AI-Powered FAQ Chatbot — Project Report

Objective

Build a chatbot that answers FAQs using lightweight NLP or a small ML model.

Tools

React (frontend), TensorFlow.js or OpenAI API (free tier) for AI logic, Node.js (backend).

Deliverable

Interactive chatbot with messaging UI that handles FAQs, accepts feedback, and logs unanswered queries for iterative improvement.

Executive Summary

This project delivers an AI-powered FAQ chatbot that provides accurate, concise responses to frequently asked questions. The system uses a small FAQ model or keyword-matching with fallback to an API. The frontend is a React-based chat UI, and the backend is Node.js. Feedback buttons let users mark answers as helpful or not; unanswered queries are stored for later review and model improvement.

System Architecture

1. Client (React): Messaging UI, input, feedback buttons.
2. Server (Node.js): APIs for messages, feedback, logs.
3. AI Layer: TensorFlow.js FAQ model or OpenAI API.
4. Data Store: Save FAQs, logs, unanswered queries.
5. Monitoring: Analytics of failed queries, accuracy.

FAQ Model & AI Logic

The chatbot can use keyword matching or a small TensorFlow.js classifier to map queries to FAQ answers. Fallback mechanisms (low-confidence threshold) ensure robustness. Training involves FAQ question variants and paraphrases.

Frontend (React) Design

Chat window with user/bot bubbles, input box, quick replies, and feedback buttons. Bot responses include ■ Helpful / ■ Not Helpful options. Mobile responsive and accessible.

Backend (Node.js) Responsibilities

APIs for message processing, AI inference, feedback, and logs. AI logic integrates with TF.js or OpenAI. Data persistence in a database for FAQs, logs, and unanswered queries.

Evaluation & Improvement

Metrics: answer accuracy, unresolved queries, response time, satisfaction score. Human-in-loop: review logs, add FAQs, retrain model. A/B test confidence thresholds.

Security & Deployment

Privacy: store only essential metadata. Secure API keys, HTTPS, and rate limiting. Deployment: containerized with Docker, deployable on Heroku, Vercel, or cloud VM with CI/CD.

Timeline (Suggested)

Week 1: Build UI + Node API, prepare FAQ data.

Week 2: Implement keyword matching + feedback.

Week 3: Integrate TF.js model or OpenAI fallback.

Week 4: Test, deploy, and collect logs.

Future Enhancements

Contextual multi-turn dialogues, rich responses (links, images), analytics dashboards, multi-language support.

Conclusion

This chatbot automates FAQ handling with AI/ML. Combining compact models, feedback loops, and iterative improvements ensures accuracy while keeping costs low and deployment simple.