagle Forum](#)
- The PR Request for Cross Compilation: [#208](#)
- Created a project proposal using the [proposal template](#)

1.4 About

- **Resume** - Find my resume [here](#)
- **Forum:** [u/ashish_upadhyay](#)
- **OpenBeagle:**  [ashishu23](#)
- **IRC:**  [ashishu23](#) (Ashish Upadhyay)

- **Github:**  ashishu23 (Ashish Upadhyay)
- **School:** Indian Institute of Technology, Kanpur
- **Country:**  India
- **Primary language:**  English, Hindi
- **Typical work hours:** 9AM-5PM Indian Standard Time
- **Previous GSoC participation:**  This would be my first time participating in GSoC

Chapter 2

About the Project

Project name: A Conversational AI Assistant for BeagleBoard using RAG and Fine-tuning

2.1 Description

The BeagleBoard ecosystem currently lacks an AI-driven assistant capable of facilitating user support, particularly in troubleshooting and onboarding new contributors. This project aims to bridge this gap by developing a domain-specific AI assistant that employs Retrieval-Augmented Generation (RAG) and fine-tuning an open-source Large Language Model (LLM) such as Llama 3, Mixtral, or Gemma. The integration of these 2 methodologies will enable the chatbot to retrieve and generate precise, context-aware responses, thereby enhancing user experience and engagement within the BeagleBoard community.

2.2 Objectives

The objectives of the project are -

1. Phase 1:-

- Implement an **RAG-based chatbot** for BeagleBoard, leveraging domain-specific data.
- Fine-tune a **state-of-the-art LLM** on BeagleBoard-related technical documentation and forum discussions.
- Deploy the optimized model using **Hugging Face Inference Endpoints** to ensure scalable access.
- Construct a **retrieval system** utilizing a vector database (e.g., Qdrant, FAISS, ChromaDB) for real-time knowledge extraction.

2. Phase 2:-

- Develop an interactive **Gradio-based web interface** for seamless user interaction.
- Optimize model inference using quantization techniques for deployment on native hardware accelerators present in **BeagleBone AI-64** for enhancing real-time performance.
- Extend the solution into a **full stack application** with authentication and historical query storage.

2.3 Goals

The goals of the project are -

1. Phase 1:-

- **Collecting** and **preprocessing** data from various sources.

- **Fine-tuned LLM** incorporating BeagleBoard-specific knowledge.
- **RAG pipeline** integrating a retriever mechanism to enhance contextual accuracy.
- A **CLI based chatbot** interface.
- **Comprehensive Evaluation Report, including:**
 - **Quantitative Metrics:** BLEU, ROUGE, METEOR, latency benchmarks.
 - **Qualitative Assessments:** Expert validation from BeagleBoard developers.
 - **Cost Analysis:** Deployment and maintenance expenditure estimates.

2. Phase 2:-

- **Interactive Web Interface:** A Gradio-based user-friendly application for chatbot interaction using text and voice.
- **Implementation Guide:** Detailed documentation on dataset preprocessing, fine-tuning methodologies, and deployment strategies.
- **Hardware Deployment Testing:** Performance benchmarking on BeagleBone AI-64 and BeagleY-AI with quantization.
- **Advanced Full-Stack Features:**
 - Secure authentication and access control.
 - Database integration for persistent chat history storage.

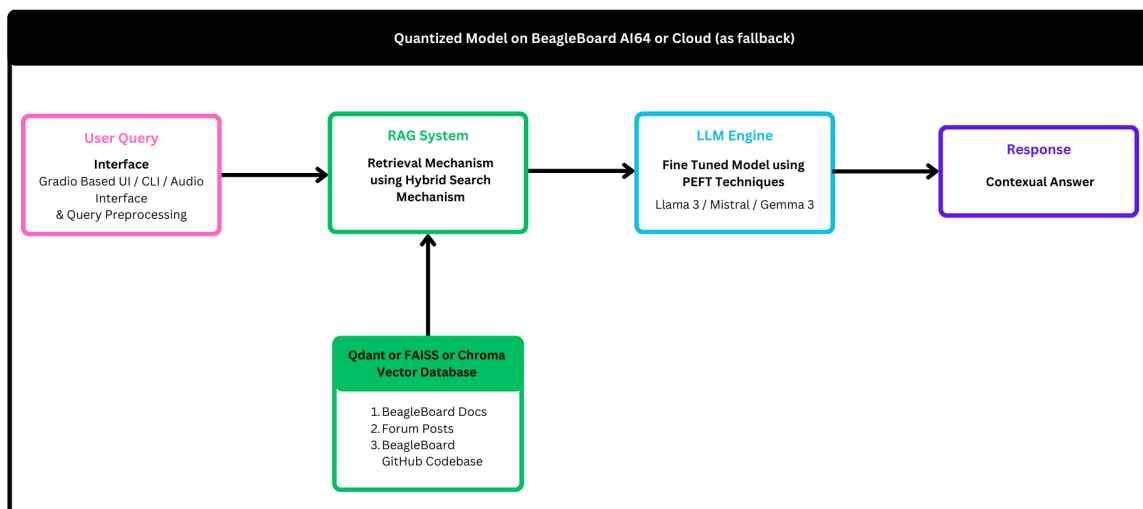


Figure 1: Final Project Pipeline

Chapter 3

Methods

In this section, I will individually specify the methods that I plan on using in greater details.

3.1 Collecting training Dataset and Preprocessing

1. **Data Collection** - Extract relevant content from BeagleBoard documentation, GitHub, forums.
2. **Scraping** - Employ BeautifulSoup and Scrapy for structured data extraction.
3. **Preprocessing** - Remove extraneous elements, such as HTML tags and code blocks.
4. **Structuring** - Organize data into question-answer pairs and troubleshooting dialogues.
5. **Chunking** - Segment documents into semantically coherent, retrievable units.
6. **Tokenisation** - Utilize the LLM's tokenizer for optimized dataset preparation.
7. **Dataset Format** - Convert structured data into the Hugging Face datasets.Dataset format.

3.2 Retrieval-Augmented Generation (RAG) Implementation

1. **Vector Database** - Store BeagleBoard knowledge embeddings using Qdrant/FAISS.
2. **Search Strategy** - Implement a hybrid search mechanism like combining BM25 and dense vector similarity.
3. **Response Pipeline** - Retrieve relevant documents → Augment query context → Generate response.

3.3 LLM Fine-Tuning Strategy

1. **Model Selection** - Choose from LLaMA 3, Mixtral, Gemma 3 based on performance benchmarks.
2. **Domain Adaptation** - Train on refined BeagleBoard datasets for optimal domain knowledge.
3. **Instruction Tuning** - Format Q&A data for instruction-following conversational flow.
4. **PEFT Techniques** - Implement LoRA/QLoRA for compute-efficient fine-tuning.
5. **Evaluation** - Use BLEU, ROUGE, METEOR, Exact Match, Latency as assessment metrics.
6. **Frameworks** - Leverage Transformers, PEFT, Accelerate, Datasets, Evaluate.

3.4 Model Deployment and Web Interface Development

1. **Model Hosting** - Deploy on Hugging Face Inference Endpoints for optimized access.
2. **Web Interface** - Develop an intuitive Gradio-based UI for user interactions.
3. **Voice-Chatbot Integration** - Audio integration in UI using either Gradio's inputs.Audio and outputs.Audio or using speech-to-text (STT) functionality (using Whisper (OpenAI) or SpeechRecognition library) and text-to-speech (TTS) functionality (using pyttsx3 or gTTS). So, when a user speaks: Audio is transcribed to text → Sent to the chatbot → Response generated → Converted to audio (TTS) → Played back and when a user types: Text is sent directly → Response generated → Also converted to audio (optional).
3. **Cost Estimation** - Compute inference costs based on model complexity and query volume.
4. **Performance Evaluation** - Assess model latency, throughput, and real-time response accuracy.
5. **Hardware Testing** - Benchmark quantized models (4-bit or int8) on BeagleBone AI-64 and BeagleY-AI.

3.5 Full-Stack Implementation

1. **Authentication** - Implement secure user login and query tracking.
2. **Database** - Store chatbot interaction history using SQLite/PostgreSQL.
3. **Frontend** - Integrate Gradio UI with optional admin dashboard.
4. **Analytics** - Log user queries, response accuracy, and performance trends.

3.6 Performance Evaluation Metrics

1. **Quantitative** - BLEU, ROUGE, METEOR, Latency (ms).
2. **Database** - Expert evaluation from BeagleBoard engineers.
3. **Frontend** - Estimation of Hugging Face endpoint expenses.
4. **Analytics** - Real-world testing with community feedback.

3.7 Hardware Testing and Optimization

1. **Model quantization** to 4-bit or lower using QLoRA, GPTQ, AWQ.
2. **Deployment and performance evaluation** on BeagleBone AI-64.
3. **Comparison with cloud inference** to analyze local vs. cloud-based trade-offs.

3.8 Project Workflow

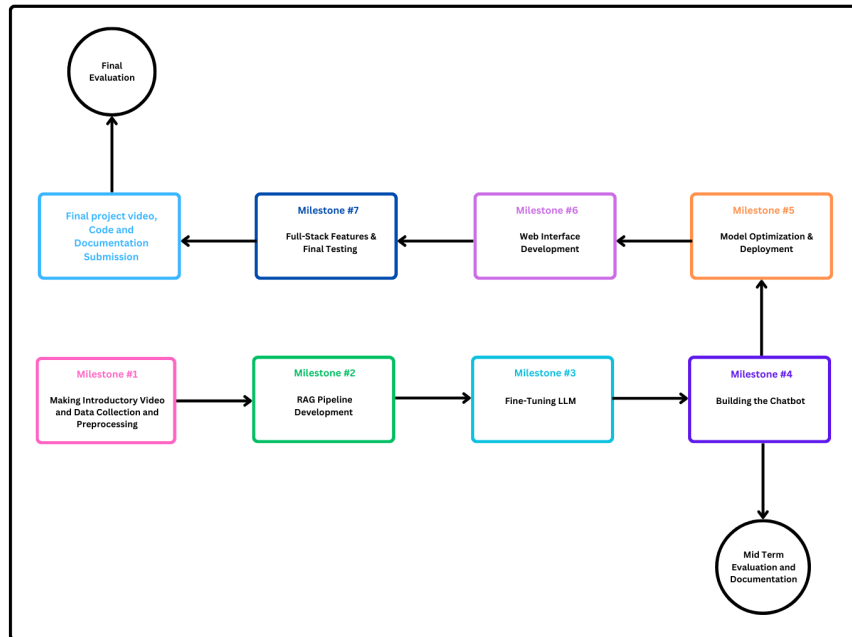


Figure 2: Project Workflow

3.9 Software

- Python
- NLTK
- spaCy
- BeautifulSoup
- Fine tuning LLMs(techniques like LoRA and QLoRA)
- Gradio
- Hugging Face Inference Endpoints
- Git

3.10 Hardware

- Ability to test applications on [BeagleBone AI-64](#) and optimize for performance using quantization techniques.

3.11 Timeline summary

Date	Activity
February 27 - March 23	Connect with possible mentors, complete prerequisites, verify value to community and request review on proposal draft
March 24 - April 7	Proposal review and Submit application
April 8 - May 7	Research more on the project pipeline and its implementation on BeagleBone AI-64 and focus on college exams
May 8 - June 1	ACRBonding
June 2 - June 4	Start coding and introductory video
June 5 - June 15	Milestone #1, Releasing Introductory Video and Data Preparation & Preprocessing (June 15)
June 16 - June 29	Milestone #2, RAG Pipeline Development (June 29)
June 30 - July 7	Milestone #3, Fine-Tuning LLM (July 7)
July 8 - July 13	Milestone #4 - Building the Chatbot (July 13)
July 14 - July 18	ACRSubmit-midterm-evaluations
August 11 - August 23	Milestone #5 - Model Optimization & Deployment (July 31)
July 22 - July 28	Milestone #6 - Web Interface Development (August 10)
July 29 - August 4	Milestone #7 - Full-Stack Features & Final Testing (July 29)
August 24	Submit final project video, documentation and final work to GSoC site and complete final mentor evaluation (August 24)
August 25 - September 1	Final Submission (August 25)

3.12 Timeline detailed

3.12.1 Community Bonding Period (May 8th - June 1st)

- Discuss implementation ideas with mentors.
- Discuss the scope of the project.

3.12.2 Milestone #1, Releasing Introductory Video and Data Preparation & Preprocessing (June 15)

- Making an Introductory Video
- **Data Preparation & Preprocessing:**
 - Scrape relevant BeagleBoard data (docs, forums, GitHub) using Scrapy & BeautifulSoup
 - Clean and preprocess text (remove HTML tags, code blocks, noise)
 - Structure data into Q&A pairs, troubleshooting dialogues, and chunk semantically
 - Tokenize and format dataset for Hugging Face's datasets library

3.12.3 Milestone #2, RAG Pipeline Development (June 29)

- Implement a vector database (Qdrant/FAISS/ChromaDB) to store knowledge embeddings
- Develop hybrid search mechanism that combines BM25 (sparse retrieval) with dense vector similarity (e.g., using MiniLM, BGE, or OpenAI embeddings) for document retrieval. I plan to experiment with different fusion strategies—such as simple score combination, Reciprocal Rank Fusion (RRF), or reranking methods—and adopt the one that provides the best performance based on empirical evaluation.
- Integrate retrieval pipeline with LLM to augment responses

3.12.4 Milestone #3, Fine-Tuning LLM (July 7)

- Select base model (Llama 3 / Mixtral / Gemma 3) based on performance benchmarks
- Train model on domain-specific BeagleBoard data
- Use Parameter-Efficient Fine-Tuning (PEFT) (LoRA/QLoRA) for compute efficiency

3.12.5 Milestone #4 - Building the Chatbot (July 13)

- Functional RAG pipeline with basic chatbot responses
- Fine-tuned model with preliminary accuracy benchmarks
- Demo web UI with initial chatbot interface

3.12.6 Submit midterm evaluations (July 18th)

- Document the progress made during the first phase of the project.

Important: July 18 - 18:00 UTC: Midterm evaluation deadline (standard coding period)

3.12.7 Milestone #5 - Model Optimization & Deployment (July 31)

- **Model Optimization:**
 - Optimize model inference using quantization (4-bit/int8) using QLoRA, GPTQ, AWQ for efficiency
- **Model Deployment:**
 - Deploy model using Hugging Face Inference Endpoints
 - Benchmark performance (latency, accuracy, cost analysis)

3.12.8 Milestone #6 - Web Interface Development (August 10)

- Develop an interactive UI using Gradio for chatbot interaction
- Audio integration in the UI for interaction along text.
- Integrate real-time inference with backend model
- Implement authentication & user session tracking

3.12.9 Milestone #7 - Full-Stack Features & Final Testing (July 29)

- **Full-Stack Features:**
 - Integrate SQLite/PostgreSQL database for storing chat history
 - Log user queries and response accuracy for analytics
- **Final Testing on Hardware:**
 - Deployment and performance evaluation on BeagleBone AI-64
 - Comparison with cloud inference to analyze local vs. cloud-based trade-offs

3.12.10 Submit final project video, documentation and final work to GSoC site and complete final mentor evaluation (August 24)

- Submit final project video, documentation and final work to GSoC site and complete final mentor evaluation.

3.12.11 Final Submission (August 25)

Important: August 25 - September 1 - 18:00 UTC: Final week: GSoC contributors submit their final work product and their final mentor evaluation (standard coding period)

September 1 - September 8 - 18:00 UTC: Mentors submit final GSoC contributor evaluations (standard coding period)

3.12.12 Initial results (September 1)

Important: September 1 - November 9: GSoC contributors with extended timelines continue coding

November 10 - 18:00 UTC: Final date for all GSoC contributors to submit their final work product and final evaluation

November 17 - 18:00 UTC: Final date for mentors to submit evaluations for GSoC contributor projects with extended deadline

Chapter 4

Experience and approach

This project requires prior experience with NLP, RAG, LLM and embedded systems.

- I have worked in a problem statement by Pathway in the Inter IIT Tech Meet 13.0 held at IIT Bombay where we were required to build a dynamic RAG using Pathway platform. Here is the presentation and demonstration submitted by IIT Kanpur - [Presentation](#) and [Video](#)
- I am working on a project to build a [multimodal LLM for Indian Sign Language](#) where I collected the largest ISL dataset from YouTube, extracted poses from it, preprocessed using NLP techniques, trained a multimodal LLM from scratch and made a benchmark for ISL translation. The [project](#) is under the guidance of Ashutosh Modi from CSE department, IIT Kanpur.
- I have also built a [automated reviewer for GitHub PR](#) for an organisation named GodSpeed. Here, I used OpenAI LLM to review any instance of a WebHook event which is triggered whenever a PR is made on the selected GitHub repository. [Implementation without GodSpeed platform](#).
- I have also worked in a project to make a [smart interactive voice and touch controlled mirror](#) where I used Arduino Nano BLE 33 to initiate the voice input like “Alexa” or “Ok Google” and RaspberryPi for LLM deployment. The LLM was fine tuned on IITK campus related information like courses details etc. The touch mirror was made by making a coordinate system using IR sensors. An implementation video of the project is on the page 15 of the [presentation](#)
- I have also worked in a problem statement by Kalyani Bharat Forge in the Inter IIT Tech Meet 13.0 held at IIT Bombay where we used ML and RL techniques for swarm robotics and achieved 6th position among all 23 IITs. Here is the documentation and presentation submitted by IIT Kanpur - [Documentation](#) and [Presentation](#).
- As the coordinator of Electronics Club, IIT Kanpur and the team head of Team AUV, IIT Kanpur, I have worked on embedded systems and various development boards including BeagleBoard AI-64, STM32 and Arty Z7.

4.1 Contingency

- **If I get stuck on my project and my mentor isn't around, I will use the following resources:-**
 - [Data Camp tutorial on fine-tuning Llama 3](#)
 - [How to scrape web using BeautifulSoup](#)
 - [Converting Git repository into a simple text digest of its codebase for feeding to LLM](#)
 - [BeagleBoard docs site codebase](#)
 - [Different Fine tuning techniques](#)
 - [Creating Demos with Spaces and Gradio](#)
 - [LLM inference optimization](#)
 - [Hugging face spaces pricing](#)

- [BeagleBone AI-64 docs](#)

- Moreover, the BeagleBoard community is extremely helpful and active in resolving doubts, which makes it a great going for the project resources and clarification.
- I intend to remain involved and provide ongoing support for this project beyond the duration of the GSoC timeline.

4.2 Benefit

Some of the benefits of the project will be -

- Faster Troubleshooting - AI-powered assistant helps users quickly resolve errors.
- Streamlined Onboarding - Guides new contributors, reducing learning time.
- Efficient Deployment - Uses RAG + LoRA/QLoRA for cost-effective model tuning.
- Optimized for BeagleBoard Hardware - Supports quantization for better performance.
- Enhanced Documentation - Acts as an interactive knowledge base.
- Open-Source Contribution - Advances LLM fine-tuning for hardware-focused AI.
- Web & Local Access - Gradio interface for ease of use; local deployment for flexibility.
- Community-Driven - Feedback loop ensures continuous improvement.

4.3 Misc

- My LinkedIn Account: [Ashish Upadhayay](#)