PROJECT REPORT

Project Title： Clinical Trial Patient Recruitment and Adherence Monitoring

Submitted by: Varun Panchal, Rajeshwari Acharya, Karanam Akhil, Nitisha Nagarkar

Company Name: Infotact Solutions

Date: 04/10/2025

INFOTACT

SOLUTIONS

1. Introduction

Clinical trials require efficient patient recruitment and tracking of protocol adherence across study sites. Dashboard solutions streamline workflow and improve data integrity.

2. Objectives of the Project

- Build a secure, scalable dashboard integrating multi-site EDC/EMR data.
- Visualize recruitment funnels and site KPIs.
- Provide actionable insights for trial managers.

3. Architecture & Technology

Layer	Technology	Description
Frontend	React	Dashboard UI, visualizations, live updates
Backend	FastAPI	REST APIs, business logic, authentication
Data Pipeline	Python/ETL	Clean, anonymize, transform raw exports
Storage	CSV/DB	Stores processed trial data

4. Key Features

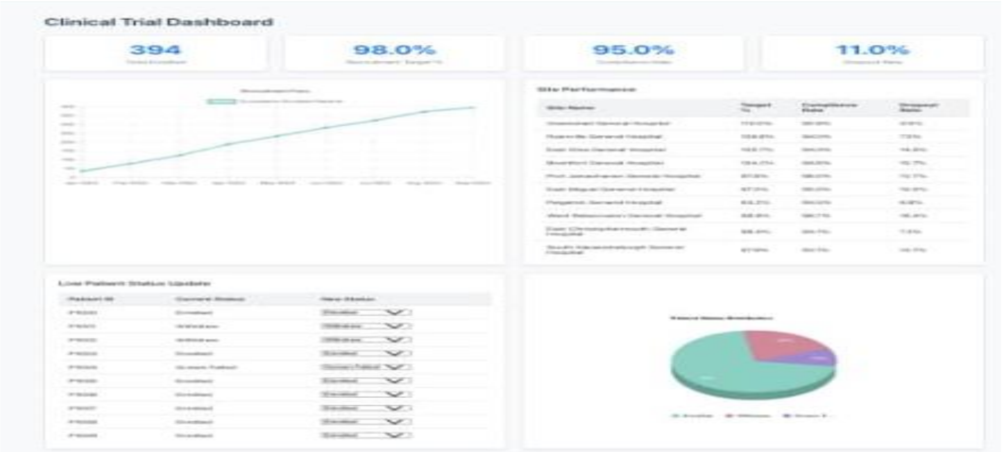
Multi-Site Data Integration:
Aggregates and standardizes exports from diverse sources.

Funnel Visualization:
Monitors screened, enrolled, and randomized patients.

Performance Leaderboard:
Tracks enrollment velocity and data quality per site.

Adherence Dashboard:
Flags at-risk patients based on metrics (missed visits, adherence %).

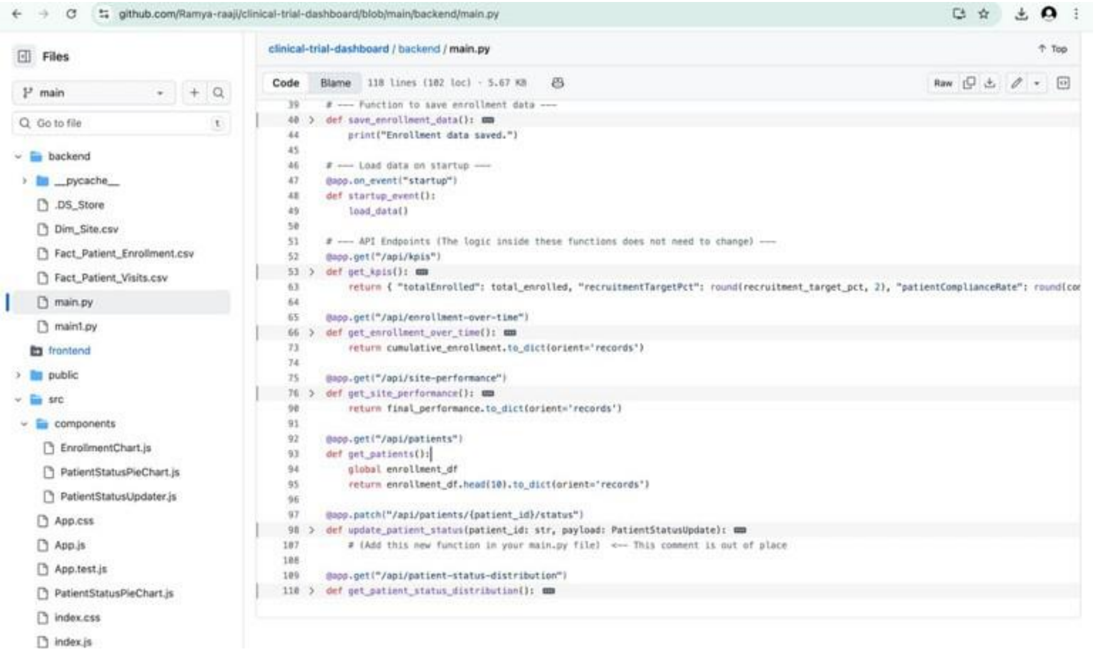
Patient Status Updater:
Enables managers to update status and see live impact.



5. Implementation Details

Backend (FastAPI)

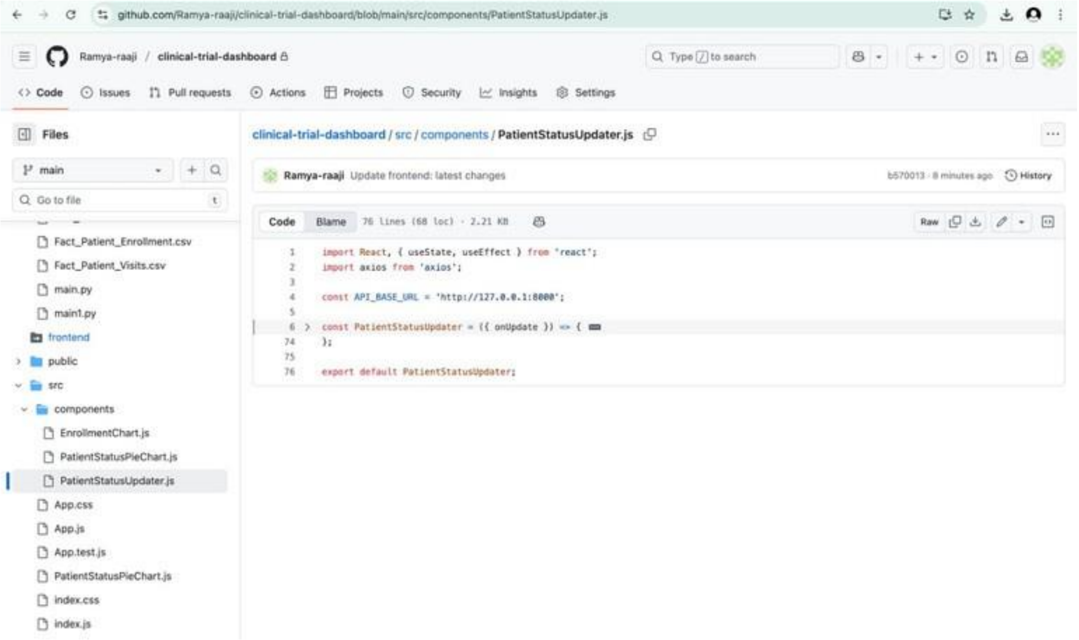
- API endpoints: patient enrollment, status distribution, update API, site leaderboards.
- Modular Python functions for ETL and transformations.
- Middleware for CORS, JWT/OAuth2 recommended for auth.



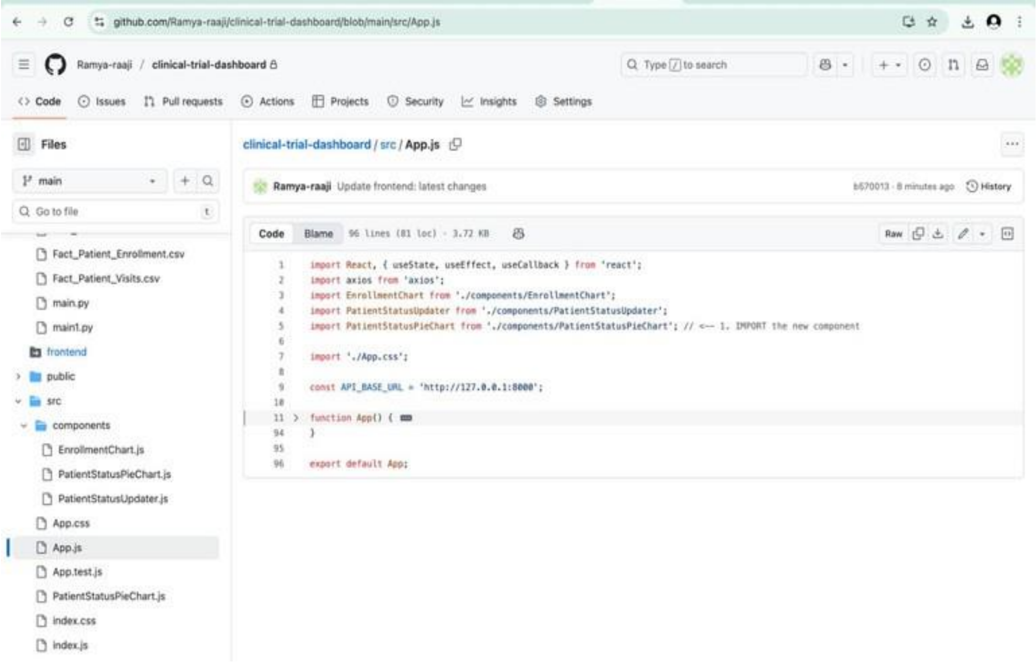
```
# --- Function to save enrollment data ---
def save_enrollment_data():
    print("Enrollment data saved.")

# --- Load data on startup ---
@app.on_event("startup")
def startup_event():
    load_data()

# --- API Endpoints (The logic inside these functions does not need to change) ---
@app.get("/api/kpis")
def get_kpis():
    return { "totalEnrolled": total_enrolled, "recruitmentTargetPct": round(recruitment_target_pct, 2), "patientComplianceRate": round(co

@app.get("/api/enrollment-over-time")
def get_enrollment_over_time():
    return cumulative_enrollment.to_dict(orient='records')

@app.get("/api/site-performance")
def get_site_performance():
    return final_performance.to_dict(orient='records')

@app.get("/api/patients")
def get_patients():
    global enrollment_df
    return enrollment_df.head(10).to_dict(orient='records')

@app.patch("/api/patients/{patient_id}/status")
def update_patient_status(patient_id: str, payload: PatientStatusUpdate):
    # (Add this new function in your main.py file)  <-- This comment is out of place

@app.get("/api/patient-status-distribution")
def get_patient_status_distribution():
```

Frontend (React)

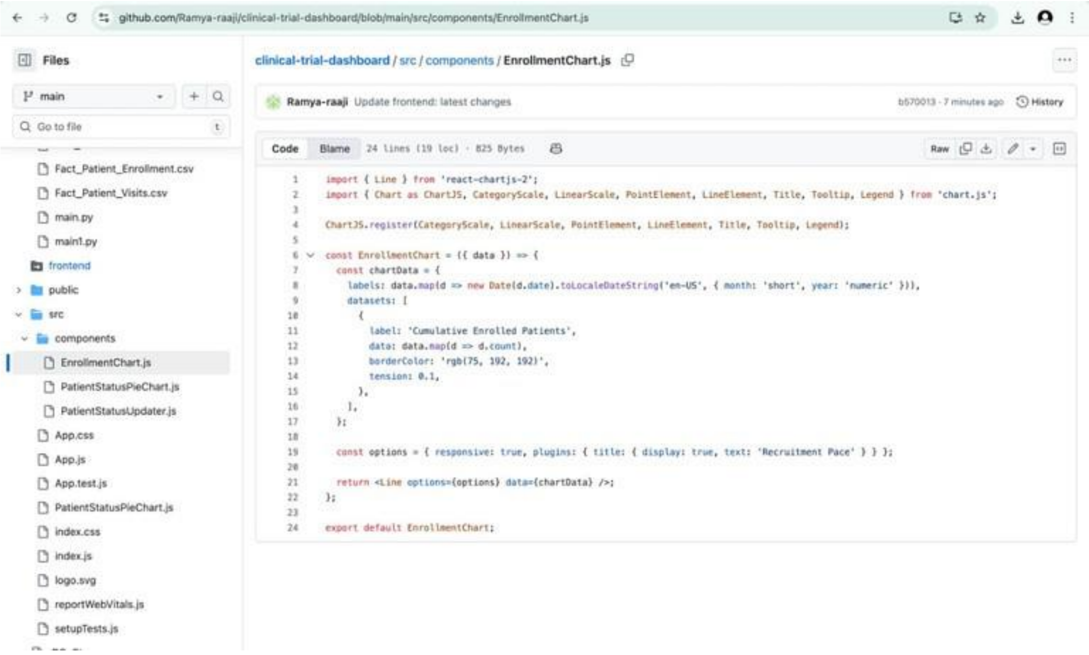
- Modular components for charts, tables, and live status updates (EnrollmentChart.js, PatientStatusPieChart.js, PatientStatusUpdater.js).
- State management for live updates.
- Integrated with backend endpoints via Axios.



```
import React, { useState, useEffect } from 'react';
import axios from 'axios';

const API_BASE_URL = 'http://127.0.0.1:8000';

const PatientStatusUpdater = ({ onUpdate }) => {
};

export default PatientStatusUpdater;
```



```
import React, { useState, useEffect, useCallback } from 'react';
import axios from 'axios';
import EnrollmentChart from './components/EnrollmentChart';
import PatientStatusUpdater from './components/PatientStatusUpdater';
import PatientStatusPieChart from './components/PatientStatusPieChart'; // <-- 1. IMPORT the new component

import './App.css';

const API_BASE_URL = 'http://127.0.0.1:8000';

function App() { ⋯
}

export default App;
```



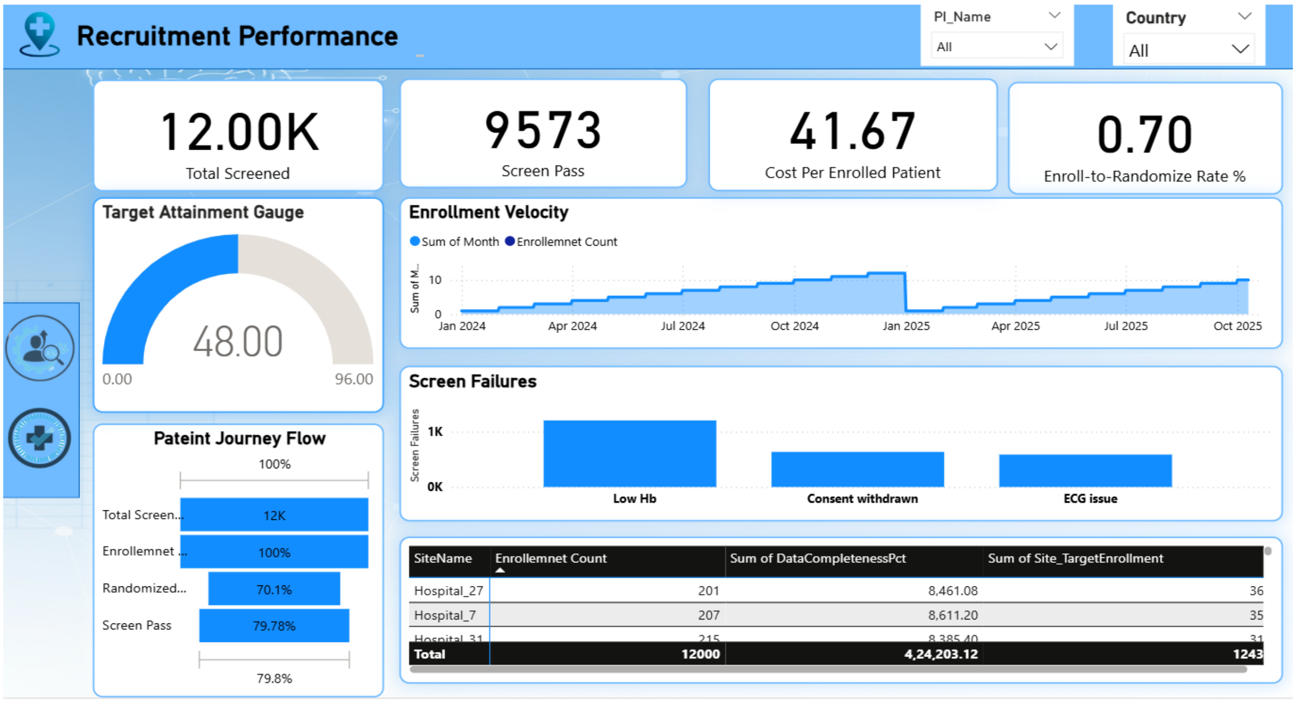
```
import { Line } from 'react-chartjs-2';
import { Chart as ChartJS, CategoryScale, LinearScale, PointElement, LineElement, Title, Tooltip, Legend } from 'chart.js';

ChartJS.register(CategoryScale, LinearScale, PointElement, LineElement, Title, Tooltip, Legend);

const EnrollmentChart = ({ data }) => {
  const chartData = {
    labels: data.map(d => new Date(d.date).toLocaleDateString('en-US', { month: 'short', year: 'numeric' })),
    datasets: [
      {
        label: 'Cumulative Enrolled Patients',
        data: data.map(d => d.count),
        borderColor: 'rgb(75, 192, 192)',
        tension: 0.1,
      },
    ],
  };

  const options = { responsive: true, plugins: { title: { display: true, text: 'Recruitment Pace' } } };

  return <Line options={options} data={chartData} />;
};

export default EnrollmentChart;
```

Data Model

- Star schema with fact tables (enrollment, visits, data quality) and dimension tables (site, patient, visit type).
- Sample pseudonymized CSV structures for privacy compliance.

6. Dashboard Demonstration

- Screenshot(s) of live dashboard visualizations and patient status management.
- Explanation of real-time updating mechanics and chart refresh.



PI_Name: All

Country: All

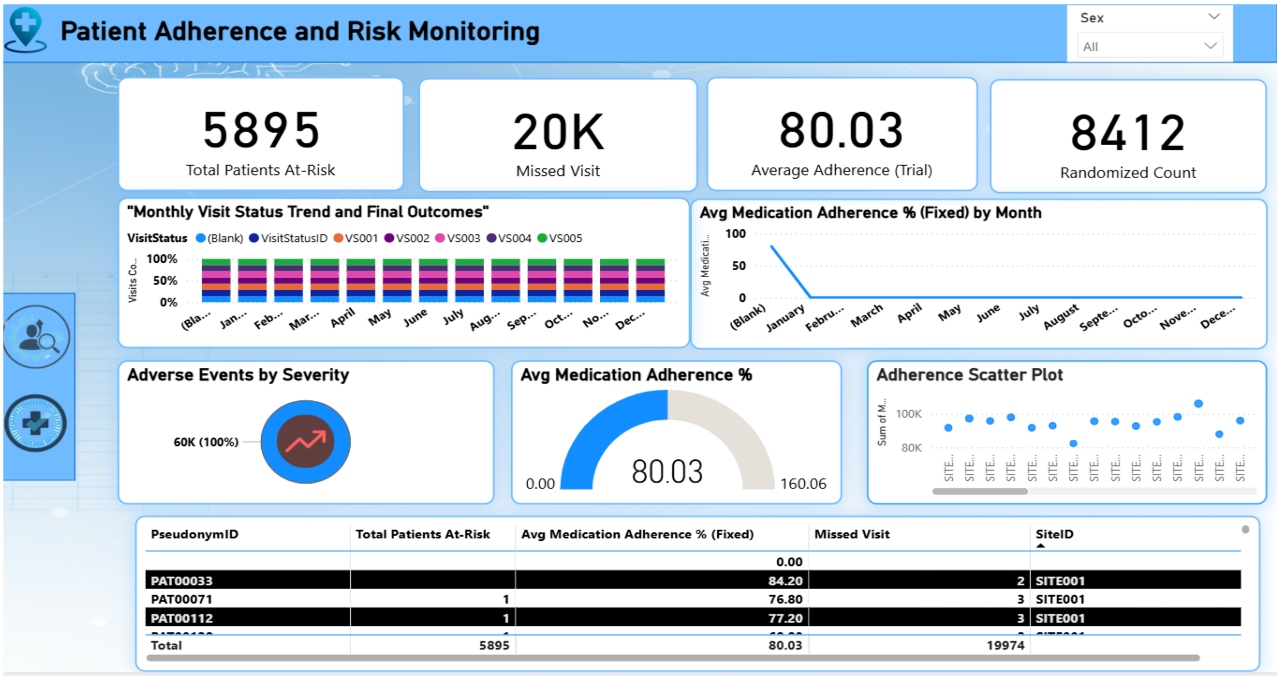
12.00K Total Screened

9573 Screen Pass

41.67 Cost Per Enrolled Patient

0.70 Enroll-to-Randomize Rate %

SiteName	Enrollemnet Count	Sum of DataCompletenessPct	Sum of Site_TargetEnrollment
Hospital_27	201	8,461.08	36
Hospital_7	207	8,611.20	35
Hospital_31	215	8,385.40	31
Total	12000	4,24,203.12	1243



Sex: All

5895 Total Patients At-Risk

20K Missed Visit

80.03 Average Adherence (Trial)

8412 Randomized Count

PseudonymID	Total Patients At-Risk	Avg Medication Adherence % (Fixed)	Missed Visit	SiteID
		0.00		
PAT00033		84.20	2	SITE001
PAT00071	1	76.80	3	SITE001
PAT00112	1	77.20	3	SITE001
Total	5895	80.03	19974	

7. EDA

- The exploratory data analysis of the clinical trial dataset revealed clear patterns in patient demographics, adherence scores, and adverse events.
- The data was largely clean with minimal missing values, ensuring reliability for further modeling and clinical interpretation. These insights support data-driven improvements in trial evaluation and patient outcome analysis.

8. Security & Compliance

- Data anonymization practices, PHI removal.
- Role-based access foe users/sites.
- Logging and audit trail for regulatory validation.

9. Testing and Deployment

- Unit/integration tests for backend and frontend endpoints.
- Docker containerization recommended; CI/CD, scheduled refresh for production.

10. Future Enhancements

- Integrate SMS/email alert automation for at-risk patients.
- Extend model explainability and mobile-friendly report views.

11. Conclusion

- Integrate SMS/email alert automation for at-risk patients.
- Extend model explainability and mobile-friendly report views.

PROJECT REPORT

Project Title: Public Sector Urban Development & Sentiment Analysis Project Report

Submitted by: Varun Panchal, Rajeshwari Acharya, Karanam Akhil, Nitisha Nagarkar

Company Name: Infotact Solutions

Date: 02/11/2025

1. Introduction

Urban municipal governments face challenges in responding effectively to citizen needs due to fragmented data and reactive management. This project delivers a comprehensive data analytics solution leveraging 311 service requests, public infrastructure data, and real-time social media sentiment analysis. It empowers city officials to monitor urban issues, assess service equity, and track public opinion, facilitating proactive, data-driven urban planning and service delivery.

2. Objectives of the Project

- Integrate multiple data sources for a unified view of urban service requests and public sentiment.
- Analyze spatial distribution and neighborhood-level equity in service delivery.
- Correlate social media sentiment with reported issues to gain real-time public feedback.
- Provide an interactive Power BI dashboard for city officials to make informed decisions.

3. Architecture & Technology

- **Data Sources:** 311 non-emergency service request logs, social media feeds, and geospatial shapefiles of city boundaries.
- **Data Preparation:** Power Query within Power BI for data transformation, including resolution time calculation, geospatial joins, and sentiment score extraction using Azure Cognitive Services Text Analytics.
- **Data Modeling:** Star schema model created in Power BI with fact and dimension tables for request types, neighborhoods, and aggregated sentiment data.
- **Visualization:** Power BI embedded visuals such as heat maps, choropleth maps, line charts, and word clouds. Azure Maps integrated for spatial analytics.
- **Security:** Role-level security (RLS) implemented in Power BI to control data access by user roles.

4. Key Features

- Real-time visualization of service request hotspots using geospatial heat maps.
- SLA compliance tracking with key metrics like average resolution time and percentage of requests met within 48 hours.
- Neighborhood comparison dashboards highlighting disparities in service delivery.
- Sentiment monitoring combining social media analysis and 311 request volume trends.

Interactive filters and slicers for dynamic exploration by request

type, time period, and geography.

5. Implementation Details

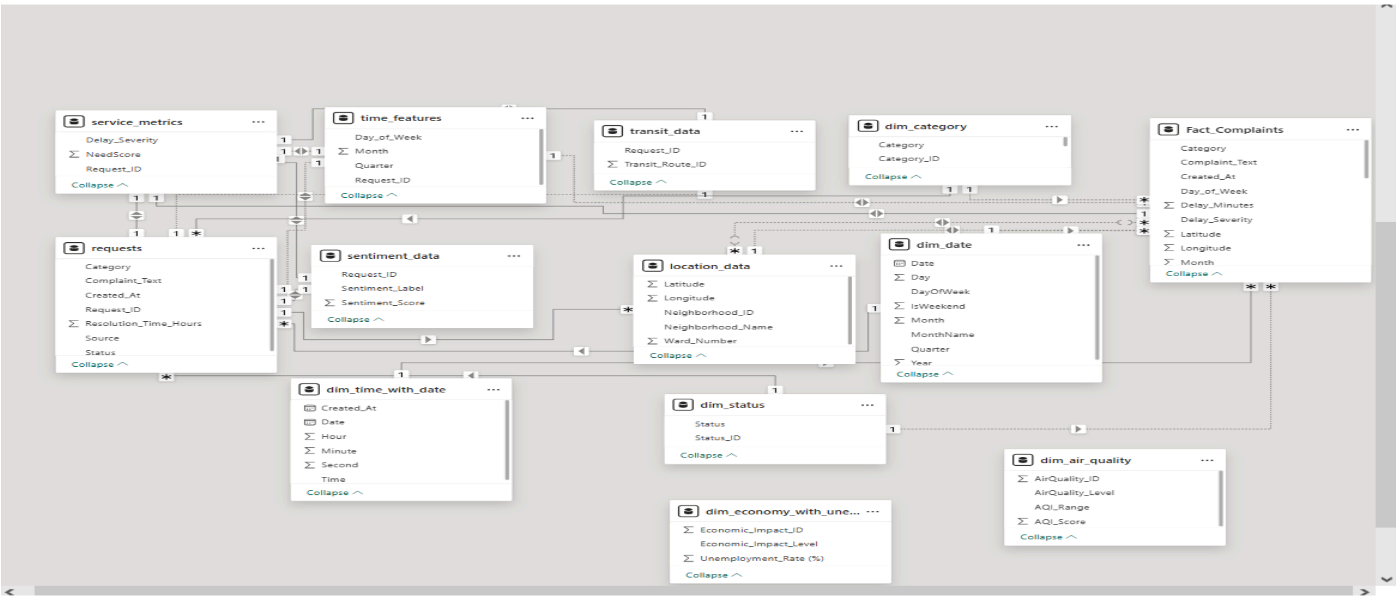
Phase 1: Data Preparation and Modeling

- Aggregated data from public sources stored in Power BI.
- Resolution Time calculated as difference between creation and completion timestamps.
- Raw social media text analyzed through Azure Cognitive Services to derive sentiment scores.
- Geospatial data joined with requests by latitude/longitude to assign neighborhoods where missing.
- Star schema created with `Fact_311_Requests`, `Dim_Request_Type`, and `Dim_Neighborhood` tables.

Phase 2: DAX Calculations and KPIs

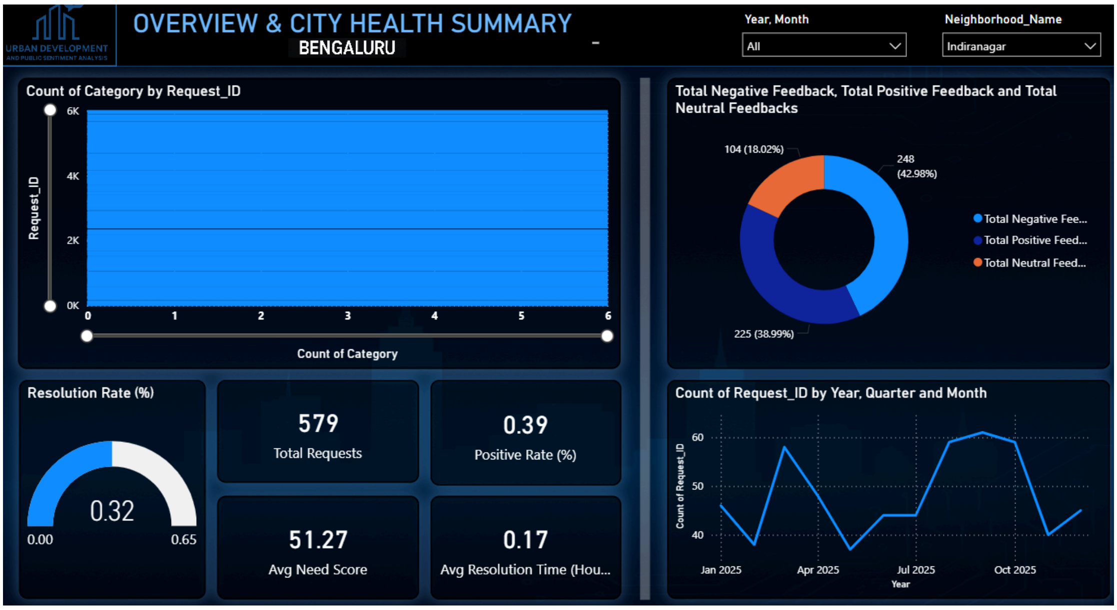
- Implemented average resolution times and SLA compliance percentage using DAX.
- Computed average sentiment scores and request volume to analyze trends.
- Measures written to allow filtering by request types and neighborhoods for granular analysis.

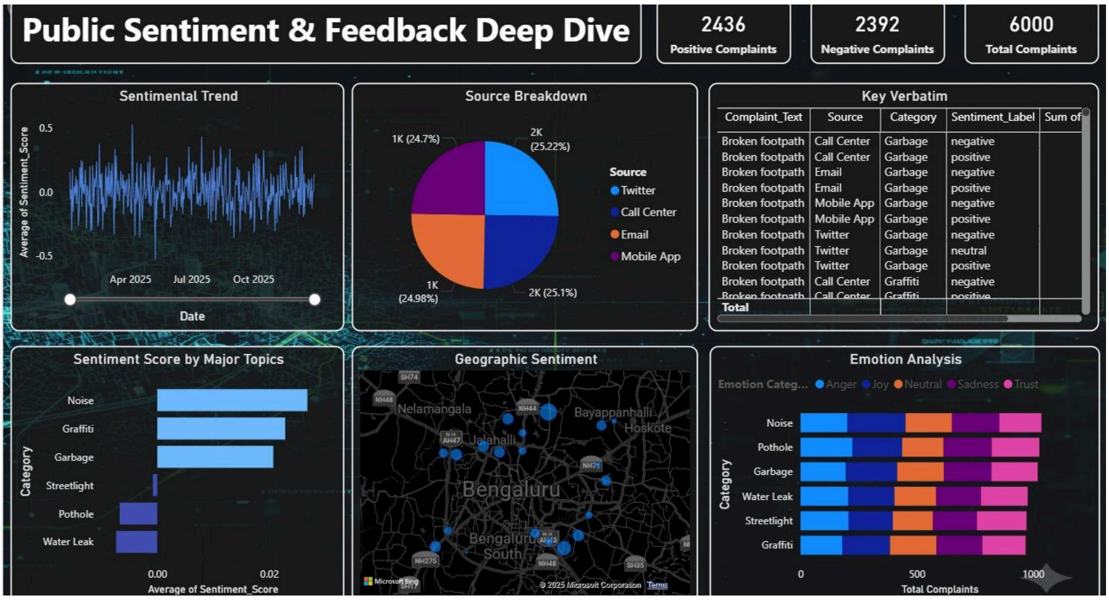
Data Model



6. Dashboard Demonstration

- **Service Request Heatmap:** Displays spatial density of service requests by category, enabling hotspot identification.
- **Neighborhood Equity Dashboard:** Choropleth map shading neighborhoods based on resolution times reveals service inequities.
- **Public Sentiment Monitor:** Dual-axis time series chart shows sentiment scores alongside request volumes; word cloud highlights prevalent complaint keywords.





Complaint_Text	Source	Category	Sentiment_Label	Sum of
Broken footpath	Call Center	Garbage	negative	
Broken footpath	Call Center	Garbage	positive	
Broken footpath	Email	Garbage	negative	
Broken footpath	Email	Garbage	positive	
Broken footpath	Mobile App	Garbage	negative	
Broken footpath	Mobile App	Garbage	positive	
Broken footpath	Twitter	Garbage	negative	
Broken footpath	Twitter	Garbage	neutral	
Broken footpath	Twitter	Garbage	positive	
Broken footpath	Call Center	Graffiti	negative	
Broken footpath	Call Center	Graffiti	positive	
Total				

7. Security & Compliance

- Power BI role-level security configurated ensuring sensitive data access aligned with user roles and responsibilities.
- Integration with Azure Active Directory for user authentication and data governance compliance.
- Data handling compliant with relevant public data privacy policies, anonymizing user-identifiable social media information where applicable.

8. Testing and Deployment

- Functional testing of data transformation logic in Power Query and DAX calculations for accuracy.
- User acceptance testing (UAT) with municipal officials to validate dashboard usability and insights.
- Deployed Power BI reports using Power BI Service with scheduled data refresh to maintain near real-time updates.
- Monitoring implemented for data pipeline failures and performance bottlenecks.

9. Future Enhancements

- Integrate additional data sources such as public transit data and emergency services for a more comprehensive urban analytics platform.
- Enhance social media sentiment analysis using multilingual models and topic modeling for deeper insights.
- Implement predictive analytics to forecast emerging service demand hotspots and resource allocation needs.
- Mobile-friendly dashboard versions for field officers and quick response teams.

10. Conclusion

This project offers a powerful analytics framework for municipal governance by unifying diverse data streams into actionable insights. Through data-driven visualization and SLA benchmarking, city officials are equipped to improve service delivery effectiveness and equity. Real-time public sentiment integration ensures citizen voices continually inform urban management. With deployment in Power BI and Azure technologies, the solution is scalable, secure, and poised for future expansion.