Citizen AI – Intelligent Citizen Engagement Platform

# 1. Introduction

Project Title: Citizen AI – Intelligent Citizen Engagement Platform

Team Members: YERUVA SRI DEEKSHITHA

Team ID : LTVIP2025TMID35138

# 2. Project Overview

Purpose:

Citizen AI is an intelligent civic engagement platform enabling real-time sentiment analysis and personalized AI responses to improve citizen services.

Features:

- Real-time chatbot for civic queries

- Sentiment analysis dashboard

- AI-generated responses

- Admin and citizen views

- MongoDB-powered data store

# 3. Architecture

Frontend:

Built with React.js, leveraging Axios for API requests and Bootstrap for UI components.

Backend:

Developed with Node.js and Express.js to manage RESTful APIs, sentiment routing, and authentication.

Database:

MongoDB is used for storing user data, feedback, and session logs.

# 4. Setup Instructions

Prerequisites:

- Node.js

- MongoDB

Installation:

1. Clone the repository:

git clone https://github.com/22pa1a05i8/Citizenai-SmartInternz.git

2. Navigate to project:

cd citizenai-smartinternz

3. Install dependencies:

npm install (in both client and server folders)

4. Set environment variables in `.env` file:

MONGO\_URI, SECRET\_KEY

# 5. Folder Structure

Client:

- src/

- components/

- pages/

- App.js

Server:

- routes/

- controllers/

- models/

- app.js

# 6. Running the Application

Frontend:

Run the following commands:

cd client

npm start

Backend:

Run the following commands:

cd server

npm start

# 7. API Documentation

Sample Endpoint:

GET /api/sentiment

Description: Analyze sentiment from user input

Response: {

sentiment: 'positive'

}

POST /api/chat

Description: Handle AI response

Request Body: {

message: 'What is garbage collection timing?'

}

# 8. Authentication

Authentication is handled using sessions and JWT tokens. Users log in and receive a secure token stored in local storage.

Routes are protected via middleware verification.

# 9. User Interface

The UI is clean, responsive, and accessible.

Pages:

- Home

- Chat Interface

- Dashboard (for admin)

- Login/Signup

**Technologies Used**

**Frontend**

* **React.js** – JavaScript library used to build the user interface.
* **React Router DOM** – Enables client-side routing for seamless navigation.
* **Axios** – Handles HTTP requests to the backend APIs.
* **Bootstrap** – Used for responsive design and pre-styled UI components.

**Backend**

* **Node.js** – JavaScript runtime used for executing backend logic.
* **Express.js** – Lightweight framework for building RESTful APIs.
* **Mongoose** – ODM (Object Data Modeling) library for MongoDB integration.
* **MongoDB** – NoSQL database used for storing user data, chat logs, and sentiment data.
* **dotenv** – Manages environment variables securely.
* **CORS** – Handles cross-origin resource sharing for secure API communication.

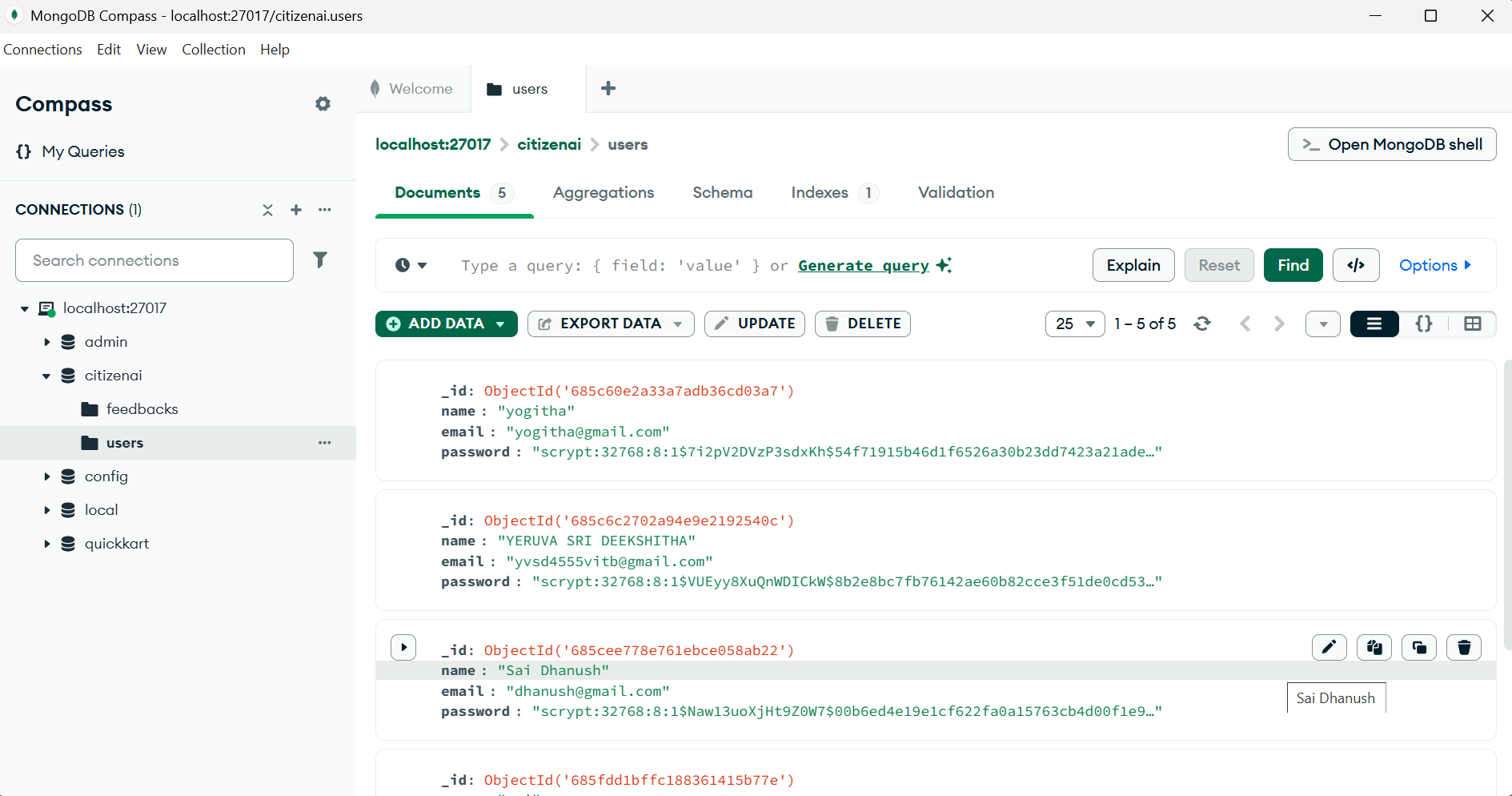
**Authentication**

* **Session-Based Authentication** – Login sessions are created and stored securely.
* **Middleware** – Used for route protection and user verification.

**Other Tools**

* **Chart.js** *(optional/expected)* – For rendering visual charts in the admin dashboard (based on UI structure).
* **Git & GitHub** – Version control and collaboration.
* **VS Code** – Code editor (assumed common dev tool).

**Project Working Screen Shots**:



Working of the MongoDB saving the details of the users and their feedbacks

