Q/A ChatBot – Project Documentation

1. Project Title

Q/A ChatBot with Login Authentication (Firebase + HuggingFace + Streamlit)

---

2. Problem Statement

Users often need a simple, fast, and secure platform where they can ask questions and receive AI-based responses.
Most AI tools do not include user authentication or personalized access, making them unsuitable for private or controlled usage.

This project solves that problem by creating a **secure login-based chatbot system** using Firebase authentication and HuggingFace LLM models.

---

3. Objective of the Project

The main objective of this project is to build a **secure, lightweight, and easy-to-use AI chatbot** with:

- Email/password login system
- Token-based secure authentication
- Interactive Q/A chatbot
- Real-time AI responses using HuggingFace API
- Clean and minimal UI using Streamlit

---

4. Technologies Used

Frontend / UI

- **Streamlit** (for building the web interface)

Backend

- **Python**
- **HuggingFace LLM API**
- **Firebase Authentication (Pyrebase)**

Environment Handling

- `.env` file
- `python-dotenv`

---

5. System Architecture

1. User Interface (Streamlit)

- Login Page
- Chat Interface
- Session Management

2. Firebase Authentication

- Verifies email & password
- Provides secure login
- Stores session user ID

3. HuggingFace LLM API

- Accepts user message
- Sends it to HF model

- Returns AI-generated response

4. Chat Display

- Shows the conversation in order
- Maintains chat history in `session_state`

---

6. Key Features

🔐 User Authentication

- Firebase email/password login
- Wrong login → error message
- Session stored until logout

🤖 AI Chatbot

- Uses **Llama 3.2 3B Instruct** model
- Maintains conversation history
- Handles multiple user queries

🧠 Session Management

- Stores:
    - User state
    - Chat history
    - Bot responses

🧹 Reset and Logout

- Sidebar reset button

- Logout clears session

---

7. How It Works (Workflow)

Step 1 – User Opens the App

Streamlit loads and shows **Login Page**.

Step 2 – User Enters Credentials

Firebase verifies the credentials.

Step 3 – Redirect to ChatBot Page

Shows:

- Welcome message
- Chat input box

Step 4 – User Sends Message

The message is forwarded to HuggingFace API.

Step 5 – Bot Generates Response

Response displayed in chat window.

Step 6 – Logout

Session resets; user returns to login screen.

---

8. Code Overview (Short Summary)

main.py responsibilities:

- Setup Streamlit UI
- Handle Firebase login

- Call HuggingFace API
- Display chat
- Manage session history

Important Functions

- `query_hf()` → Sends message to HF model
- `sign_in_with_email_and_password()` → Firebase login
- `session_state` → Stores states

---

9. Challenges Faced

1. Firebase databaseURL missing
2. HuggingFace model not supported
3. Session issues due to missing `st.set_page_config()`
4. Slow inference due to free API limits

---

10. Future Enhancements

- Add UI styling
- Add image support
- Add role-based access
- Deploy on cloud
- Add speech-to-text input
- Integrate vector search for memory chatbot

---

11. Conclusion

This project successfully builds a **secure and functional AI chatbot** integrating:

- Firebase login
- HuggingFace AI model
- Streamlit UI

It demonstrates the integration of authentication + AI + frontend in a simple yet powerful way.