

Civic Issue Reporter System

Ritesh kumar¹, Nikhil Nagarkoti²

¹ Assistant Professor Department of Artificial Intelligence & Data Science, Dr. Akhilesh Das Gupta Institute of Professional Studies, Delhi, India.

² Scholar of B. Tech 4th Year Department of Artificial Intelligence & Data Science, Dr. Akhilesh Das Gupta Institute of Professional Studies, Delhi, India.

riteshchandel@gmail.com

nikhilsinghnagarkoti@gmail.com

ABSTRACT

Urban areas face increasing challenges in managing civic issues such as potholes, waste accumulation, and infrastructure problems. Traditional complaint systems are often inefficient and lack transparency, leading to delays in resolution. The proposed Civic Issue Reporter System is a web-based application designed to streamline the process of reporting, tracking, and resolving civic issues.

The system allows users to submit complaints with location details and media, while administrators can manage and resolve issues through a dedicated dashboard. Real-time notifications and proof-based updates improve transparency and communication between citizens and authorities. The application is developed using React, Node.js and Supabase, ensuring efficient data handling and scalability.

This system demonstrates how modern web technologies can be used to build an effective and user-friendly solution for improving civic issue management.

Abbreviations

PWA: Progressive Web Application API: Application Programming Interface UI: User Interface

UX: User Experience DB: Database

GPS: Global Positioning System HTTP: Hyper Text Transfer Protocol JSON: JavaScript Object Notation CRUD: Create, Read, Update, Delete

1. INTRODUCTION

In the modern urban environment, efficient management of civic issues has become essential for ensuring high-quality public infrastructure and smooth administrative functioning. Municipal bodies deal with large volumes of complaints every day, including pothole reports, waste accumulation, water leakage, streetlight failures, and infrastructure damage. When these tasks are handled manually, the process becomes time-consuming, error-prone, and difficult to track, leading to delays in resolution, lack of accountability, and reduced trust in local governance.

The Civic Issue Reporter System developed using React, Node.js, and Supabase is designed to overcome these challenges by providing an integrated, reliable, and user-friendly digital solution. The system streamlines core civic operations such as complaint registration, location tracking, admin management, status updates, and feedback collection.

The proposed system follows a modular approach, combining a React-based frontend interface with secure and optimized Supabase database operations. Node.js ensures seamless communication between the application and database, enabling real-time data retrieval and updates. Furthermore, features like role-based login, media-based complaint submission, and automated notifications help improve operational efficiency and reduce manual responsibilities.

By digitalizing civic issue management, this system aims to enhance citizen experience, ensure faster issue resolution, and provide a scalable foundation for future smart city automation.

2. LITERATURE REVIEW

Efficient management of civic issues has been the subject of numerous studies due to the increasing demand for digital urban governance systems. Traditional manual methods of receiving complaints, tracking field operations, and updating resolution status often result in delays, human errors, and lack of transparency. Researchers have highlighted the importance of digitizing these processes through integrated Civic Issue Management Systems to ensure better workflow, accountability, and citizen satisfaction.

- Studies on civic complaint platforms show that digital systems significantly reduce response times and improve citizen engagement by centralizing issue data and enabling real-time tracking.
- Research on React-based applications demonstrates that component-driven frontend architecture combined with REST APIs offers strong reliability and smooth user interaction for citizen-facing platforms.
- Supabase is widely used in modern web applications due to its open-source nature, built-in authentication, real-time database capabilities, and seamless media storage support.
- Implementation of geo-tagged complaint submission and location-based filtering reduces administrative confusion and improves assignment of issues to the correct field teams, leading to faster resolution.
- Role-based access control and modular system design have been shown to strengthen data security and prevent unauthorized access to complaint records and administrative operations.

2.1 Objectives

The primary objective of the Civic Issue Reporter System developed using React, Node.js, and Supabase is to digitalize and streamline civic issue management through a secure, efficient, and user-friendly web-based solution. The system is designed to automate various reporting and resolution functions, reducing manual workload and enhancing service quality between citizens and urban authorities.

3. METHODOLOGY

The architecture of the Civic Issue Reporter System follows a modular and layered approach designed to ensure reliability, scalability, and smooth integration between user interfaces, application logic, and the database. The system is divided into three core layers: the presentation layer, the application layer, and the data layer each responsible for executing specific functionalities essential to delivering an efficient civic issue management solution.

The presentation layer, developed using React, provides an interactive and user-friendly interface for both citizens and administrators. Through various UI components and forms, users can submit complaints, upload media evidence, track issue status, and view notifications. The presentation layer communicates with the application layer using REST API calls and event-driven state management, ensuring fast and responsive interactions.

The application layer, implemented with Node.js and Express.js, acts as the functional core of the system. It handles data validation, business logic execution, role-based authorization, notification triggering, and secure communication with the database. RESTful API endpoints are used to manage all CRUD operations, establish secure connections, and facilitate seamless data exchange between the React frontend and the Supabase database.

The data layer uses Supabase, a PostgreSQL-based backend-as-a-service to store all civic issue related data, including user records, complaint details, location coordinates, media references, notification logs, and feedback entries. The database follows a normalized schema to prevent redundancy, improve query performance, and maintain data integrity. Queries are optimized using indexing and relational constraints to support fast data retrieval.

Additionally, the system incorporates security measures such as JWT-based role authentication and input validation middleware to prevent unauthorized access and injection attacks. Robust error handling, transaction management, and real-time database connectivity testing further ensure stability and reliability in real-world civic management environments.

3.1 Data Collection and Preprocessing

Data collection involves gathering essential civic information required to build different system modules. The dataset includes:

- **User information:** name, email, phone number, city, role (citizen/admin)
- **Issue details:** title, description, category, severity, location (latitude & longitude)
- **Media data:** attached images or audio files, file type, file size, storage reference
- **Notification data:** issue ID, message, notification type, read/unread status
- **Feedback data:** issue ID, citizen rating (1–5), comment, submission date

During preprocessing, the collected data is cleaned, structured, and normalized to fit into relational database tables. Invalid or duplicate entries are removed, field formats (such as dates, phone numbers, and coordinates) are standardized, and relational mapping between tables is defined using primary and foreign keys. This ensures the Supabase PostgreSQL database is optimized for accurate storage, efficient retrieval, and smooth integration with the Node.js backend.

3.2 Evaluation and Testing

Evaluation focuses on system performance, functionality, and reliability. Various testing techniques are employed to ensure the system meets real-world requirements:

- **Functional Testing:** Verifies that all modules complaint submission, admin dashboard, notifications, and feedback operate correctly.
- **Unit Testing:** Checks individual API endpoints and React components for correctness, such as insert, update, and fetch operations.

- **Integration Testing:** Ensures smooth communication between the React frontend, Node.js backend, and Supabase database.
- **Validation Testing:** Ensures data correctness, prevents duplicate issue entries, and validates location coordinate formats.
- **Security Testing:** Verifies JWT authentication, role-based access control, and protection against unauthorized API access.

The system demonstrates fast response times, secure authentication, and accurate results in all modules, confirming its effectiveness in real-world civic issue management workflows.

3.3 Implementation Workflow

The implementation workflow of the Civic Issue Reporter System follows a structured and iterative process:

1. **User Authentication:** Users (Citizen/Admin) log into the system using credentials validated through Supabase Auth with JWT token generation.
2. **Issue Reporting:** The citizen fills in the complaint form with title, category, severity, description, location, and optional media. The system stores the data securely via Node.js REST API.
3. **Admin Assignment:** Admin reviews incoming complaints on the dashboard and assigns priority levels (P1/P2/P3) based on category and severity.
4. **Status Management:** Admin updates issue status (Pending / In Progress / Resolved / Rejected). Each update automatically triggers a real-time notification to the citizen.
5. **Proof-Based Resolution:** Upon resolving an issue, the admin uploads resolution proof (images/video). Rejected complaints include a written justification for full transparency.
6. **Notification Delivery:** Real-time notifications are pushed to the citizen dashboard using Supabase subscriptions, keeping them informed at every stage.
7. **Feedback Submission:** After resolution, the citizen submits a rating (1–5) and comment. Feedback is stored and reflected in the admin performance analytics.
8. **Report Generation:** The admin can generate summary reports on issue trends, resolution rates, and area-wise complaint distribution using database aggregation queries.

4. RESULTS AND DISCUSSION

The Civic Issue Reporter System successfully achieved all its intended functions using React, Node.js, and Supabase. The system performed reliably during testing, showing fast data retrieval, accurate record updates, and smooth interaction between all three layers.

The login, complaint submission, and admin management modules operated correctly without errors. Issue status updates and automated notifications were delivered to citizens in real time. All CRUD operations executed successfully through RESTful API endpoints, ensuring data integrity and security.

Overall, the system demonstrated improved speed, transparency, and accountability compared to traditional manual civic complaint processes.

5. CONCLUSION AND FUTURE WORK

The Civic Issue Reporter System developed using React, Node.js, and Supabase demonstrates how digital solutions can significantly improve the efficiency and transparency of civic issue management. By automating core processes such as complaint registration, location tracking, admin assignment, status updates, and feedback collection, the system minimizes manual work and reduces the chances of delays and miscommunication.

The integration of a React-based interface with a Supabase PostgreSQL database ensures seamless data flow, secure storage, and real-time access to critical information. Node.js serves as a reliable bridge between the frontend and the database, enabling fast execution of API requests and maintaining data integrity throughout all operations.

Overall, it provides a structured, user-friendly, and scalable platform that enhances civic issue resolution, improves citizen-authority communication, and strengthens administrative accountability. The successful implementation validates the system's potential as an effective digital governance solution for urban municipalities.

Future enhancements to the system may include the following:

- Integration of SMS/Email notifications for instant complaint status updates and resolution alerts.
- Cloud-based deployment to enable large-scale usage across multiple cities and regions.
- AI-based issue categorization and priority detection to automate classification and improve response time.
- Heatmap visualization to identify high-complaint areas for better resource allocation.
- Advanced Analytics and Dashboards for monitoring resolution trends, admin performance, and citizen satisfaction.
- Duplicate issue detection to prevent redundant entries for the same location and category.
- Mobile application support for citizens and administrators to manage complaints on the go.

These enhancements will further improve the system's adaptability, security, and usability, allowing it to evolve into a comprehensive smart city management platform.

6. REFERENCES

1. Mozilla Developer Network . Web Development Concepts and Practices. <https://developer.mozilla.org>
2. React Documentation. <https://react.dev>
3. Node.js Documentation. <https://nodejs.org>
4. Express.js Documentation. <https://expressjs.com>
5. Supabase Documentation. <https://supabase.com/docs>
6. Leaflet.js Documentation. <https://leafletjs.com>
7. Web Development Tutorials and References. <https://www.w3schools.com>