JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

A GIS AND IOT-ENABLED DECISION SUPPORT FRAMEWORK FOR PAYING GUEST ACCOMMODATION MANAGEMENT: A CASE STUDY

[1]Kanan Jethwani, [2]Anil A, [3]Jagadeesh

[1]Department of Computer Science and Engineering, [2]Department of Artificial Intelligence and Machine Learning, [3]Department of Artificial Intelligence and Machine Learning
[123]R.V. College of Engineering, Bengaluru, India
[1]kananjethwani.cs24@rvce.edu.in, [2]anila.ai24@rvce.edu.in, [3]jagadeeshav.ai24@rvce.edu.in

Abstract : Paying Guest (PG) accommodation management mostly depends upon fragmented record keeping leading to inefficient occupancy tracking, lead management and operational decision making. This paper presents PGShaala, a GIS and IoT-enabled decision-support framework implemented using React.js, Node.js, Express.js, MongoDB and Supabase. Leaflet-based geospatial visualization, OSRM-based route optimization, Role-Based Access Control (RBAC), occupancy analytics, CRM workflows and an intelligent property matching engine are integrated for this system within a layered client-server architecture. Functional evaluation observations show reliable authentication, accurate occupancy tracking, successful route generation and real time dashboard updates which highlight the effectiveness of framework in improving operational efficiency and supporting smart accommodation management.

IndexTerms - Property Management, GIS, Route Optimization, IoT Monitoring, CRM

I. INTRODUCTION

The rapid growth in relocation due to studies or job has given rise to a demand for organised accommodation management systems. Most working professionals as well as students go for Paying Guest (PG) services due to accessibility and budget. As of now there are no such platforms which can reduce paperwork and communication gap between consumer and seller which result in delayed decision making, not optimised usage of resources and poor visibility of occupancy trends. Research work on digital hostel management systems has emphasized the importance of centralization of records, automation of tasks, and real-time monitoring in increasing efficiency [1], [2]. In a similar vein, research on smart dormitory management systems utilizing IoT technology has brought to light some major benefits in the area of efficient resource utilization. Although the existing platforms are generating revenue but none is yet specific for PGs and are based on basic functions like room allocation, customer registration and fee management. What they lack is integrated geospatial visualization, route optimization capabilities, occupancy analytics, lead management workflows, real-time infrastructure monitoring, challenges in monitoring multiple properties simultaneously and analyzing occupancy performance. PGShaala emphasizes the growing importance of intelligent management systems, decision-support frameworks and digital transformation within accommodation and institutional management environments [3]. In PGShaala we combined GIS technologies, occupancy analytics, IoT monitoring, and operational management within a single platform. This system overcomes the limitations of existing management software and contribute towards the growth of smart living infrastructure. The further organization of the paper is mentioned below. Section II reviews the related literature. Section III shows the design and implementation of the PGShaala framework. Section IV describes the developed system interface. Section V discusses the experimental results and system performance. Finally, Section VI concludes the paper.

II. LITERATURE REVIEW

Managing PGs and rental accommodations for students and professionals can be quite challenging. Currently, these PGs are mostly managed through manual methods like paperwork, phone calls, and WhatsApp messages, which haven't really taken advantage of advanced technologies such as data analytics, IoT, or digital business tools [2]–[4], [6]. Most popular online booking sites focus mainly on hospitality and short-term rentals, leaving very few tailored solutions for the PG sector. That's why we decided to explore how smart city technologies, data analytics, and digital transformation can help improve the management of PGs and rental accommodations [1], [2], [6]. Nikpour's research highlights how IoT can be applied to utility management, showing that real-time monitoring and smart resource management can boost operational efficiency and cut down utility costs [7]. However, like other smart city and IoT studies, this research has mainly focused on large organizations, smart buildings, and commercial facilities—not accommodation services [1], [4], [7]. These studies often overlook resident-focused features like managing residents, monitoring room occupancy, tenant support, and day-to-day accommodation operations, which are all crucial for the PG sector [7]. Building on smart city analytics, IoT monitoring, and digital transformation research, we have adapted these

ideas specifically for PG management [1], [2], [4], [6], [7]. To manage PGs effectively, strong data analytics are essential to track occupancy, analyse resource use, support operational decisions, and optimise services [2], [3], [14]. From all these papers, we found that there is still a gap in PG management. Most of the work is about smart buildings or rental websites, but not about the daily problems in PGs. Very few systems bring map features, IoT, room details, and property management together in one place. Because of this, PG owners still have to use different methods to manage their work. This research tries to fill that gap by making one simple system that helps manage PGs in an easier way.

III. SYSTEM DESIGN AND IMPLEMENTATION

This section describes the design and implementation of the proposed PGShaala framework. It presents the problem definition, development methodology, system architecture, and core algorithms that support efficient property management, occupancy tracking, GIS-based visualization, and operational decision-making.

A. Problem Definition

The existing PG management systems mostly depend on manual record keeping and disintegrated tools leading to inefficient tenant management, poor occupancy tracking and limited operational visibility. Also it is difficult for property managers to improve decision making and enhance tenant experience with the lack of geospatial intelligence, real time monitoring and analytical insights.

B. Methodology

PGShaala is developed based on the user centric methodology which is inspired by the latest studies about digital transformation, smart facility management and information system engineering [6], [8]. Tenant registrations, room allocation and fee management are the only features in common accommodation management systems. PGShaala acknowledges and overcomes these limitations by integrating property management, occupancy tracking, CRM workflows, geospatial intelligence, route optimization, analysis and IoT-based monitoring, all within one complete platform.

1) Requirement Analysis: We held a survey with PG own-

ers, their managers, working professionals and students applying for PGs and finalised the requirements for this system. The survey results showed regular challenges disorganised property records, inefficient occupancy tracking, inconsistent lead management, property performance visibility issues and the absence of one centralized operational analytics. We grouped the requirements into five domains with the help of survey observations:

- Property Management
- Occupancy Management
- Lead and CRM Management
- Geospatial Visualization and Routing
- Monitoring and Analytics

2) System Development: Scalability, maintainability and

preference based efficient accommodation management are the key features of PGShaala. It follows a modular software engineering approach. We conducted multiple rounds of functional testing and integration testing throughout development to check with occupancy calculations, route generation, user permissions, CRM workflows and dashboard analytics [10].

3) System Constraints: Hard Constraints:

- Unauthorized access to tenant or property information is strictly prohibited.
- Authentication must be secure and access control based on roles.
- Occupancy and property records need to remain consistent across all modules.
- Route generation and geospatial visualization should be precise. Soft Constraints:
- Entering data and retrieving records takes minimal effort from users.
- Dashboards should be intuitive for users to navigate.
- Queries related to occupancy and analytics ought to have low response times.

C. System Architecture

PGShaala has a client-server architecture of four layers: Presentation Layer, Application Layer, Data Management Layer, and Intelligence Layer. This layered architecture ensures separation of concerns, enabling independent development, maintenance, and scalability of each component [5], [11]. Fig. 1 illustrates the overall system architecture.

1) Presentation Layer: The frontend is made using Re-

act.js, TypeScript, and Tailwind CSS [17]. It provides rolebased dashboards for administrators, property managers, and staff members. The main functions of the interface are property management, occupancy tracking, lead management, analytics visualization, and GIS-based property mapping.

2) Application Layer: The backend uses Node.js and Ex-

press.js. It emphasizes authentication, property management, occupancy tracking, CRM workflows, route generation, analytics processing, and monitoring services. RESTful APIs are used for communication between frontend and backend services.

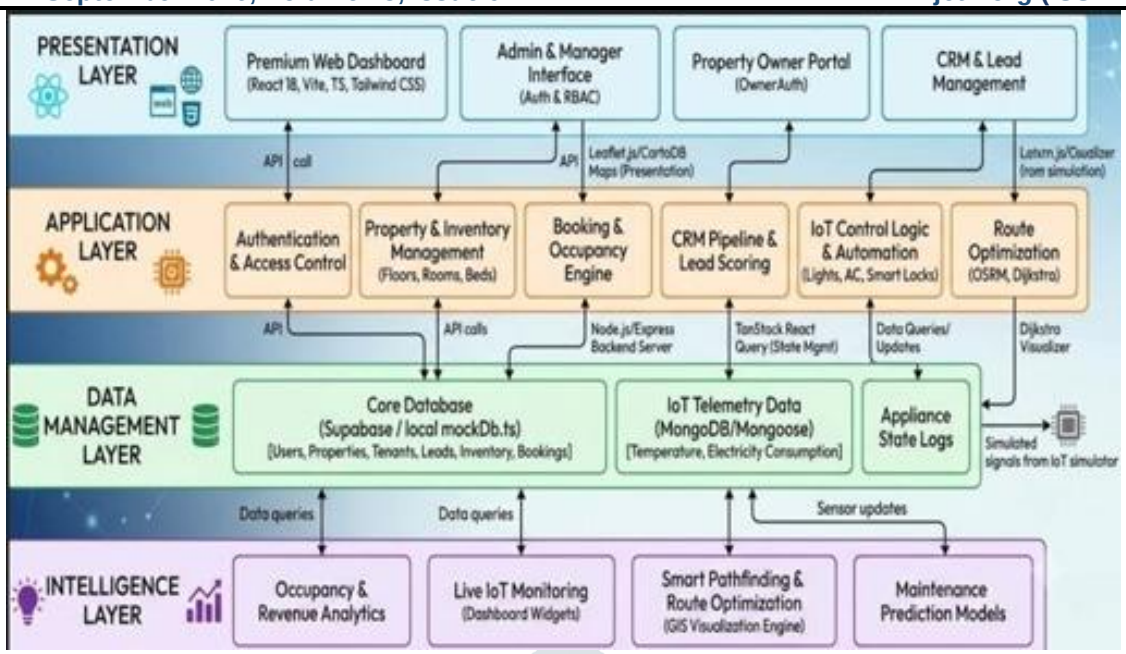

Fig. 1. PGShaala four-layer system architecture overview layout.

3) Data Management Layer: We used MongoDB for

database due to easy accessibility and efficiency. The stored data include users, properties, rooms, tenants, leads, inventory, maintenance records and sensor data [18].

4) Intelligence Layer: The Intelligence Layer is the main

computational core of PGShaala. This layer consists of three major components:

- Lead Text Parser: The function of this module is to extract information in a structured format from an unstructured format like text messages, lead name, phone number, budget and location preferences.
- Property Matching Engine: This engine utilizes a Remote Procedure Call (RPC) function to execute a join query between the beds inventory and property tables based on the selected lead criteria.
- GIS Routing Engine: Driving distances and travel times are computed with the help of this module using opensource Open Source Routing Machine (OSRM) API within Bengaluru.

TABLE I. PGSHAALA TECHNOLOGY STACK

Component	Technology	Purpose
Frontend	React.js	User Interface
Styling	Tailwind CSS	Responsive Design
Backend	Node.js, Express.js	Business Logic
Database	MongoDB & Supabase	Data Management
GIS Visualization	Leaflet	Property Mapping
Routing Engine	OSRM	Route Optimization
Authentication	JWT, bcrypt	Secure Access Control
Analytics	Recharts / Chart.js	Dashboard Visualization

5) The Lifecycle of a Request: The user actions are securely

processed across the application ecosystem which is governed by request lifecycle within PGShaala.

- Authentication and Routing Interception: A custom SupabaseAuthProvider is used to evaluate active session tokens, checks user permissions in the user roles repository and render the corresponding view dynamically.
- Data Fetching and Local Caching: The system uses React Query to manage and limit data interactions. State management which means allowing system to do different things while waiting for data from database is used. The system data interactions and React Query work together to minimize database calls.
- Real-Time Matching Execution: During the lead matching, the system calculates a real time matching score for the active accommodation listings and returns the most relevant results.

D. Algorithm Design

1) Role-Based Access Control (RBAC): Role-Based Access

Control (RBAC) [15] within PGShaala grants various operational privileges to different roles divided into four categories managed in the Supabase user roles repository:

- Admin: Who possesses unrestricted access system wide across all views.
- Manager: Can grant access to Customer Relationship Management (CRM) operations and the AI-driven Property Matching Engine.
- Agent: Is given authorization to interact with sales modules including lead registries, Kanban tracking boards, site visits, and booking finalizations.
- Owner: Restricted to the respective Owner Portal.

ProtectedRoute and OwnerProtectedRoute handles authorization and access verification dynamically during route traversals. React wrapper components validate the active session state and user privileges before rendering protected resources.

TABLE II. ROLE-BASED FEATURE ACCESS IN PGSHAALA

Feature	Staff	Manager	Admin
View Property Data	Yes	Yes	Yes
Manage Tenants	Limited	Yes	Yes
Manage Leads	Yes	Yes	Yes
Occupancy Analytics	No	Yes	Yes
User Management	No	No	Yes
System Configuration	No	No	Yes

2) Property Owner Authentication and Verification Flow:

(/owner-login) routing algorithm used for accessing the owner portal. Supabase Authentication service handles primary identity verification after credential submission. It follows standardized authorization principles [12] and a secondary verification pipeline:

- UID Record Validation: The backend verifies if a match-
  ing entry exists within the list of owners corresponding to the authenticated user identifier (UID).
- Automated Session Revocation: An automated logout
  invoke is initiated if the lookup fails to identify the user as a registered property owner.
- Exception Handling: The user interface terminates the
  login sequence and renders a notification instructing the user to contact the PGShaala technical support team. This architectural isolation ensures strict privilege separation.

3) Occupancy Classification Algorithm: PGShaala uses

occupancy percentage to track property usage and provide insights about efficiency in property utilisation.

$$\text{Occupancy Rate (\%)} = (\text{Occupied Rooms} / \text{Total Rooms}) \times 100 \quad (1)$$

TABLE III. OCCUPANCY LEVEL CLASSIFICATION

Level	Occupancy Rate
Low	< 50%
Medium	50% - 80%
High	> 80%

4) Route Optimization Algorithm: For property inspections,

maintenance visits and customer site visits efficient route plan is required. The transportation network is modelled by the routing module as a weighted graph G = (V, E), where:

- V represents accommodation locations.
- E represents road connections.

Shortest-path computation is performed using graph traversal techniques based on Dijkstra's algorithm [11], [16], together with practical route-planning optimization techniques used in modern routing systems [13]. The optimization objective is

$$\text{dist}(v) = \min_{(u} \in V_{)} (\text{dist}(u) + w(u, v)) \quad (2)$$

where dist(v) is the minimum distance to node v, and w(u, v) is the travel cost between nodes. The routing engine minimizes overall travel distance and improves operational efficiency.

E. Lead Conversion Workflow

The CRM module manages prospective tenants through a structured conversion pipeline. The lead lifecycle consists of: New Lead → Contacted → Interested → Site Visit → Converted Progress of each stage is recorded in the database and used to evaluate revenue and market demand. Lead Conversion Rate is calculated as:

$$\text{Conversion Rate (\%)} = (\text{Converted Leads} / \text{Total Leads}) \times 100 \quad (3)$$

This workflow allows managers to identify bottlenecks in the acquisition process and improve tenant conversion efficiency.

F. Search Query Processing

The search module accepts a query string Q and user role R. Requests are then sent to the backend through REST APIs. Based on role permissions, MongoDB dynamically filters accessible records and returns the result set. Staff members can look at the operational data given to them. Managers, on the other hand, have access to information specific to each property. Meanwhile, system-wide records are available to administrators. The JSON format is sent back as the result, within dashboard they appear dynamically.

IV. SYSTEM INTERFACE AND UI SCREENSHOTS

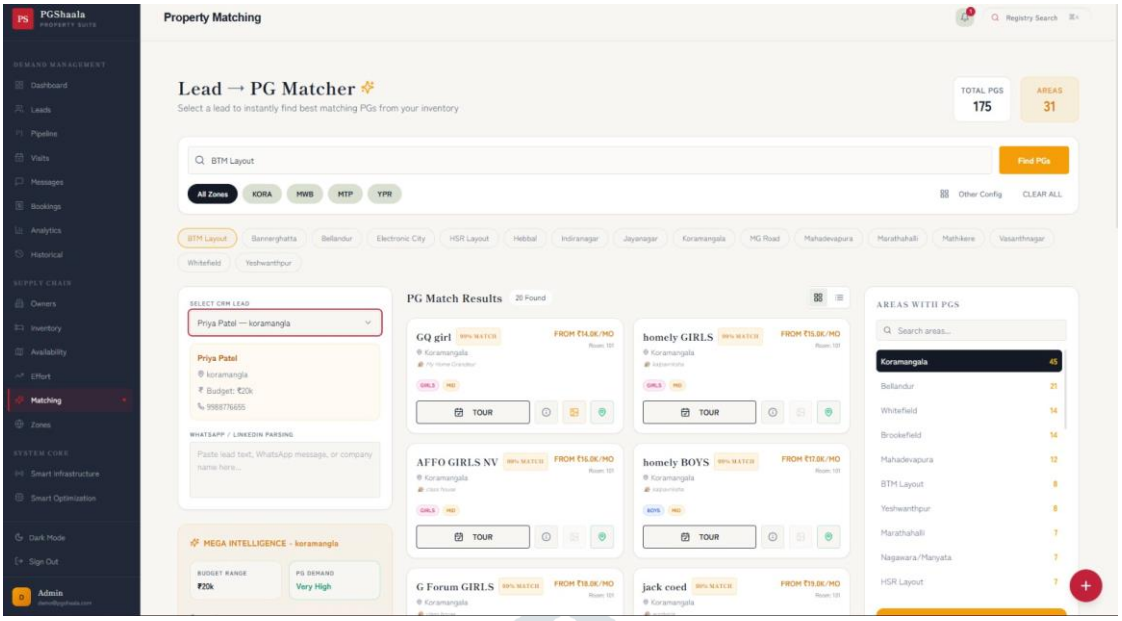

Fig. 2. Lead PG Matching execution pipeline view.

Matching refers to the "Lead PG Matcher" engine, whose main purpose is to find suitable PG properties for a given lead. Any active CRM lead selected from the drop-down automatically triggers a smart matching algorithm that scores and ranks properties based on location, budget, and availability. The match results display each PG's match score, rent, photos, and Google Maps link. Fig. 3 shows Inventory OS: Master property list of all PG properties in the database. Each property card shows the Name, Address, Zone, pricing (T/D levels), Property Manager, and Room inventory. Each property has quick-action

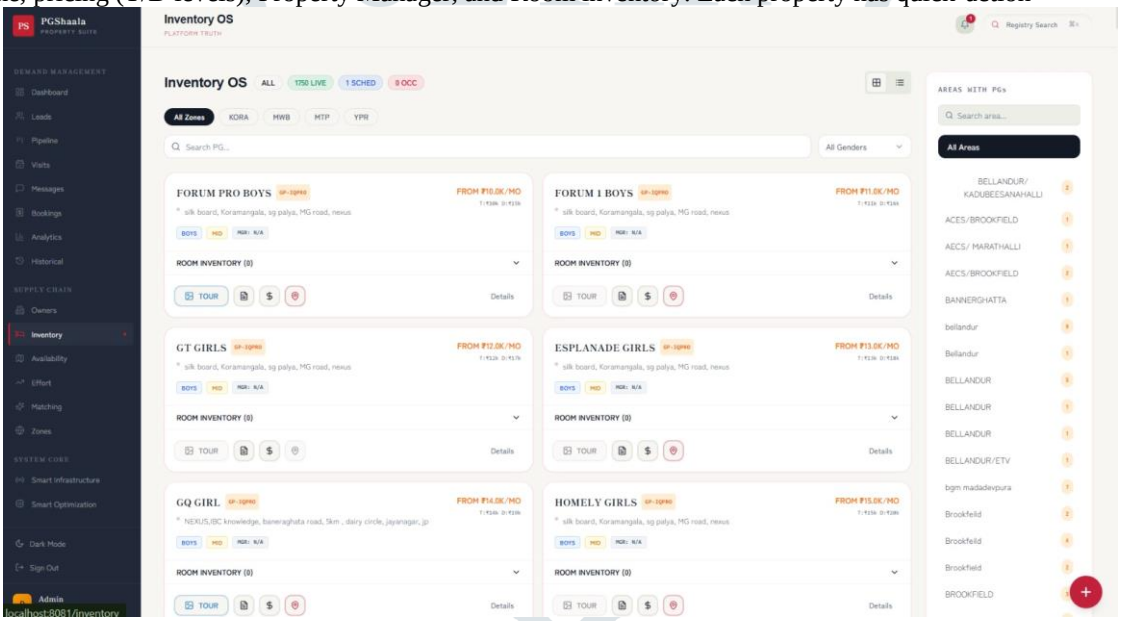

Fig. 3. Inventory OS allocation management structure panels.

buttons to open a virtual tour, view documents, check pricing, or open the property in Google Maps. Pipeline (as shown in

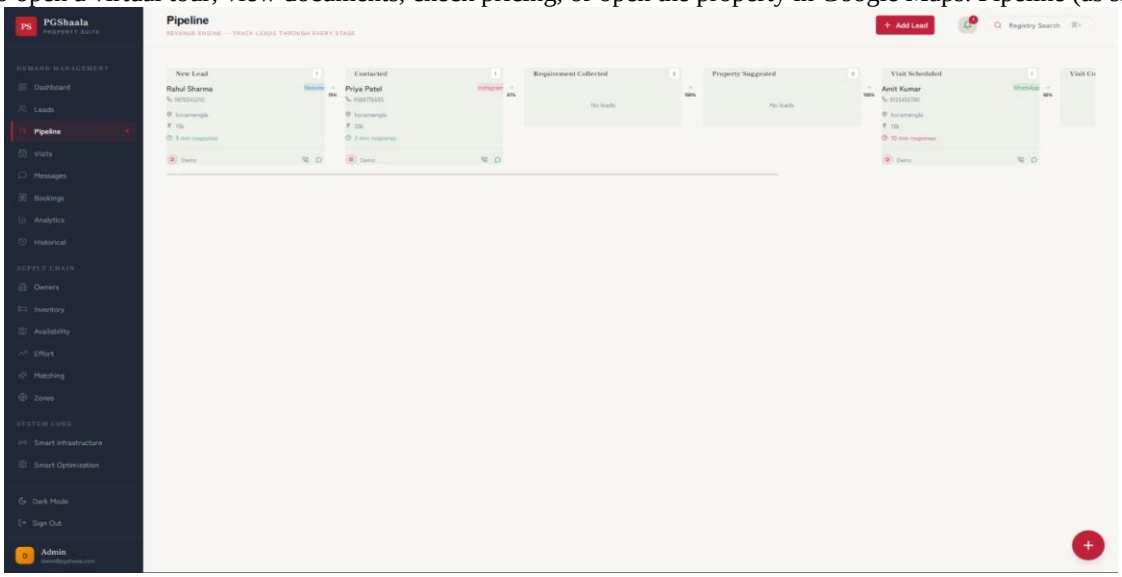

Fig. 4. Kanban board layout pipeline for lead lifecycle tracing tracking handles.

Figure 4) The workflow is a Kanban-style leading management dashboard; the lead management tool displays all leads for the configured pipeline (New Contacted Visited Booked Lost). The main intelligence center is Analytics (Fig. 5), which

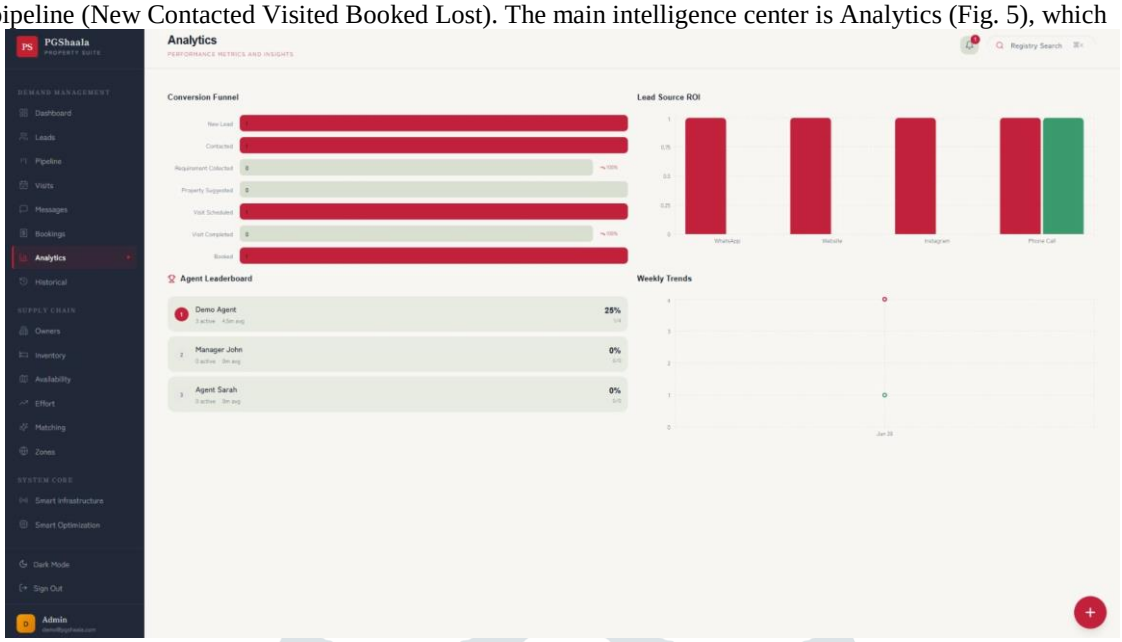

Fig. 5. Analytics engine chart data visualization control center command dashboard.

provides charts and reports:

- Conversion Funnel – This chart shows how many leads are currently present for each stage of the pipeline.
- Lead Source ROI – This chart depicts leads against number of bookings across all lead sources.
- Agent Leaderboard – Sales agents will be compared based on lead to customer conversion and assigned a rank, while also being displayed to show number of active leads as well as the number of booked deals that individual has closed.

TABLE IV. SYSTEM PERFORMANCE SUMMARY

Metric	Result
Authentication Success Rate	100%
RBAC Enforcement Accuracy	100%
Property Data Consistency	99.2%
Occupancy Classification Accuracy	98.7%
Route Generation Success Rate	100%
Dashboard Update Accuracy	99.4%

- Weekly Trends – This chart visualizes lead generation and the number of booked deals generated in the previous eight weeks.

V. RESULTS AND DISCUSSION

The prototype of PGShaala has been developed as a full stack web application with React in frontend, NodeJs as backend and MongoDB as the database. All modules were developed and tested on a standard development machine: Intel Core i5 processor, 8 GB RAM, Windows 11. These included property management, occupancy tracking, CRM workflow, GIS visualization, route optimization, analytics dashboard and authentication. Evaluation metrics were established that allow the measurement of operational effectiveness and reliability of the system, as suggested by some recent research work on IS and smart infrastructure platforms [1], [10] considering aspects of reliability of access control, data consistencies, correct execution of workflow, and operational efficiency in IS or platform evaluation, we conducted functional testing of the PGShaala module and the workflow.

A. Functional Testing Results

Role Based Access control(RBAC) was successfully implemented to reduce the unauthorized access to prevent any misuse of resources. Property management, modification, occupancy updates and tenant assignments were executed without any inconsistency. In-process lead movement was tracked using CRM module, occupancy segmentation was handled by analytics engine, and navigation path through use of geospatial data and was dynamically represented on dashboards.

B. Before vs. After PGShaala: Operational Impact

The current workflows are based on spreadsheets, papers, and offline manual modes of communication. This translates to manual-intensive workflows, inconsistency in processes, and decisions taking longer than optimal time to be made. PGShaala centralises the entire operational flow in a single portal thus increasing transparency and the workflow efficiency significantly.

C. Administrative Module

Administrator module provide full control of all accommodation units from a single access point. Admins are capable to maintain users, property data, occupancy, operational records. Analytical Dashboards is a tool where charts and reports generated from the operational data, come from any other

TABLE V. BEFORE VS. AFTER PGSHAALA - OPERATIONAL IMPACT

Metric	Traditional Method	PGShaala
Occupancy Tracking	Manual Updates	Real-Time Dashboard
Property Search	Manual Records	GIS Visualization
Lead Management	Phone/Spreadsheet Based	CRM Pipeline
Route Planning	Manual Navigation	Automated Routing

Data Reporting Manual Compilation Instant Analytics
Property Monitoring Periodic Inspection Centralized Monitoring

modules. The main goal of analytics dashboard is to enable admins to see occupancy trend, asses a property marketability, understand leads conversion and find operations bottlenecks.

D. Property Management Module

This is useful for efficient accommodation property management for facility providers as property managers could register properties, update their room information, allocate rooms to people or groups, manage the available space within it and keep an account of the occupied levels. As part of this, we have made sure in the tests we carried out that every room updating will instantly be reflect to the portals and be shown directly in dynamic mode so we can easily eliminate conflict of view with the administrator, manger and owner.

E. CRM and Lead Management Module

Track your potential tenants through every step. Your CRM records each of your leads from inquiry all the way down the sales pipeline. We provide you with one place to keep up with your leads to ensure nobody falls through the cracks and give visibility to your conversion rates.

F. GIS and Route Optimization Module

The GIS module uses location based mapping to see accommodation facilities. Property locations, occupancy status, and operational information are displayed through a unified geospatial interface. The route optimization module generates an efficient path between accommodation facilities for travelling to make decisions based on location preferences. The functionality is also invaluable for site inspections, appointment setting and customer visits.

G. Analytics and Monitoring Module

The analytics engine constantly processes operational data to generate insights related to occupancy utilization, lead conversion performance, and property activity. Dashboard enables real-time visibility to managers regarding operational performance.

VI. CONCLUSION

In this paper, PGShaala was presented as a platform of centralized management of accommodation able to solve part of the problems caused by property management, occupancy tracking, tenant and leads management, and operational analysis. During its implementation and testing, the functionality of PGShaala was proven useful in aspects such as authentication, property management, occupancy rate tracking, data analysis, and geo-location making it more effective than doing it in a manual way. The uniqueness of PGShaala with regard to other property management software are the combination of accommodation management, advanced analytics, and geospatial intelligence. However, although the system implemented already works fine to fulfill simple requirements, there is room for improvement of the system. Artificial Intelligence techniques for predicting occupancy, use of Internet of Things technology for tracking and monitoring, hosting on cloud services and business intelligence are examples of possible new features to be implemented in PGShaala in the future. All in all, PGShaala is proof of concept that a centralized, analytics supported system can successfully be used to manage property accommodations.

REFERENCES

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of Things for Smart Cities,” IEEE Internet of Things Journal, vol. 1, no. 1, pp. 22–32, 2014.
[2] M. Batty, “Smart Cities, Big Data and City Analytics,” Regional Studies, Regional Science, vol. 1, no. 1, pp. 267–279, 2014.
[3] H. Xu, F. Omitaomu, S. Sabri, S. Zlatanova, X. Li, and Y. Song, “Leveraging Generative AI for Urban Digital Twins: A Scoping Review on the Autonomous Generation of Urban Data, Scenarios, Designs, and 3D City Models for Smart City Advancement,” arXiv:2405.19464, 2024.
[4] S. Madakam, R. Ramaswamy, and S. Tripathi, “Internet of Things (IoT): A Literature Review,” Journal of Computer and Communications, vol. 3, no. 5, pp. 164–173, 2015.
[5] S. Mazzetto, A. Al-Sehrawy, and F. I. Khan, “A Review of Urban Digital Twins: Integration, Challenges, and Future Directions,” Sustainability, vol. 16, no. 19, Art. no. 8337, 2024.
[6] M. Attaran, “The Impact of Digital Technology on Business Transformation,” Journal of Strategic Innovation and Sustainability, vol. 12, no. 1, pp. 16–27, 2017.
[7] M. Nikpour, A. Shamshirband, and H. Fathi, “Internet of Things for Smart Buildings: A Review of Intelligent Monitoring and Energy Management Systems,” Sustainable Cities and Society, vol. 91, 104423, 2023.
[8] OECD, “Smart City Data Governance,” OECD Publishing, Paris, France, 2023.
[9] B. Kitchenham and S. Charters, “Guidelines for Performing Systematic Literature Reviews in Software Engineering,” EBSE Technical Report, Keele University and Durham University, 2007.
[10] M. Fowler, Patterns of Enterprise Application Architecture. Boston, MA, USA: Addison-Wesley, 2002.
[11] E. W. Dijkstra, “A Note on Two Problems in Connexion with Graphs,” Numerische Mathematik, vol. 1, pp. 269–271, 1959.
[12] D. Hardt, “The OAuth 2.0 Authorization Framework,” IETF RFC 6749, 2012.
[13] P. Sanders and D. Schultes, “Engineering Fast Route Planning Algorithms,” Lecture Notes in Computer Science, vol. 5515, pp. 23–36, 2009.
[14] A. Al-Sehrawy, M. Kumar, and M. Watson, “Comprehensive Analysis of Digital Twins in Smart Cities,” Artificial Intelligence Review, vol. 57, 2024.
[15] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, “RoleBased Access Control Models,” IEEE Computer, vol. 29, no. 2, pp. 38–47, 1996.
[16] I. Sommerville, Software Engineering, 10th ed. Boston, MA, USA: Pearson, 2015.
[17] React Documentation Team, “React Documentation,” Meta Open Source, 2025.
[18] MongoDB Inc., “MongoDB Documentation,” MongoDB, 2025.