Unified AI Command Centre –

1. Project Overview

The **Unified AI Command Centre** is a full-stack communication dashboard that automates WhatsApp text, WhatsApp voice, and email notifications while supporting multi-step workflows and AI-based conversation understanding.

It enables operations teams to manage customers, send reminders, handle onboarding, track message history, and auto-respond using AI-detected **language, intent, and sentiment**.

The system supports **English, Hindi, Kannada, and Nepali**.

2. Key Features

Notification System

- WhatsApp Text (via API simulation)
- WhatsApp Voice (via TTS simulation)
- Email notifications
- Dynamic template variables (e.g., {name}, {salary_amount})

AI Processing

- Auto language detection
- Intent classification (completion, confusion, opt-out, etc.)
- Sentiment tagging (positive, negative, confused)
- Rule-based auto replies

Workflow Automation

Supports multi-step flows like:

- Onboarding Flow
- Salary Reminder Flow
- Document Reminder Flow

Triggers:

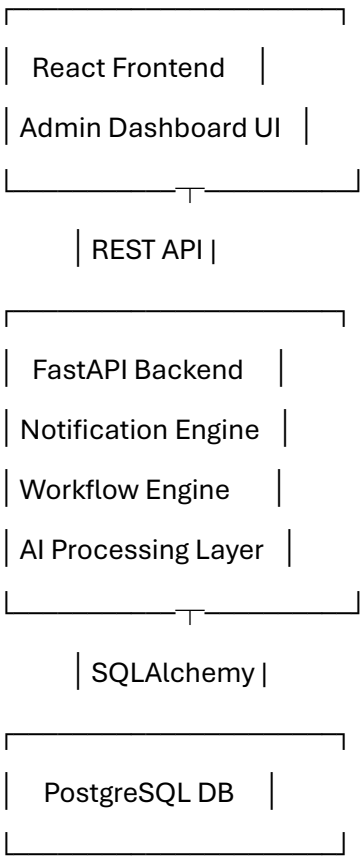
- Time-based
- Reply-based
- Event-based

Admin Dashboard (React)

- Add / manage users
- Create & manage templates

- Send notifications to selected users
- View real-time conversations timeline
- Clean, modern UI with dark theme

3. System Architecture



Backend also integrates:

- TTS mock service
- ASR mock service
- WhatsApp Webhook receiver

4. Tech Stack

Frontend:

- React (JS)
- React Router
- Custom CSS

Backend:

- FastAPI
- Python
- SQLAlchemy ORM
- Uvicorn

Database:

- SQLite (local development)
- PostgreSQL (production-ready)

5. API Endpoints (Summary)

Users

- GET /users – list users
- POST /users – create user

Templates

- GET /templates – list templates
- POST /templates – create template

Notifications

- POST /notifications/send – send WhatsApp/voice/email message

Conversations

- GET /conversations – conversation history

Webhooks

- POST /webhook/whatsapp – handle inbound replies

6. Project Folder Structure

unified-ai-command-centre/

```
│
├── backend/
│   ├── main.py
│   ├── models.py
│   ├── database.py
│   └── ...
│
```

```
├── frontend/
│   ├── src/
│   │   ├── pages/
│   │   ├── App.js
│   │   ├── api.js
│   │   └── styles.css
│   └── public/
│
└── README.md
```

7. Setup & Running

Backend (FastAPI)

cd backend

python -m venv venv

venv\Scripts\activate

pip install -r requirements.txt

python main.py

Backend runs at:

http://localhost:8000

http://localhost:8000/docs

Frontend (React)

cd frontend

npm install

npm start

Frontend runs at:

http://localhost:3000

8. How the System Works (Flow Summary)

Admin selects users + a template

Admin sends WhatsApp/email notifications from dashboard.

Backend merges templates

Dynamic data fills placeholders ({name}, {amount}, etc.)

Messages are sent

Via mock integrations.

User replies on WhatsApp

Webhook receives the reply → backend processes:

- ASR (if audio)
- Language detection
- Intent detection
- Sentiment analysis

Workflow status updates

System auto-replies or escalates based on rules.

Dashboard displays the conversations timeline.

9. Future Enhancements

- Real WhatsApp Cloud API integration
- Real TTS / ASR APIs
- ML-based intent classification
- Admin login & role-based access
- Analytics dashboard

10. License

MIT License (can be changed as needed)

11. Maintainer

Rushikesh Hadawale
Unified AI Command Centre — 2025