

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:

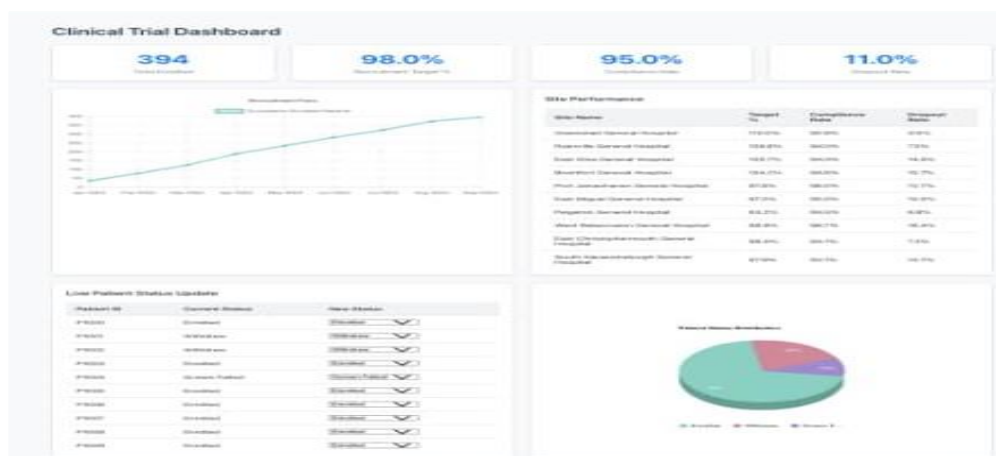
Tracks enrollment velocity and data quality per site.

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.

The screenshot shows the GitHub web interface for the repository 'Ramya-raaji / clinical-trial-dashboard'. The file 'App.js' is selected in the 'src' directory. The code is as follows:

```
1 import React, { useState, useEffect, useCallBack } from 'react';
2 import axios from 'axios';
3 import EnrollmentChart from './components/EnrollmentChart';
4 import PatientStatusUpdater from './components/PatientStatusUpdater';
5 import PatientStatusPieChart from './components/PatientStatusPieChart'; // <-- 1, IMPORT the new component
6
7 import './App.css';
8
9 const API_BASE_URL = 'http://127.0.0.1:8080';
10
11 > function App() {
12   //
13 }
14
15 export default App;
```

The screenshot shows the GitHub web interface for the repository 'Ramya-raaji / clinical-trial-dashboard', specifically the 'EnrollmentChart.js' file in the 'src/components' directory. The code is as follows:

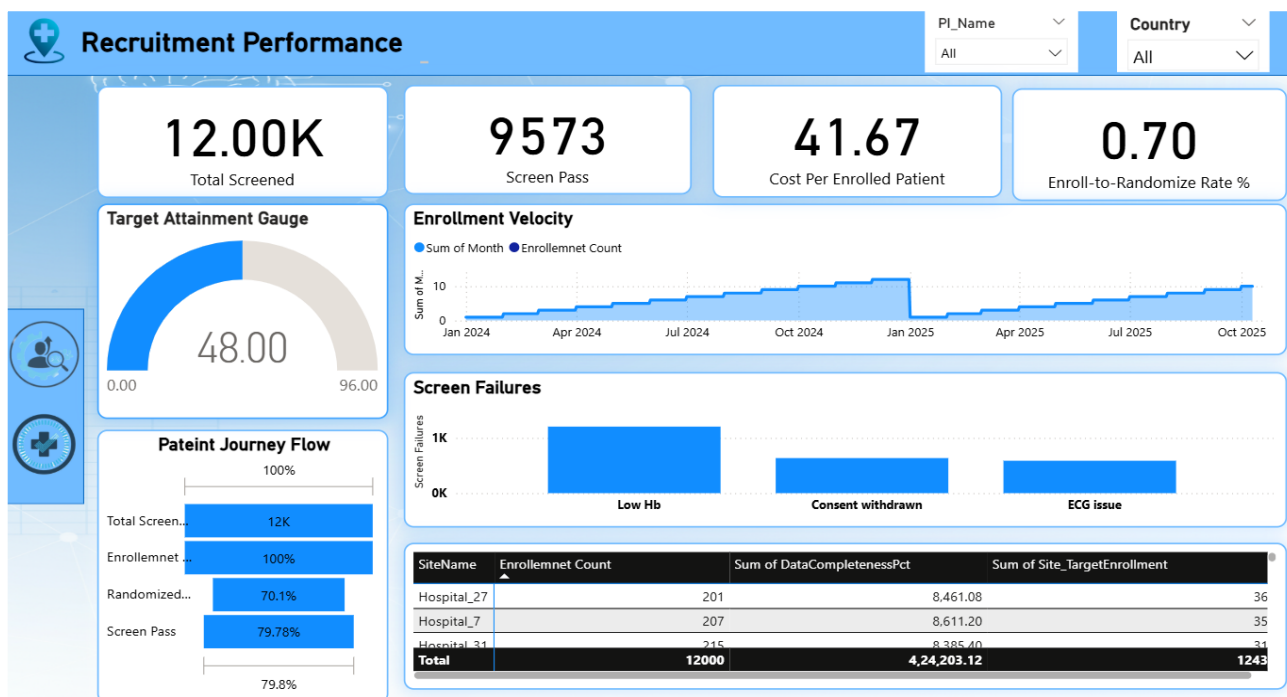
```
1 import { Line } from 'react-chartjs-2';
2 import { Chart as ChartJS, CategoryScale, LinearScale, PointElement, LineElement, Title, Tooltip, Legend } from 'chart.js';
3
4 ChartJS.register(CategoryScale, LinearScale, PointElement, LineElement, Title, Tooltip, Legend);
5
6 > const EnrollmentChart = ({ data }) => {
7   const chartData = {
8     labels: data.map(d => new Date(d.date).toLocaleDateString('en-US', { month: 'short', year: 'numeric' })),
9     datasets: [
10       {
11         label: 'Cumulative Enrolled Patients',
12         data: data.map(d => d.count),
13         borderColor: 'rgb(75, 192, 192)',
14         tension: 0.1,
15       },
16     ],
17   };
18
19   const options = { responsive: true, plugins: { title: { display: true, text: 'Recruitment Pace' } } };
20
21   return <Line options={options} data={chartData} />;
22 }
23
24 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.



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 for 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.