Project Summary: Balasore Alloys AI Chat Support Chatbot

Purpose:

Create an AI-powered chatbot embedded on the Balasore Alloys website to answer general queries about the company, products, contact details, and location. The goal is to provide instant, interactive customer support using AI.

Main Components:

1.Intent Classifier
An NLP model that understands what users want by classifying their messages into predefined intents (e.g., greetings, product inquiries).

2.Backend API Server
A Flask-based web service that receives chat messages, uses the intent classifier to determine intent, and returns text responses.

3.Frontend Chat UI
A browser-based chat interface where users type messages, view responses, and interact with the chatbot.

Detailed Steps

1. Planning & Data Preparation
Identify key user intents relevant to Balasore Alloys:
Greetings, Company Info, Product Inquiry, Contact Info, Location, Feedback, Goodbye.
Collect FAQs and content from the company website.
Prepared example user sentences mapped to each intent as training data.

2. Build Intent Classifier
Used Python's scikit-learn to create a simple NLP pipeline:
Convert text with TF-IDF vectorizer.
Trained logistic regression on example inputs to classify intents.
Save the trained model to a file (intent_classifier.pkl) for reuse.

3. Develop Backend Server (Flask)
Create REST API endpoint /chat that accepts POST requests with user messages.
Load the saved intent classifier model.
Predict the intent of incoming messages.
Return a predefined response mapped to the predicted intent.
Handle unknown inputs with a default fallback message.
Optionally enable CORS so frontend can communicate across domains.

4. Create Frontend Chat Interface
Build the UI in HTML, CSS, and JavaScript:
Chatbox to display messages.
Input field with send button.

JavaScript to send user messages to backend API and display bot replies.
Simple, user-friendly black & white themed design.

5. Run and Test
Train intent classifier and save model.
Start Flask backend server locally.
Open frontend in a web browser served via HTTP to prevent CORS errors.
Test chatbot by sending queries like “hello”, “product details”, “contact number”.

Technology Stack

Component	Technology
NLP & Intent Classifier	Python (scikit-learn)
Backend API	Python Flask (+ flask-cors)
Frontend UI	HTML, CSS, JavaScript
Data Storage	File-based model storage (pickle)
Development	Local machine, terminal, code editor

How It Works Together

User types in the browser chat UI.

Frontend sends the message to Flask server at /chat.

Flask loads the model and predicts the intent.

Flask returns a canned response mapping for the intent.

Frontend displays the response as chatbot message.

Potential Extensions

Add multi-turn dialogue with context handling.
Use advanced NLP models (e.g., transformer-based) for better understanding.
Connect to real customer service or CRM platforms for human handoff.
Add logging, analytics, and usage tracking.
Support multilingual queries.
Host backend and frontend on cloud for production deployment.

Setup Instructions

Clone or extract the project folder.

Set up a Python virtual environment and install dependencies:

python -m venv venv

Activate the virtual environment:

Windows:

venv\Scripts\activate

macOS/Linux:

source venv/bin/activate

Install required packages:

pip install flask scikit-learn flask-cors joblib

Train the intent classifier:

Run the training script to build and save the ML model:

python train_intent_classifier.py

Start the Flask backend server:

python app.py

The server runs locally on: http://127.0.0.1:5000

Open the frontend UI:

Open the **index.html** file in your browser.

For best results, serve it through a local HTTP server, e.g.,

python -m http.server 8000

Then visit **http://localhost:8000** in your browser.

Test the chatbot:

Type queries like “hello”, “where are you located?”, or “product details” to interact with the bot.

balasore_chatbot/

```
|
├── train_intent_classifier.py   # Script to train the intent classifier
├── app.py                       # Flask backend API server
├── intent_classifier.pkl        # Saved ML model after training
├── index.html                   # Frontend chat UI
└── README.md                    # This file
```

Additional Notes

The chatbot currently does not save chat history.
CORS is enabled on the backend to allow frontend requests.
The project can be enhanced by adding context handling, persistent sessions, and connecting to databases.

Contact

For any questions or clarifications, please contact:

Kamlesh Ojha
Email: kamleshojha90@gmail.com
Phone: +91-7873360250