

AI-Powered Customer Service

AI-powered customer service platform with automatic message analysis, issue classification, sentiment understanding, and intelligent response suggestions.

Features

-  **Dual Authentication System**
 - Customer login at [/login](#) (email/password or Google)
 - Support Agent login at [/agent](#) (email/password + agent key)
 - Separate API endpoints for security
-  **Role-Based Access Control**
 - Customer dashboard for submitting support requests
 - Agent dashboard with AI-powered tools
 - Automatic role-based routing
-  **AI-Powered Tools** (for agents)
 - Conversation analysis
 - Issue classification
 - Sentiment analysis
 - Response suggestions

Getting Started

Prerequisites

- Node.js 18+
- Firebase project with Firestore and Authentication enabled
- Firebase Admin SDK credentials

Installation

PROF

1. Clone the repository:

```
git clone <repository-url>
cd ai-customerservice
```

2. Install dependencies:

```
npm install
```

3. Setup environment variables:

Create a [.env.local](#) file in the root directory:

```

# Firebase Admin SDK
FIREBASE_PROJECT_ID=your-project-id
FIREBASE_PRIVATE_KEY="-----BEGIN PRIVATE KEY-----\n...\\n-----END PRIVATE
KEY-----\\n"
FIREBASE_CLIENT_EMAIL.firebaseio-adminsdk-xxxxx@your-
project.iam.gserviceaccount.com

# Firebase Web Config
NEXT_PUBLIC_FIREBASE_API_KEY=your-api-key
NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN=your-project.firebaseio.com
NEXT_PUBLIC_FIREBASE_PROJECT_ID=your-project-id
NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET=your-project.firebaseiostorage.app
NEXT_PUBLIC_FIREBASE_MESSAGING_SENDER_ID=123456789
NEXT_PUBLIC_FIREBASE_APP_ID=1:123456789:web:xxxxx

# Server Configuration
PORT=3001

# Agent Key (CHANGE THIS IN PRODUCTION!)
AGENT_KEY=support-agent-key-2026-secure

```

4. Run the development server:

```
npm run dev
```

5. Open <http://localhost:3001> in your browser.

Agent Key Setup

The application uses an **Agent Key** system to authenticate support agents. See [AGENT_KEY_SETUP.md](#) for detailed instructions.

PROF

Quick Start:

Customer Login:

- Navigate to [/login](#) or click "Continue as Customer" from homepage
- Login with email/password or Google
- Redirects to [/customer](#) dashboard

Support Agent Login:

- Navigate to [/agent](#) or click "Continue as Agent" from homepage
- Enter email, password, AND agent key
- Default agent key: [support-agent-key-2026-secure](#) (change in production!)
- Redirects to [/agent/dashboard](#)

Project Structure

```
src/
├── app/
│   ├── agent/
│   │   ├── page.js          # Agent login page
│   │   └── dashboard/
│   │       └── page.js      # Agent dashboard (protected)
│   ├── (public)/
│   │   ├── customer/       # Customer dashboard
│   │   ├── login/          # Customer login page
│   │   └── register/       # Registration page
│   └── api/
│       └── auth/
│           ├── login/        # Customer login API
│           └── agent/
│               └── login/    # Agent login API (with key)
└── components/
    └── auth/              # Auth components (LoginForm, etc.)
└── contexts/
    └── AuthContext.js     # Authentication context
└── lib/
    ├── firebase.js         # Firebase client config
    └── firebaseAdmin.js    # Firebase Admin SDK config
```

Authentication Flow

User → Homepage

```
PROF
    |-> Customer
        |-> Navigate to /login
        |-> Email/Password or Google
        |-> API: /api/auth/login
        |-> Redirect to /customer

    |-> Support Agent
        |-> Navigate to /agent
        |-> Email/Password + Agent Key
        |-> API: /api/auth/agent/login
        |-> Verify agent key
        |-> Update role to 'agent'
        |-> Redirect to /agent/dashboard
```

Security

- ✓ Separate login endpoints for customer and agent
- ✓ Agent key verification on dedicated endpoint
- ✓ Role-based access control

- ✓ Firebase Authentication integration
- ✓ Server-side token verification
- ✓ Automatic role assignment
- ✓ Security isolation between customer and agent flows

Important: Always use strong, random agent keys in production and rotate them regularly!

API Routes

- POST `/api/auth/register` - User registration
- POST `/api/auth/login` - Customer login
- POST `/api/auth/agent/login` - Agent login with key verification
- POST `/api/auth/verify` - Token verification

Technologies

- Frontend:** Next.js 14, React, Tailwind CSS
- Authentication:** Firebase Authentication
- Database:** Cloud Firestore
- Backend:** Next.js API Routes, Firebase Admin SDK

Learn More

- [Next.js Documentation](#)
- [Firebase Documentation](#)
- [Agent Key Setup Guide](#)

License

This project is licensed under the MIT License.