CivicEye: A Smart Civic Issue Reporting and Management System for Urban Governance

Ayush Alliwar[1], Sahil Memane[2], Vivek Gadhave[3], Ms. Prajakta[4]
[1,2,3,4]*School of Engineering, Ajeenkya DY Patil University, Pune, Maharashtra*

***Abstract*—Rapid urbanisation in Indian cities has created significant challenges for municipal infrastructure management. Civic issues such as potholes, illegal waste dumping, drainage failures, water leaks, and non-functional streetlights remain unresolved for extended periods due to fragmented, paper-based complaint processes that offer citizens little to no visibility into resolution progress. This paper presents CivicEye, a web-based civic issue reporting and management system built using Python (Flask), MySQL, and Bootstrap. The system provides a centralised digital platform through which citizens can submit geo-tagged, image-evidenced complaints and track their status in real time, while municipal administrators use a dedicated panel to review, triage, and resolve issues efficiently. Functional, performance, and usability evaluations demonstrate that CivicEye reduces manual effort, improves transparency, and accelerates complaint resolution. The findings indicate that CivicEye is an effective and scalable foundation for smart-city civic governance, with planned extensions including a native mobile application, push notifications, and AI-assisted classification.**

***Index Terms*—civic complaint management, e-governance, smart city, Flask, web application, urban infrastructure, real-time tracking, urban governance.**

I. INTRODUCTION

India's urban population crossed 500 million in 2021 and is projected to reach 600 million by 2031 [14]. This unprecedented growth places enormous strain on civic infrastructure. Potholes, uncollected garbage, drainage blockages, water-main leaks, and broken streetlights are among the most common complaints received by Urban Local Bodies (ULBs). Despite the digital revolution, the predominant complaint-handling mechanism in many regions remains manual, involving physical visits to municipal offices and paper-based documentation
[6].

The Government of India's Smart Cities Mission recognises digital citizen engagement as a core pillar of urban development [14]. Platforms such as the Swachh Bharat App address sanitation-specific complaints but are narrowly scoped [13]. Centralised Grievance Redress systems like CPGRAMS provide a federal-level interface but often lack the localized agility required for municipal-level issue resolution [16]. CivicEye addresses these gaps by delivering a lightweight, transparent web application that serves as a single window for all civic complaint categories.[1]

II. LITERATURE REVIEW

2.1. Existing Systems

Several digital platforms for civic engagement have been deployed globally. The Swachh Bharat App allows citizens to report sanitation violations via geo-tagging [13]. However, as noted by Balaji et al. (2020), citizen centric e-governance frameworks must move beyond single-issue reporting to integrated management systems that offer cross-departmental visibility [16].

Current research emphasizes the need for accountability in public service delivery. The lack of a feedback loop in traditional systems often leads to citizen apathy.

By implementing a real-time status update mechanism, CivicEye aims to foster a sense of participation and trust between the residents and the local government body.

2.2. Architectural Foundations

Modern urban governance requires robust, scalable systems. Batty et al. (2012) argue that the "Smart City" of the future relies on real-time data flow between the public and administration [17]. CivicEye adopts this philosophy by utilizing a high-performance stack comprising Flask and MySQL [10, 11]. The use of Python-based frameworks allows for rapid development cycles and ease of integration with future AI/ML modules for automated image recognition of potholes or garbage piles.

III. METHODOLOGY

The development of CivicEye followed the Software Development Life Cycle (SDLC) using an iterative approach. This allowed for continuous refinement based on testing feedback.

3.1. Requirement Analysis

Initial research identified four major stakeholders: Citizens (Complainants), Field Staff (Resolvers), Administrators (Managers), and Local Government Entities. Functional requirements included secure authentication, multi-media complaint submission, geo-location logging, and status tracking.[2]

3.2. System Modelling

The system was modelled using standard UML diagrams. User personas were developed to simulate common usage scenarios, such as a citizen reporting a broken streetlight at night or an administrator assigning a garbage collection task to a specific ward team.

IV. SYSTEM DESIGN AND ARCHITECTURE

4.1. Technology Stack

CivicEye follows a three-tier web architecture [7, 9]:

1. Frontend: Developed using Bootstrap 5 for a responsive, mobile-first design [12]. It ensures that citizens can report issues directly from their smartphones at the site of the incident.
2. Backend: Python 3 and Flask handle application logic, session management, and routing [10]. Flask was chosen for its lightweight nature and extensive library support.
3. Database: MySQL 8 manages relational data with strict ACID compliance [11].

4.2. Database Schema

The database is designed to minimize redundancy while maximizing query performance. The schema includes tables for users, complaints, departments, and status logs.

Table 1: User Table Schema

Field	Data Type
user_id	UUID (PK)
full_name	VARCHAR(100)
email	VARCHAR(255)
password_hash	VARCHAR(255)
role	ENUM('Citizen', 'Admin')

Table 2: Complaints Table Schema

Field	Data Type
complaint_id	INT (PK, AI)
user_id	UUID (FK)
category	VARCHAR(50)
description	TEXT
image_path	VARCHAR(255)
latitude	DECIMAL(10,8)
longitude	DECIMAL(11,8)
status	ENUM('Pending', 'In-Progress', 'Resolved')
created_at	TIMESTAMP

V. IMPLEMENTATION

The implementation phase focused on modular development. The application was divided into a citizen portal and an admin dashboard.

5.1. Citizen Portal

The portal allows users to register using their email. Upon login, they are presented with a dashboard showing their past complaints and a "New Complaint" button. The submission form uses HTML5 geolocation APIs to capture the user's current coordinates.[3]

5.2. Admin Dashboard

The admin dashboard provides a high-level overview of all active issues. Admins can filter complaints by category, ward, or status. Each complaint can be viewed in detail, including the uploaded image evidence.

VI. TESTING AND RESULTS

6.1. Functional Testing

We conducted extensive functional testing to ensure all modules work as intended. Test cases included valid/invalid logins, image upload limits, and role-based access control.

Table 3: Functional Test Results

Test ID	Component	Expected	Status
FT-01	Login	Success on valid	PASS
FT-02	Submission	Data saved with GPS	PASS
FT-03	Image Upload	Store path in DB	PASS
FT-04	Role Check	Citizen can't access Admin	PASS

6.2. Performance Benchmarks

As per testing protocols defined by Sommerville [7], the system was benchmarked against standard industry response times.

Table 4: Performance Evaluation

Operation	Measured	Threshold
Authentication	1.8 s	2.0 s
Data Submission	2.6 s	3.0 s
Query Response	0.9 s	1.0 s
Admin Dashboard Load	1.4 s	2.0 s

VII. DISCUSSION

The integration of real-time tracking directly addresses the lack of visibility identified in traditional paper-based systems. By providing an auditable history of every complaint, CivicEye aligns with the transparency goals of modern e-governance frameworks [16, 17].

During the pilot testing, users noted that the ability to upload images significantly increased their confidence that the issue would be addressed. From an administrative standpoint, having the exact GPS coordinates eliminated the need for manual location verification, which is a major bottleneck in existing workflows.

VIII. CONCLUSION AND FUTURE WORK

CivicEye demonstrates that a resource-efficient, opensource technology stack can deliver a transparent and accountable civic service platform. The system successfully reduces the communication gap between urban residents and local authorities. Future work includes:

- Mobile App: Developing a native Android/iOS application for better accessibility.
- AI/ML Integration: Implementing automated image classification to verify the category of the reported issue (e.g., distinguishing between a pothole and a garbage dump).
- Public API: Opening the data to third-party developers for urban planning research.

REFERENCES

[1] J. L. Galay and R. Encarnacion, "Web-Based Event Management System for Educational Institutions," International Journal of Computer Applications, vol. 182, 2018. [6] R. S. Pressman, Software Engineering: A Practitioner's Approach, 8th ed. McGraw-Hill, 2014.

[2] I. Sommerville, Software Engineering, 10th ed. Pearson, 2016.

[3] Pallets Projects, "Flask: Web Development, One Drop at a Time (v3.x)," [Online]. Available: https://flask.palletsprojects.com. [Accessed: Apr. 2025].

[4] Oracle Corporation, "MySQL 8.0 Reference Manual," [Online]. Available: https://dev.mysql.com/doc/. [Accessed: Apr. 2025].

[5] The Bootstrap Authors, "Bootstrap v5 — The World's Most Popular Front-End Framework," [Online]. Available: https://getbootstrap.com. [Accessed: Apr. 2025].

[6] Government of India, Ministry of Housing and Urban Affairs, "Swachh Bharat Mission — Digital Reporting Portal," [Online]. Available: https://swachhbharatmission.gov.in. [Accessed: Apr. 2025].

[7] Smart Cities Mission, Government of India, "Smart Cities Mission — About the Mission," [Online]. Available: https://smartcities.gov.in. [Accessed: Apr. 2025].

[8] IEEE Xplore Digital Library, “Smart City and Civic Issue Management Research Collection,” [Online]. Available: https://ieeexplore.ieee.org. [Accessed: Apr. 2025].
[9] P. Balaji, R. Venkatesan, and S. Moorthy, “Citizen-Centric e-Governance Frameworks for Urban Local Bodies in India,” International Journal of e-Government Research, vol. 16, no. 2, pp. 34–48, 2020.
[10] M. Batty et al., “Smart Cities of the Future,” European Physical Journal: Special Topics, vol. 214, no. 1, pp. 481–518, 2012.