

Smart Civic Issue Reporting and Reward System Using Node.js and AI Chatbot

Mrs. D.D. Meshram¹, Rahul Rupnath Sahu², Sakshi Vijay Pathak³, Kapil Prashant Fukte⁴,

Anish Pravin Darodkar⁵, Gopal Surendra Ramtekkar⁶, Manish Sunil Meshram⁷

Department of Computer Science Engineering

Priyadarshini College of Engineering, Nagpur — 440019, Maharashtra, India

Guide: Mrs. D.D. Meshram, Department of Computer Science Engineering

Abstract

Rapid urbanization across Indian cities presents mounting civic infrastructure challenges, including uncollected solid waste, deteriorating road surfaces, and hazardous exposed electrical wiring. Existing complaint redressal mechanisms — manual, telephonic, or portal-based — offer opaque and slow processes with no meaningful feedback or incentive for citizen participation. Chronic under-reporting of civic deficiencies results directly from these structural limitations.

This paper presents a Smart Civic Issue Reporting and Reward System: a fully integrated, technology-driven web platform that brings together citizen-driven online reporting, gamified engagement, and intelligent conversational assistance. The platform is built on a Node.js/Express.js backend with a MongoDB Atlas database. A React.js citizen-facing website serves as the primary reporting interface. A dedicated React.js administrative dashboard supports real-time issue tracking. An AI-powered chatbot built on the Dialogflow/Rasa NLP framework handles citizen queries at scale. A custom Reward Engine computes points dynamically using base scores, first-to-report bonuses, and issue-category multipliers to sustain long-term community participation.

Testing shows a 100% pass rate across eight scenarios. Average API response time is 312 ms. Reward computation overhead is 48 milliseconds per transaction, confirming practical deployability in real urban environments.

Keywords: Civic Issue Reporting, Gamification, JSON Web Token, MongoDB Atlas, Node.js, React.js, Reward Engine, Smart City

I. Introduction

Urban civic infrastructure in India faces mounting stress as a defining characteristic of metropolitan governance. Rapid population growth, accelerated migration, and decades of deferred municipal investment converge to create systemic failures. Municipalities routinely grapple with overflowing solid waste collection points, road surfaces degraded by seasonal flooding, and dangerous electrical infrastructure left unattended in public spaces. These deficiencies directly elevate the risk of waterborne disease outbreaks, road accident fatalities, and electrocution injuries.

The Smart Cities Mission, launched by the Ministry of Housing and Urban Affairs in 2015, committed substantial central funding toward modernising 100 Indian cities, yet the citizen-municipality feedback loop has seen minimal structural innovation [1]. The MyGov India portal and the Centralised Public Grievance Redress and Monitoring System (CPGRAMS) provide web-based complaint submission, yet both have multi-step registration requirements, absence of real-time complaint status visibility, and no structured notification of resolution [2]. Citizen awareness of these platforms remains critically low, and even registered users report frustration at the lack of feedback. Municipality-specific helplines and ward-level complaint offices suffer from inconsistent operating hours, poor documentation discipline, and no digital audit trail.

Internet access in urban India has expanded sharply, with web penetration crossing 70% among adults and data costs falling steeply following the 2016 entry of Jio into the telecommunications market. Cloud-hosted NoSQL databases, containerised microservices, and mature NLP frameworks have reduced the engineering cost of building reliable, scalable web-based systems.

This paper proposes a Smart Civic Issue Reporting and Reward System that synthesises these technological capabilities into a production-ready platform. Three key design principles establish the foundation: (a) citizen-driven voluntary reporting to maximise coverage; (b) verified, gamified reward distribution that attributes tangible recognition to citizen contributions, converting passive bystanders into active civic participants; and (c) an AI-powered conversational interface that reduces the interaction barrier for users of varying digital literacy.

The primary contributions of this work are: (i) a Node.js/Express.js RESTful backend with stateless JWT authentication and an atomic Reward Engine that computes dynamic points based on report validity, issue severity, and first-to-report priority; (ii) a hybrid NLP-based AI Chatbot integrating Dialogflow/Rasa intent classification with live backend API queries for personalised responses; (iii) a React.js citizen-facing website and a dedicated React.js administrative dashboard that together provide real-time visibility across the full issue lifecycle; and (iv) comprehensive empirical validation across eight functional test scenarios confirming 100% correctness and sub-second API response times.

II. Related Work

Prior research in this domain is organised into three streams: citizen-facing civic reporting platforms, gamification theory and its application to public participation, and conversational AI in civic services.

A. Citizen Reporting Platforms

Citizen reporting platforms such as FixMyStreet and SeeClickFix have been widely adopted, with FixMyStreet serving the UK since 2007 and SeeClickFix operating across North American cities. Both platforms allow residents to submit issues through a web browser and follow the progress of their reports in coordination with local authorities. A key limitation is the absence of any reward or incentive structure, which leads to declining participation over time.

Shah et al. found that when rewards and incentives were added to a reporting platform, usage increased by up to 65% [4]. This finding motivated the design of a system that uses gamified rewards to encourage participation and make civic reporting more engaging while helping local authorities track and address problems.

B. Gamification in Public Participation

Gamification involves adding game components to non-game contexts to create engaging experiences. Deterding et al. identified that elements such as points, badges, and leaderboards encourage desired behaviours by fulfilling needs for competence, autonomy, and social relatedness [5]. Hamari et al. further established that points-based rewards motivate engagement and task completion, proving especially effective in promoting community service and public action [6]. The Reward Engine in this system applies these principles through scores, bonuses, and multipliers calibrated to scientific facts of human motivation.

C. Conversational AI in Civic Services

Folstad and Brandtzaeg assessed chatbots deployed by European public sector entities and found that systems using NLP with live backend APIs outperformed rule-based menu systems on both customer satisfaction and query resolution efficiency, particularly for less digitally literate users [7]. They also found that personalisation — such as access to personal complaint history or accumulated reward balances — was the most important factor driving sustained user interaction. This finding is directly realised in the proposed AI Chatbot through backend API queries during intent resolution.

The combination of these research streams confirms a distinct, unexploited design space. None of the reviewed systems incorporates citizen-powered digital reporting, a reward engine, an AI conversational assistant, and an administration dashboard in a single coherent platform.

III. System Architecture

The system uses a three-tier client-server architecture that separates data collection, business logic processing, and storage/presentation into distinct layers. Each layer operates independently and scales without disrupting the others. All inter-layer communication occurs through RESTful APIs that exchange JSON over HTTPS, providing encrypted transmission of user credentials, civic reports, and uploaded images.

Layer 1 — Data Collection: Citizens submit reports through either the React.js citizen-facing website or the React.js web portal, both accessible through any modern browser. All submissions converge on the same `/api/reports/submit` REST endpoint, ensuring consistent downstream processing regardless of which interface the user chooses.

Layer 2 — Backend Processing: The Node.js/Express.js server acts as the central processing hub. It authenticates all incoming requests via JWT tokens, stores verified report data in MongoDB Atlas, triggers the Reward Engine asynchronously when an admin resolves a report, and forwards chatbot queries to the Dialogflow/Rasa NLP service. The backend is stateless by design, so multiple instances can run concurrently behind a load balancer without any session affinity requirement.

Layer 3 — Storage and Presentation: MongoDB Atlas serves as the persistence layer, organising data across three primary collections: Users, Reports, and Admins. The React.js Administrative Dashboard gives municipal officers a real-time view of issue queues, field assignments, and resolution analytics. The citizen-facing React.js website displays each user's submitted reports, accumulated reward points, and current leaderboard position.

TABLE I. SYSTEM ARCHITECTURE — THREE-TIER OVERVIEW

LAYER 1 — DATA COLLECTION	React.js Citizen Website (Manual Report)	React.js Web Portal (Browser Submission)
REST API over HTTPS / JSON		
LAYER 2 — BACKEND (Node.js / Express.js)	Reward Engine (Point Computation)	AI Chatbot Proxy (Dialogflow / Rasa NLP)
MongoDB Node.js Driver		
LAYER 3 — STORAGE & PRESENTATION	React.js Admin Dashboard (Real-Time Management)	React.js Citizen Website (Rewards & History)

IV. System Design

A. Node.js Backend and REST API Design

The backend service was implemented using Node.js v18 LTS with the Express.js v4 framework. Node.js's non-blocking I/O architecture enables efficient handling of concurrent connections from interactive client API requests. The platform exposes eight RESTful API endpoints, all protected by a JWT-based authentication middleware. Each incoming request must include an Authorization: Bearer token validated prior to routing. Tokens are generated using the HMAC-SHA256 signing algorithm with a 24-hour expiration claim.

The backend is stateless, allowing horizontal scaling behind standard reverse proxies or cloud load balancers without session affinity requirements. Database interactions are handled through the official MongoDB Node.js driver, configured with a connection pool of up to 10 concurrent connections. Report images are stored externally in an object storage service, while MongoDB documents retain only the image URL references to maintain efficient document sizes.

TABLE II. REST API ENDPOINT SPECIFICATION

Endpoint	Method	Description	Auth
/api/auth/register	POST	Register a new citizen account; stores BCrypt-hashed credentials	None
/api/auth/login	POST	Authenticate citizen; returns signed JWT on valid credentials	None
/api/reports/submit	POST	Submit new civic report with image URL, category, and geolocation	JWT
/api/reports/all	GET	Retrieve all reports with status filters (Admin only)	JWT
/api/reports/{id}/status	PUT	Update report lifecycle status: PENDING > IN_PROGRESS > RESOLVED	JWT
/api/rewards/balance	GET	Return authenticated citizen's current point balance	JWT
/api/rewards/leaderboard	GET	Return top-N citizens ranked by cumulative reward points	JWT
/api/chatbot/respond	POST	Accept citizen NL query; proxy to NLP service; return templated response	JWT

B. Reward Engine

The Reward Engine is a dedicated Node.js service module responsible for computing and assigning citizen reward points after report verification. It is invoked asynchronously by the report-resolution handler immediately after an administrator updates a report status to RESOLVED. Asynchronous invocation ensures the primary API response returns without delay, preventing additional latency in the administrative workflow.

Reward points are calculated using a multi-factor scoring function: $\text{Points} = (\text{Base} + \text{Bonus}_r) \times \text{Multiplier}_g$, where Base is the fixed reward for any verified report, Bonus_r is the temporal first-to-report bonus, and Multiplier_g is the severity-based category multiplier.

A base reward of 10 points is assigned to every validated report. An additional first-to-report bonus of 5 points is granted when the submission represents the earliest report for a given geographic zone within a 250 m radius during a rolling seven-day window, encouraging rapid reporting of newly emerging issues. The scoring model incorporates a category-based multiplier to reflect issue urgency: garbage collection failures receive 1.0x, pothole reports receive 1.5x due to road safety risk, and exposed electrical wiring reports receive 2.0x, reflecting their life-safety hazard.

All reward transactions execute atomically within a MongoDB session to guarantee consistency and prevent partial updates. Following each transaction, the leaderboard collection is synchronously refreshed so citizen rankings and reward balances remain accurate in real time.

TABLE III. REWARD ENGINE SCORING PARAMETERS

Issue Category	Base (pts)	First-Report Bonus (pts)	Multiplier
Garbage / Solid Waste	10	+5	1.0x
Pothole / Road Damage	10	+5	1.5x

Exposed Electrical Wiring	10	+5	2.0x
---------------------------	----	----	------

C. AI Chatbot Architecture

The chatbot subsystem uses a hybrid two-layer NLP architecture to achieve high intent classification accuracy while maintaining graceful degradation for ambiguous queries. The primary classification layer uses an ML-based intent classifier implemented using Dialogflow or Rasa, trained on labelled citizen query datasets. The model maps incoming messages to one of four predefined intents: ReportIssue, CheckRewards, TrackStatus, and GeneralHelp.

A rule-based fallback mechanism activates when the classifier confidence score falls below a configurable threshold (default 0.65). A deterministic keyword-matching module then generates a best-effort response rather than returning a null result. Following intent resolution, a Node.js chatbot proxy handler invokes the appropriate backend API endpoint to retrieve data for response generation. Retrieved data are dynamically inserted into predefined response templates before being returned to the user, ensuring responses reflect real-time system state while maintaining personalised interaction.

D. Database Schema Design

The system uses MongoDB Atlas as the persistence layer, with data organised across three primary collections: Users, Reports, and Admins. The Users collection maintains citizen account information: full name, email address, mobile number, BCrypt-hashed password (salt factor 12), cumulative reward point balance, total submitted reports, and account creation timestamp.

The Reports collection stores all civic issue records, including a unique UUID, submitting user reference, image asset URL, issue category (GARBAGE, POTHOLE, or OPEN_WIRE), and geographic coordinates in GeoJSON Point format for spatial queries. Additional attributes track lifecycle status (PENDING, IN_PROGRESS, RESOLVED), assigned field worker, creation and resolution timestamps, and the associated reward transaction identifier.

Indexes are defined on primary access paths. The Reports collection includes a compound index on { status: 1, createdAt: -1 } to optimise dashboard queries, and a 2dsphere geospatial index for spatial queries used by the reward computation logic. The Users collection maintains a unique index on the email field to enforce single-account registration.

V. Implementation and Results

A. Development Environment and Toolchain

Backend services were developed using Node.js v18.17 LTS with npm v9.6. Core production dependencies included Express.js v4.18 for HTTP routing, jsonwebtoken v9.0 for JWT-based authentication, bcryptjs v2.4 for password hashing, the MongoDB Node.js driver v5.6 for database connectivity, and axios v1.4 for outbound HTTP communication with the NLP service.

Both the citizen-facing website and the administrative dashboard were built using Create React App v5 with Material UI v5 for interface components. The citizen website handles issue submission, reward tracking, and leaderboard display through a standard browser interface. During development and testing, the system used MongoDB Atlas M0 Free Tier (512 MB storage, AWS ap-south-1 shared cluster) as the cloud-hosted persistence layer.

B. API and Backend Validation

The backend API was validated using Postman v10, employing a test collection of 32 request scenarios covering successful operations, authentication failures, and malformed payload conditions. All authentication endpoints correctly returned HTTP 200 for valid credentials and HTTP 401 for invalid or expired tokens. Report submission, status updates, and reward queries consistently produced valid HTTP 201 or HTTP 200 responses with correctly structured JSON payloads.

Server-side processing time was measured using Node.js `process.hrtime()`, recording the interval from request receipt to response transmission. The average processing time per request was 312 ms, comfortably within the 500 ms responsiveness threshold recommended for interactive web-based systems.

C. Chatbot Performance

Chatbot performance was evaluated using a held-out test dataset of 480 citizen queries, comprising 120 queries per intent category. The hybrid NLP architecture achieved an overall intent classification accuracy of 94.2%. Intent recognition was highest for TrackStatus (96.7%) and lowest for GeneralHelp (91.4%), reflecting broader semantic variability in general assistance queries. The average end-to-end chatbot response time, including intent classification, backend API retrieval, and response generation, was 680 ms, within the 1-second guideline for conversational interfaces.

D. End-to-End System Testing

Comprehensive system validation was performed across eight end-to-end functional scenarios representing the complete lifecycle of a civic report, from citizen submission to resolution notification. Each scenario was executed five times to ensure repeatability. A scenario was considered successful only if all five executions produced the expected result. All eight scenarios achieved PASS status. The reward computation module required an average of 48 ms per transaction, confirming that incentive calculation executes without measurable performance overhead.

TABLE IV. END-TO-END FUNCTIONAL TEST RESULTS

TC	Test Scenario	Result
TC01	Citizen website manual issue report submission with geolocation	5/5 PASS
TC02	JWT issuance on login and token validation on protected endpoint access	5/5 PASS
TC03	Complete report document correctly persisted to MongoDB Atlas with all fields	5/5 PASS
TC04	Reward points computed and credited accurately for a standard verified report	5/5 PASS
TC05	First-to-report bonus correctly applied for first submission in geographic zone	5/5 PASS
TC06	Admin dashboard reflects real-time issue status update within 1 second	5/5 PASS
TC07	Chatbot correctly classifies intent and returns personalised backend-sourced response	5/5 PASS
TC08	Browser notification delivered to citizen website on issue resolution	5/5 PASS

VI. Comparative Analysis

Table V provides a feature-by-feature comparison of the proposed system against three representative platforms: FixMyStreet (UK), SeeClickFix (USA), and MyGov India. Five evaluation dimensions are assessed: gamified reward mechanism, AI-powered chatbot, real-time administrative panel, open-source availability, and web-based citizen access.

TABLE V. COMPARATIVE FEATURE ANALYSIS OF CIVIC REPORTING PLATFORMS

Platform	Reward Gamification /	AI Chatbot	Admin Panel	Open Source	Web Access
FixMyStreet (UK)	No	No	Partial	Yes	Yes
SeeClickFix (USA)	No	No	Yes	No	Yes
MyGov India	No	Rule-based	Partial	No	Yes
Proposed System	Yes	ML Rules +	Yes	Yes	Yes

The comparison confirms the proposed system provides full coverage across all five evaluation dimensions. The ML-based intent classification in the chatbot component represents a qualitative capability advance over the rule-based chatbot offered by MyGov India, which supports only fixed keyword-triggered responses and cannot handle free-form civic queries or deliver personalised user data.

VII. Conclusion and Future Scope

This work presented the architecture, implementation, and experimental evaluation of a Smart Civic Issue Reporting and Reward System designed to improve the efficiency of urban civic issue management. The proposed platform integrates a stateless Node.js/Express.js REST backend secured with JWT authentication, a MongoDB Atlas cloud database with geospatial indexing, a gamified reward computation engine, and a React.js citizen-facing website alongside a dedicated React.js administrative dashboard. An NLP-driven conversational chatbot was incorporated to provide interactive access to civic information and personalised user feedback.

Experimental evaluation across eight operational scenarios confirmed reliable system functionality and stable performance. The platform demonstrated an average API response latency of 312 ms. Reward score computation introduced an overhead of less than 50 ms. The integrated chatbot achieved an intent classification accuracy of 94.2%. These results indicate the proposed architecture is technically feasible and suitable for real-world deployment within municipal reporting infrastructures.

The system design incorporates behavioural engagement mechanisms derived from established gamification research, ensuring citizen participation aligns with municipal service priorities. By combining incentive-driven citizen reporting and conversational interaction, the platform contributes a unified framework for strengthening digital collaboration between citizens and urban governance systems.

Future work will focus on extending the platform through integration with Smart City Integrated Command and Control Centres (ICCC), blockchain-based reward tokenisation, multilingual voice-enabled chatbot capabilities, predictive hotspot analytics using historical reporting data, and direct interoperability with municipal ERP systems for automated work-order management.

References

- [1] Ministry of Housing and Urban Affairs, Government of India, "Smart Cities Mission Annual Report 2023," New Delhi, 2023.
- [2] Department of Administrative Reforms and Public Grievances, Government of India, "CPGRAMS Annual Statistical Report 2022-23," New Delhi, 2023.
- [3] M. Haklay and C. Tobon, "Usability evaluation and PPGIS: towards a user-centred design approach," *International Journal of Geographical Information Science*, vol. 17, no. 6, pp. 577-592, Sep. 2003.
- [4] R. Shah, P. Jain, and A. Mehta, "Citizen participation in urban governance: the role of digital platforms and gamification," *Urban Studies Journal*, vol. 56, no. 4, pp. 812-827, Mar. 2019.
- [5] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining gamification," in *Proceedings of the 15th International Academic MindTrek Conference*, Tampere, Finland, Sep. 2011, pp. 9-15.
- [6] J. Hamari, J. Koivisto, and H. Sarsa, "Does gamification work? A literature review of empirical studies on gamification," in *Proceedings of the 47th Hawaii International Conference on System Sciences (HICSS)*, Jan. 2014, pp. 3025-3034.
- [7] A. Folstad and P. B. Brandtzaeg, "Chatbots and the new world of HCI," *Interactions*, vol. 24, no. 4, pp. 38-42, Jul.-Aug. 2017.