Google Summer of Code 2025 Proposal

Personal Information
Name: Shubhanshu Kumar
Email: subhanshukumar290@gmail.com
GitHub: github.com/kumarshubhh
LinkedIn: linkedin.com/in/shubhanshu-kumar-6a961525a
University: Dr. A.P.J. Abdul Kalam Technical University
Degree Program: B.Tech in Information Technology
Year of Study: Final Year (8th Semester)

Title: FAQ Detection Assistant

Synopsis

Rocket.Chat is a powerful open-source communication platform used by large developer communities and enterprises. A recurring challenge in such environments is responding to Frequently Asked Questions (FAQs) in a scalable and efficient manner.

This project proposes an intelligent **FAQ Detection Assistant** that automatically detects if an incoming message matches an existing FAQ and takes a predefined action (e.g., reply, notify, or DM). The solution will integrate LLM-powered prompt chaining and vector-based semantic matching to ensure robust detection and relevance.

Key Features:

- Monitor configured channels or GitHub Discussions for queries.
- Detect message-to-FAQ similarity using LLMs and embeddings.
- Configurable response logic: automated reply, notification, or approval.
- Customizable actions via a JSON config schema.
- Feedback loop and optional UI for FAQ management.

Benefits to the Community

- **Scalability:** Reduces dependency on human moderators.
- **Efficiency:** Speeds up query resolution.
- **Consistency:** Ensures accurate and uniform responses.
- **Integrable:** Adaptable to GitHub Discussions, support forums, etc.
- **Configurable:** Users can set thresholds and define custom behaviors.

Deliverables

Time Period	Deliverable
Community Bonding	Understand Rocket.Chat's APIs and architecture. Set up local dev environment. Review existing FAQ structures. Start contributions.
Phase 1	- Build webhook listener for capturing messages. - Set up LLM-based prompt chaining. - Create FAQ schema and integration. - Develop initial similarity detection.
Evaluation 1	Prototype that detects similar questions and logs possible matches.
Phase 2	- Add configurable actions (DM, reply, notify). - Create a feedback loop for corrections. - Build basic FAQ management UI (if needed). - Testing and refinements.
Final Evaluation	Fully functional FAQ Detection Assistant with documentation, test cases, and deployment instructions.

Technical Approach

1. Message Listening via Webhook

Use Rocket.Chat's webhook system to detect new messages:

```
export async function webhookHandler(req, res) {
  const { message } = req.body;
  const userQuery = message.text;
  const response = await detectFAQMatch(userQuery);
  // Take configured action
  res.sendStatus(200);
}
```

2. Prompt Chaining with LLMs

LLMs will assess similarity between user queries and FAQs:

```
const systemPrompt = `You are an FAQ assistant. Given a user query and a list of FAQs, detect if there is a match.`;
const prompt = `User Query: ${query}\nFAQs:\n1. ${faqList.join('\n')}\nOutput: Most similar FAQ or "No Match"`;
const response = await callLLM(systemPrompt + prompt);
```

3. Vector-Based Similarity

Cosine similarity on sentence embeddings will improve detection accuracy:

```
function cosineSimilarity(a, b) {
  const dot = a.reduce((sum, val, i) => sum + val * b[i], 0);
  const magA = Math.sqrt(a.reduce((sum, val) => sum + val ** 2, 0));
  const magB = Math.sqrt(b.reduce((sum, val) => sum + val ** 2, 0));
  return dot / (magA * magB);
}
```

4. Configurable Actions

Actions based on a JSON config file:

```
{
 "actions": {
   "onMatch": "notify",
   "notifyUsers": ["admin"]
 }
}
```

Timeline

Date Range	Milestone
May 20 – June 16	Community bonding, understanding the codebase, initial discussions, first contribution
June 17 – July 28	Implement core detection logic, LLM prompt chaining, webhook integration
July 29 – August 26	Implement configurable actions, feedback loop, UI enhancements, testing
August 27 – Sept 9	Final testing, documentation, and blog submission

About Me

I'm a backend-focused full-stack developer with solid experience in building scalable applications. I've worked on real-time chat apps, AI integrations, and webhook-based systems. I'm deeply aligned with Rocket.Chat's mission of open, secure communication and excited to contribute to its ecosystem through this project.

Skills:

- TypeScript, JavaScript, Node.js
- Prompt Engineering, OpenAI APIs
- MongoDB, Express, React
- REST APIs, WebSockets
- Git, CI/CD, Debugging

Projects:

- **Tradexa** – A Zerodha-inspired stock trading platform (MERN).
- **NEXAMEET** – A Zoom-like real-time video app using WebSocket.
- **Wandernest** – A travel community platform to connect explorers.
- **Gemini Clone** – AI chatbot interface using Gemini API.

Contributions to Rocket.Chat

- Setting up Rocket.Chat dev environment and exploring `apps-engine`.
- Studying PRs and issues related to bot automation and support tools.
- Plan to contribute through beginner-friendly issues during bonding phase.

Why Rocket.Chat?

Rocket.Chat's focus on transparency, real-time messaging, and developer-first integrations aligns perfectly with my interests in building impactful tools for communities. Its vibrant contributor ecosystem is ideal for learning and giving back.

Final Thoughts

This project resonates with my technical strengths and open-source enthusiasm. I am confident that I can deliver a robust, scalable, and intelligent FAQ Detection Assistant that improves Rocket.Chat’s support experience. I look forward to growing as a developer and giving back to the community through GSoC 2025.

Thank you for considering my application!