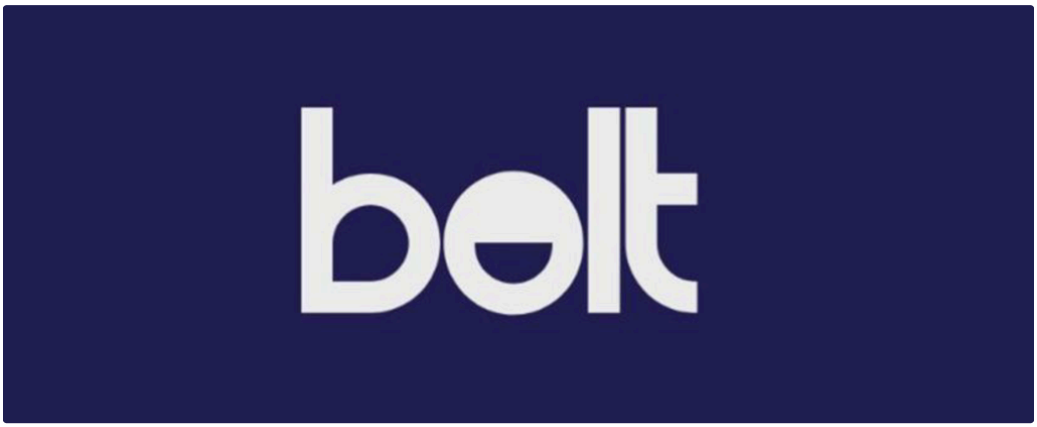
Technical Case Document: Chatbot Application



Assignment

In this project, you will develop a simple chatbot application using the following technologies:

- **Frontend:** ReactJS
- **Backend:** NestJS or ExpressJS (your choice)
- **Database:** MongoDB
- **Bonus:** Incorporate OpenAI API for generating questions or enhancing user responses.

The chatbot should ask the user **10 predefined questions** (you can come up with them) and store the user's answers in MongoDB. The objective is to build both the frontend and the backend that communicates effectively and provides a smooth user experience.

Bonus

You can integrate OpenAI's API to:

- Dynamically generate questions.
- Improve chatbot responses.

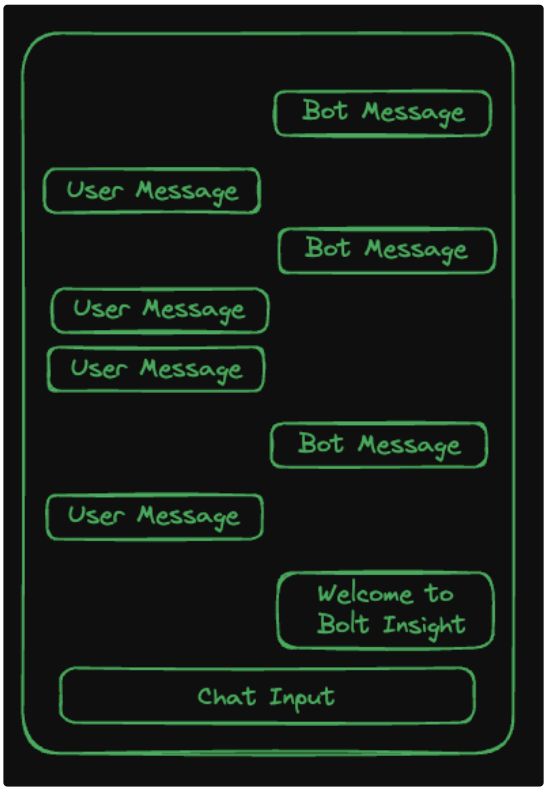
Application Requirements

Functional Requirements

Chatbot Interface (Frontend)

- The user interface should allow interaction with the chatbot.
- The chatbot will ask 10 questions one by one, and the user will respond.
- **Questions:**
  - What is your favorite breed of cat, and why?
  - How do you think cats communicate with their owners?
  - Have you ever owned a cat? If so, what was their name and personality like?
  - Why do you think cats love to sleep in small, cozy places?

- What's the funniest or strangest behavior you've ever seen a cat do?
- Do you prefer cats or kittens, and what's the reason for your preference?
- Why do you think cats are known for being independent animals?
- How do you think cats manage to land on their feet when they fall?
- What's your favorite fact or myth about cats?
- How would you describe the relationship between humans and cats in three words?



Backend

- The backend should be built using **NestJS** or **ExpressJS**.
- Create session to management to track which user is on which question.
- The chatbot will ask 10 questions one by one, and the user will respond.

Database (MongoDB)

- The responses should be saved in a **MongoDB** database with proper schema design.
- Each user session should have:
  - A unique identifier.
  - Timestamps for when the session started and ended.
  - The list of questions and answers.
- You can use **Mongoose** or any other ODM (Object Data Modeling) tool.

OpenAI API (Optional)

If integrated, OpenAI API can be used for the following:

- **Generate questions dynamically:** Instead of predefined questions, use the API to generate relevant or personalized questions.
- **Enhance user interactions:** Make the chatbot more conversational and intelligent by generating responses to the user's input instead of using predefined questions.

Non-Functional Requirements

- The chatbot should handle errors gracefully (e.g., network errors, server issues).
- Responses should be securely stored in the database.
- Follow good coding practices to ensure clean, maintainable, and well-documented code in both the frontend and backend.

Review

- Upload your codes to Github
- Share your repository with Burak Yuksel
- Burak Yuksel(Babousn): github.com/babousn