

# GenAI Chatbot - Chatbot Evolution

---

A versatile domain-adaptable GenAI chatbot that provides natural conversational interactions across multiple business domains. Built for the Chatbot Evolution Hackathon, this system showcases seamless adaptability from restaurant ordering to clinic appointments and beyond.




Made with  React  Firebase  Google Gemini  Chakra UI

## Deployment

Deployed the project on Netlify <https://chatbot-evolution.netlify.app/>

Video Demonstration: [https://drive.google.com/drive/folders/1gWyZZSborBARvexsBiYeG-gGy8ZDgk7m?usp=drive\\_link](https://drive.google.com/drive/folders/1gWyZZSborBARvexsBiYeG-gGy8ZDgk7m?usp=drive_link)

## Key Features

- **Multi-Domain Support**
  -  Restaurant Ordering System
  -  Clinic Appointment Booking
  -  Hotel Reservations (Extended Domain Example)
- **Smart Interactions**
  - Natural language processing with Google's Gemini AI
  - Context-aware conversations
  - Intelligent response enhancement with domain data
- **Advanced UI/UX**
  - Responsive design for all devices
  - Dark/Light theme support
  - Real-time typing indicators
  - Message markdown support
  - Intuitive chat interface
- **Technical Excellence**
  - Domain adapter pattern for easy extensibility
  - Smart caching for performance optimization
  - Offline support capabilities
  - Error boundary protection
  - Extensive accessibility features

## Getting Started

### Prerequisites

- Node.js (v14 or higher)
- npm/yarn
- Google Cloud Account (for Gemini AI)
- Firebase Account

## Setting Up Firebase

1. Create a new Firebase project at [Firebase Console](#)
2. Enable Authentication and Firestore
3. Get your Firebase configuration:

```
const firebaseConfig = {  
  apiKey: "your-api-key",  
  authDomain: "your-auth-domain",  
  projectId: "your-project-id",  
  storageBucket: "your-storage-bucket",  
  messagingSenderId: "your-messaging-sender-id",  
  appId: "your-app-id"  
};
```

4. Create a `.env` file in the project root and add:

```
REACT_APP_FIREBASE_CONFIG=your-stringified-config
```

## Setting Up Google Generative AI

1. Visit [Google AI Studio](#)
2. Create an API key
3. Add to your `.env`:

```
REACT_APP_GEMINI_API_KEY=your-api-key
```

## Installation

```
# Clone the repository  
git clone https://github.com/yashveer132/Chatbot-evolution.git  
  
# Install dependencies  
cd chatbot-evolution  
npm install  
  
# Start development server  
npm start
```

## 💡 Usage Examples

### Restaurant Assistant

```
User: "What vegetarian options do you have?"
AI: "We have several vegetarian dishes:
    • Vegetarian Buddha Bowl ($12.99)
    • Mushroom Risotto ($13.99)
    • Quinoa Power Bowl ($16.99)
    All are prepared fresh and can be customized to your preferences."
```

### Clinic Assistant

```
User: "I need an appointment with a cardiologist"
AI: "I can help you with that. Dr. Sarah Johnson, our cardiologist, has the
    following slots available:
    • Tuesday: 2:00 PM, 3:30 PM
    • Thursday: 10:00 AM, 11:30 AM
    The consultation fee is $150. Would you like to book an appointment?"
```

### Hotel Assistant

```
User: "What rooms are available next weekend?"
AI: "For next weekend, we have:
    • Standard King Room ($199/night)
    • Deluxe King Room ($249/night)
    Both include free WiFi, daily housekeeping, and city views."
```

## 🏗️ Architecture

```
src/
├── components/           # Reusable UI components
│   ├── chat/            # Chat interface components
│   ├── common/          # Shared components
│   └── auth/             # Authentication components
├── context/             # React contexts
├── data/                 # Domain-specific data
├── services/            # External service integrations
├── utils/                # Utility functions
└── pages/                # Main page components
```

## 🎯 Meeting Hackathon Criteria

Portability 🌟

- Domain adapter pattern enables quick integration of new domains
- Consistent interface across different business contexts
- Modular architecture for easy adaptation

## Code Quality

- Clean code architecture with separation of concerns
- Comprehensive error handling
- Consistent coding standards

## Versatility

- Successfully implemented 3 distinct domains
- Extendable prompt enhancement system
- Flexible data structure adaptation

## Usability

- Intuitive chat interface
- Real-time response indicators
- Mobile-responsive design
- Accessibility features

## Efficiency

- Smart caching system
- Optimized API calls
- Local data enhancement
- Progressive loading

## Innovation

- Context-aware conversations
- Dynamic response formatting
- Multi-modal input support
- Seamless domain switching

## Team

- Yash Veer Singh (AJAX)

---

Made with  for Chatbot-Evolution