

# AI-Powered Chatbot Project Documentation

## Table of Contents

1. Introduction
2. Project Overview
3. Installation and Setup
4. Database Schema
5. API Endpoints
6. Functional Requirements
7. Testing and Debugging
8. Conclusion
9. Screenshots

## Introduction

This document provides a comprehensive overview of the Chatbot Project developed for the assignment. It includes details about the project's setup, database schema, API endpoints, functional requirements, and testing procedures.

## Project Overview

The Chatbot Project is designed to provide users with information about products and suppliers using a natural language interface.

The chatbot can handle queries such as:

- "Show me all products under brand X."
- "Which suppliers provide laptops?"
- "Give me details of product ABC."

## Installation and Setup

Prerequisites:

- Python 3.7+
- MySQL
- FastAPI
- Uvicorn
- PyTorch
- Additional Python packages (specified below)

Steps:

1. Clone the Repository:

- ```
git clone <repository-url>
cd chatbot-frontend/backend
```
2. Create a Virtual Environment and Activate it:

```
python -m venv venv
venv\Scripts\activate # For Windows
```
  3. Install Required Packages:

```
pip install fastapi uvicorn sqlalchemy pymysql transformers torch
```
  4. Set Up the Database:
    - Create a MySQL database named chatbot\_db.
    - Run the following SQL commands to create tables and insert sample data:

```
USE chatbot_db;
```

```
CREATE TABLE Suppliers (  
    supplier_id INT AUTO_INCREMENT PRIMARY KEY,  
    name VARCHAR(255) NOT NULL,  
    contact_info VARCHAR(255),  
    product_categories VARCHAR(255)  
);
```

```
CREATE TABLE Products (  
    product_id INT AUTO_INCREMENT PRIMARY KEY,  
    name VARCHAR(255) NOT NULL,  
    brand VARCHAR(255),  
    price DECIMAL(10,2),  
    category VARCHAR(255),  
    description TEXT,  
    supplier_id INT,  
    FOREIGN KEY (supplier_id) REFERENCES Suppliers(supplier_id)  
);
```

```
INSERT INTO Suppliers (name, contact_info, product_categories) VALUES  
('TechCorp', 'techcorp@example.com, 123-456-7890', 'Electronics, Computers'),  
('OfficeSuppliesCo', 'office@example.com, 098-765-4321', 'Furniture, Office Supplies'),  
('BookWorld', 'books@example.com, 555-123-4567', 'Books, Stationery');
```

```
INSERT INTO Products (name, brand, price, category, description, supplier_id) VALUES  
('Laptop', 'BrandX', 999.99, 'Electronics', 'High-performance laptop', 1),  
('Office Chair', 'BrandY', 149.99, 'Furniture', 'Ergonomic office chair', 2),  
('Notebook', 'BrandZ', 2.99, 'Stationery', '100-page notebook', 3),  
('Smartphone', 'BrandX', 599.99, 'Electronics', 'Latest model smartphone', 1),  
('Desk', 'BrandY', 249.99, 'Furniture', 'Spacious office desk', 2),
```

```
('Pen', 'BrandZ', 1.49, 'Stationery', 'Smooth-writing pen', 3);
```

5. Run the FastAPI Server:

```
uvicorn main:app --reload
```

## Database Schema

Suppliers Table:

- supplier\_id: INT, Primary key, auto-increment
- name: VARCHAR(255), Supplier name
- contact\_info: VARCHAR(255), Contact information
- product\_categories: VARCHAR(255), Categories of products supplied

Products Table:

- product\_id: INT, Primary key, auto-increment
- name: VARCHAR(255), Product name
- brand: VARCHAR(255), Product brand
- price: DECIMAL, Product price
- category: VARCHAR(255), Product category
- description: TEXT, Product description
- supplier\_id: INT, Foreign key to suppliers table

## API Endpoints

/chat (POST):

Description: Handles user queries and returns appropriate responses.

Request Example:

```
{  
  "question": "Show me all products under brand BrandX."  
}
```

Response Example:

```
{  
  "answer": "Products under brand BrandX: ['Laptop', 'Smartphone']"  
}
```

## Functional Requirements

1. Show me all products under brand X:

- Extracts the brand name from the query and retrieves products under that brand from

the database.

2. Which suppliers provide laptops?

- Extracts the product category from the query and retrieves suppliers providing that category from the database.

3. Give me details of product ABC:

- Extracts the product name from the query and retrieves product and supplier details from the database.

## Testing and Debugging

1. Debug Prints: Added debug print statements in main.py to track data processing.

2. Postman: Used Postman to send test requests and verify API responses.

3. Logs: Checked terminal logs for any errors and resolved them accordingly.

## Conclusion

This project successfully implements a chatbot that can handle various user queries related to products and suppliers. The chatbot interacts with a database to fetch relevant information and provides accurate responses.

# Screenshots

## 1. Screenshot of Database Setup:

Description: Screenshot showing the database tables and sample data.

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

|   | supplier_id | name             | contact_info                       | product_categories         |
|---|-------------|------------------|------------------------------------|----------------------------|
| ▶ | 1           | TechCorp         | techcorp@example.com, 123-456-7890 | Electronics, Computers     |
|   | 2           | OfficeSuppliesCo | office@example.com, 098-765-4321   | Furniture, Office Supplies |
|   | 3           | BookWorld        | books@example.com, 555-123-4567    | Books, Stationery          |
| * | NULL        | NULL             | NULL                               | NULL                       |

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

|   | product_id | name         | brand  | price  | category    | description             | supplier_id |
|---|------------|--------------|--------|--------|-------------|-------------------------|-------------|
| ▶ | 1          | Laptop       | BrandX | 999.99 | Electronics | High-performance laptop | 1           |
|   | 2          | Office Chair | BrandY | 149.99 | Furniture   | Ergonomic office chair  | 2           |
|   | 3          | Notebook     | BrandZ | 2.99   | Stationery  | 100-page notebook       | 3           |
|   | 4          | Smartphone   | BrandX | 599.99 | Electronics | Latest model smartphone | 1           |
|   | 5          | Desk         | BrandY | 249.99 | Furniture   | Spacious office desk    | 2           |
|   | 6          | Pen          | BrandZ | 1.49   | Stationery  | Smooth-writing pen      | 3           |
| * | NULL       | NULL         | NULL   | NULL   | NULL        | NULL                    | NULL        |

Query 1

Limit to 1000 row

1

USE chatbot\_db;

Execute the statement under the keyboard cursor

3

CREATE TABLE Suppliers (

4

supplier\_id INT AUTO\_INCREMENT PRIMARY KEY,

5

name VARCHAR(255) NOT NULL,

6

contact\_info VARCHAR(255),

7

product\_categories VARCHAR(255)

8

);

9

CREATE TABLE Products (

10

product\_id INT AUTO\_INCREMENT PRIMARY KEY,

11

name VARCHAR(255) NOT NULL,

12

brand VARCHAR(255),

13

price DECIMAL(10,2),

14

category VARCHAR(255),

15

description TEXT,

16

supplier\_id INT,

17

FOREIGN KEY (supplier\_id) REFERENCES Suppliers(supplier\_id)

18

);

19

INSERT INTO Suppliers (name, contact\_info, product\_categories) VALUES

20

('TechCorp', 'techcorp@example.com, 123-456-7890', 'Electronics, Computers'),

21

('OfficeSuppliesCo', 'office@example.com, 098-765-4321', 'Furniture, Office Supplies'),

22

('BookWorld', 'books@example.com, 555-123-4567', 'Books, Stationery');

23

INSERT INTO Products (name, brand, price, category, description, supplier\_id) VALUES

24

('Laptop', 'BrandX', 999.99, 'Electronics', 'High-performance laptop', 1),

25

('Office Chair', 'BrandY', 149.99, 'Furniture', 'Ergonomic office chair', 2),

26

('Notebook', 'BrandZ', 2.99, 'Stationery', '100-page notebook', 3),

27

('Smartphone', 'BrandX', 599.99, 'Electronics', 'Latest model smartphone', 1),

28

('Desk', 'BrandY', 249.99, 'Furniture', 'Spacious office desk', 2),

29

('Pen', 'BrandZ', 1.49, 'Stationery', 'Smooth-writing pen', 3);

30

## 2. Screenshot of FastAPI Server Running:

Description: Screenshot showing the FastAPI server running in the terminal.

```
PS C:\Users\rohit\OneDrive\Desktop\chatbot-frontend> cd C:\Users\rohit\OneDrive\Desktop\chatbot-frontend\backend
>> venv\Scripts\activate
(venv) PS C:\Users\rohit\OneDrive\Desktop\chatbot-frontend\backend> uvicorn main:app --reload
>>
INFO: Will watch for changes in these directories: ['C:\\Users\\rohit\\OneDrive\\Desktop\\chatbot-frontend\\backend']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [18124] using StatReload
INFO: Started server process [1568]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

```
PS C:\Users\rohit\OneDrive\Desktop\chatbot-frontend> cd C:\Users\rohit\OneDrive\Desktop\chatbot-frontend\backend
>> venv\Scripts\activate
(venv) PS C:\Users\rohit\OneDrive\Desktop\chatbot-frontend\backend> uvicorn main:app --reload
>>
INFO: Will watch for changes in these directories: ['C:\\Users\\rohit\\OneDrive\\Desktop\\chatbot-frontend\\backend']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [18124] using StatReload
INFO: Started server process [1568]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

```
INFO: 127.0.0.1:35592 - "OPTIONS /chat HTTP/1.1" 200 OK
User question: "Show me all products under brand BrandX."
Cleaned question: show me all products under brand brandx
Brand: x
Products found for brand 'x': [<main.Product object at 0x0000017A800510D0>, <main.Product object at 0x0000017A80051150>]
Products: [<main.Product object at 0x0000017A800510D0>, <main.Product object at 0x0000017A80051150>]
Response: Products under brand x: ['Laptop', 'Smartphone']
```

```
INFO: 127.0.0.1:35594 - "POST /chat HTTP/1.1" 200 OK
User question: "Give me details of product Pen."
Cleaned question: give me details of product pen
Product name: pen
Product details for 'pen': <main.Product object at 0x0000017A8005CA50>
Product found: <main.Product object at 0x0000017A8005CA50>
The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass your input's 'attention_mask' to obtain reliable result s.
Setting 'pad_token_id' to 'eos_token_id':50256 for open-end generation.
The attention mask is not set and cannot be inferred from input because pad token is same as eos token. As a consequence, you may observe unexpected behavior. Please pass your input's 'attention_mask' to obtain reliable results.
Summarized info: Summarize the following supplier information: Supplier: BookWorld, Contact: books@example.com, 555-123-4567, USA Address: "BookWorld, Inc," Fax: 555-123-4567, USA
```

### 3. Screenshot of Frontend with Query Input:

Description: Screenshot showing the frontend interface with a query input box.

