Flask-Based MongoDB Query System for Appointment Management

1. Project Overview

This project is a **Flask-based MongoDB query system** designed to manage and retrieve appointment details using natural language queries. The system allows users to fetch appointment information based on various criteria such as customer name, agent name, and date filters. It is deployed on a server to ensure accessibility.

2. Features

- **List all appointments**
- **Filter appointments by customer name**
- **Filter appointments by agent name**
- **Filter appointments by status**
- **Retrieve details of a specific appointment using an appointment ID**
- **Fetch confirmed appointments with date-based filters:**
    - Today
    - This week
    - Last week
    - Last N days

3. Technologies Used

- **Backend:** Flask (Python)
- **Database:** MongoDB Atlas
- **Frontend:** HTML, CSS, Javascript
- **Deployment:** Render
- **Libraries:**

i) Flask
ii) Pymongo
iii) MongoClient
iv) Datetime
v) Timedelta
vi) Pytz
vii) Re
viii) Parser

4. Project Structure

/ProjectR

├── app.py # Main Flask application

├── templates/

│ ├── web.html # Frontend UI

├── requirements.txt # Dependencies

├── mongo_config.py # Configuration settings

├── README.md # Project Documentation

5. Setup Instructions

5.1 Prerequisites

- Python (>= 3.8)
- MongoDB Atlas account
- Render account for deployment

5.2 Installation

1. Clone the repository:

   git clone < https://github.com/harshitabhatia25/runo>

2. Install dependencies:

   pip install -r requirements.txt

3. Set up environment variables for MongoDB connection:
4. export

   MONGO_URI="mongodb+srv://bhatiaharshita25:%40Mummy07@cluster0.wutsa1o.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0"

5. Run the Flask application:
6. python app.py

6. Deployment on Render

1. Push the project to GitHub.
2. In Render, create a **new web service** and connect it to your GitHub repository.
3. Set the **start command**: Python app.py

4. Add environment variables in Render's settings.
5. Deploy the application.
6.

7. Troubleshooting

Issue: TemplateNotFound: web.html

- Ensure web.html is inside the templates/ folder.
- Restart the Flask app after adding new templates.
- Check the deployment logs for missing files.

Issue: Database Connection Failure

- Verify MONGO_URI is set correctly.
- Check if MongoDB Atlas allows external connections.
- Restart the server.

8. Future Enhancements

- Implement user authentication.
- Add support for voice-based natural language queries.
- Develop a dashboard for better data visualization.

9. Conclusion

This Flask-based system provides an efficient way to manage and retrieve appointment details using MongoDB. The project demonstrates the integration of a web framework with a NoSQL database while ensuring scalability and ease of use.

Author: [Harshita Bhatia]
Date: [March 2025]