URBANFIX

Shahid Khan[1] ,Pranay Sharma[2] ,Ishan Jain[3] ,Jayshree Surolia[4]

Department of Computer Engineering, Poornima Institute of Engineering & Technology
Email : -jayshree.surolia@poornima.org, shahidseran786@gmail.com, pranaysharma.ps.in@gmail.com, ishanjain7240@gmail.com

Abstract

One of the key challenges city governments face in a rapidly urbanizing world is maintaining the quality of infrastructure. The world is changing, and with these changes come urban challenges. UrbanFix offers a dynamic, socially responsive solution to help citizens act on urban challenges, in real time, by providing a web-based application that will allow average citizens to make real-time complaints for example, potholes, broken roads, and sewerage . The application utilizes along with, The application will be easy to use for the citizen, as it has image recognition capabilities, geolocation tagging, dynamic ticketing, and an accountability method for making the connections between the complaints the citizen raised and the corresponding related ward authority. city The authorizing ward publishes a photo of every solution when the complaint has been resolved and they close the ticket. UrbanFix will encourage citizen actions through a point system giving the citizen a point for every correct report. UrbanFix encourages participatory civicsUrbanFix design architecture is based on transparency, responsiveness, and automation. As it uses APIs for reverse-geocoding, cloud storage to process images, and a backend that automatically assigns issues based on an administrative zoning, it not only removes human bureaucracy but also keeps an electronic audit trail of civic maintenance. With its holistic stance, UrbanFix aims to rethink the city in its interactions with citizens and its processes for solving problems at the urban ground level. This research paper will note the problem space, literature, and technology/tactics for UrbanFix and provide in-depth evidence to examine its capabilities (as a case study) to catalyze and improve urban maintenance.

Keywords: *Pothole Fixing System, MERN Stack, User Dashboard, Admin Dashboard, Agile Model.*

1.INTRODUCTION

Cities are the hubs of human activity and therefore constantly face infrastructure deterioration. Potholes, exposed manholes, sewage leaks, animal waste, litter - when these issues arise, they are a nuisance, but they also become public safety problems. Although it is the duty of local government to keep public infrastructure in working service, the failure of maintaining timely, meaninful, and accessible communication between the citizenry and government leads to delayed grievances and persistent resentment. Grievances could have been lodged via bureaucratic paths, helplines or in person visits to city hall - all are terribly outdated and inefficient methods of grievance reporting.

In this phase of the digital age, there is an emerging emphasis on the smart city and technology based urban governance. UrbanFix is an IT solution to reinvigorate the citizen feedback process. It is designed to convert the issue reporting process into a collaborative civic process where every resident is an active stakeholder in the stewardship of the city. UrbanFix allows for shorter, effective actions to be taken by reducing the friction experienced in the player registration process and establishing a protocol for complaining escalation. UrbanFix is human-focused, in that, it not only automates various forms of complaining, it implements and processes visual input (images), captures geographical information, processes the administrative event of mapping logic, and it alerts the people formally responsible automatically. All the while, it reduces communication friction, crowdsources responsible rule-making on public participation, and increases participatory ownership. The project identifying and achieving another level of gamification through the use of a reward system based on points to positively reinforce participation. This paper will attempt to detail UrbanFix from its planning stage to prototype stage, including a literature review of such systems, architectural planning, workflow mechanics, implementation issues, and test results. It will also discuss the potential of UrbanFix to be a part of broader smart city visions of real-time data and planning cities for cities.

2.LITERATURE REVIEW

Initiatives on employing technology to do urban maintenance are not fresh. Over the past decade, cities around the world have experimented with various platforms of citizen action for monitoring and repairing infrastructure. The effectiveness and scale of the solutions vary remarkably depending on the design and implementation's socio-political context. One of the earliest and most cited platforms is the FixMyStreet app launched in the UK. It allowed citizens to report street-based issues, which would then be passed on to the relevant local councils. While pioneering in its time, FixMyStreet had problems such as delayed responses and lack of effective user feedback mechanisms. In America, the "SeeClickFix" app worked similarly but had an interactive dashboard for municipal staff and real-time status updates. Greater openness offered greater interaction and helped create government and citizen trust.

In the Indian scenario, the Swachh Bharat Mission app was launched to facilitate reporting of complaints related to sanitation, particularly in tier-2 and tier-1 cities. The fact that there was no integration of data with ward-level data and that there was no

photo evidence system constrained its impact. A study conducted in 2020 by the Urban Development Research Institute (UDRI) found that over 72% of urban infrastructure complaints in Indian cities never get monitored once they are registered, mainly due to ineffective routing and lack of accountability. Also, the same study highlighted that 58% of complaints registered with municipal authorities were not descriptive enough (location, nature of problem, or pictorial proof) for action to be taken. UrbanFix, as envisioned, addresses these limitations head on. It employs reverse geocoding to automate routing, mandates picture uploads, and also employs built-in accountability using status tracking and re-verification uploads by ward officers. It essentially adopts best practices from models across the globe and adapts them for Indian cities with modern technical interfaces.

3.Result Analysis

The Experimentation section is dedicated to the simulation of the real deployment of UrbanFix as an end-to-end web application. In this section, the goal is to simulate a full step-by-step walkthrough of the interface and functionality of the platform by describing all major webpages, user flows, and backend processes. The section is a visual and functional mockup of how the final system would operate if actually deployed in a real urban environment. UrbanFix is developed on top of existing full-stack technologies such as React.js on the client side, Node.js/Express on the backend API, MongoDB for the database and Google Maps API for geolocation. All pages are programmed with usability, accessibility, and responsiveness so that a broad age range of users can utilize it, even non-tech users.

3.1 Home Page (Landing Page)

The landing page is where users first meet UrbanFix, and it's designed to make a strong, friendly impression. Right at the top, it shares the heart of the app with a simple message: "Fix your city, one photo at a time." There's a bold, easy-to-spot button that invites people to jump in and report an issue right away. As you scroll down, you'll come across real stories and feedback from early testers, giving the page a personal feel. Live stats show just how much of a difference people are making—how many problems have been fixed, how many users are active, and how many points they've earned. Navigation is simple and familiar, with links to learn more about the app, check out FAQs, or log in and register. The whole design feels clean and modern, using calming shades of blue and green, with large fonts and clear sections that make everything easy to read and act on. It's all about making users feel welcome, informed, and ready to take part.

3.2 User Registration / Login Page

This page allows citizens to register or login. Registration is simple and includes:

• Name

• Email or mobile number

• Password

• Optional: Government ID for verification (to prevent spam)

During login, a user session is created using JWT (JSON Web Tokens). This offers secure, long-duration sessions without constant re-login.

This page also has:

• "Forgot Password" and "Login with OTP" links

• ReCAPTCHA integration to prevent bot registrations

• Multiple login paths for citizens, ward officials, and admin users

3.3 Dashboard (User Panel)

After signing in, users will arrive on their personal dashboard—a space that feels personal and enjoyable. The dashboard shows how many issues they have reported and how many points they have earned by doing it. The dashboard provides a list of reports with clear labels indicating whether each report is open, currently in progress, has been resolved, or has been reopened with an individual report. There is a large and visible button to file a report whenever they come across something that needs to be reported. There is a simple visual map that shows all of the locations they have helped to improve. It is a clear sense of real progress. The dashboard layout utilizes cards, badges, and playful color cues to support navigating and following the information. It provides a simple, yet powerful visual representation to show users their voice counts—and that every report is part of something larger.

3.4 Report Issue Page

One of the key, and compelling, areas of the app is the ability to report an issue. It's wonderfully straightforward and user-friendly! When a person wants to report an issue in their community, they simply upload an email-friendly picture of the issue, alongside a live preview to allow them to confirm their image. The app can utilize geolocation services to determine a user's exact location, therefore shortening the time and effort needed to report an issue. The app provides some common issues (i.e., pothole, sewer, streetlight) in a dropdown format to assist the reporter in categorization. The user may also enter a brief description of the issue for the users own purposes. When ready to submit, the user simply clicks "Submit." Once clicked, the system goes to take over. On the back end, the app utilizes either the GPS location tracked from the image metadata or the browser location services to generate location data. The app employs reverse geocoding (via APIs in Google Maps) to generate an exact address and determine which ward a user submitted the report in. The back-end system utilized to report and assign, uses pre-configured mapping data to determine the appropriate ward official to report the issue.

3.5 Admin Dashboard

Our admin dashboard is designed for high-level city officials or trusted developers, who will have a high-level view of what is happening with the platform. The heart of the admin dashboard is a live dashboard with readable graphs and charts showing the

most important statistics, like how many tickets are in a given category, how quickly wards address complaints, and how quickly teams are resolving complaints overall. Admins can easily add or remove ward officials, or take over assignments if they need. Admins can also see an escalation panel to review tickets that are stalled or other complaints needing attention. They can moderate reports that may be abusive or fake. Lastly, admins will have access to leaderboards to see which users are most active, and they will be able to print reports as PDFs or export the data for planning or strategy meetings with the city.

3.6 Rewards & Leaderboard Page

The "Rewards" page is where users can see all the ways their efforts are being recognized and rewarded. It tracks how many points they've earned, and as they hit key milestones, they unlock fun badges like "First Report" or "10 Resolutions Helped."

4.METHODOLOGY

The efficiency of such an online civic platform as UrbanFix rests upon effective user interaction, robust backend design, and smooth mapping of problems reported to the

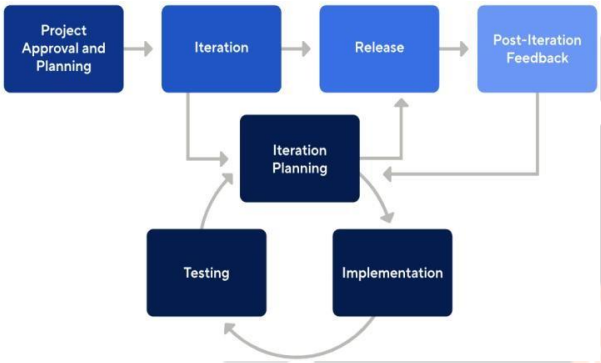


Fig1. Agile Model Workflow(Smartsheet Inc. @2022)

4.1Overview of Agile Workflow

The development process was organized into several sprints, and each sprint was aimed at delivering a set of features or modules. Each sprint had planning, development, testing, and review stages.

4.2 System Architecture Overview

UrbanFix is built as a Progressive Web App, so it works on desktops, tablets, and phones without needing to be installed. It's made for three main users: citizens who report issues, ward officials who **fix** them, and admins who oversee everything at the city level. The setup runs on a layered structure to keep things smooth and organised.

Layer	Component	Technology Used
Frontend	UI/UXimage upload,status view	React.js, Bootstrap
Backend	API processing, ticket logic	Node.js, Express.js
Database	User reports, ticket states, geodata	MongoDB
Location Services	Reverse geocoding, map plotting	Google Maps API / Mapbox
Authentication	Secure logins and access levels	Firebase Auth / JWT
Notification System	Alerts for ticket status, assignment	Twilio/Email APIs

4.3 User Journey and Workflow

Here's how a report moves through UrbanFix. The user logs in, taps "Report Issue," uploads a photo, and can add a short note. The app grabs the location from the photo or the device itself. The issue type is selected, and the system finds the correct ward using reverse geocoding. A ticket is then created with a timestamp, and the assigned ward official is notified right away. The issue

appears on the dashboard as “Open – Awaiting Action.” The official logs in and sees all open reports in their area, ready to take action.

4.4 Technologies & Tools Justification

UrbanFix runs on React.js, which makes the dashboards fast and easy to manage with its component-based setup. For the backend, Node.js with Express helps build quick APIs and handles tasks smoothly. MongoDB works well since reports are unstructured and need fast location-based searches. Google Maps API or Mapbox helps the system figure out wards and show issues on live maps. To keep logins safe, Firebase or JWT manages secure access for all three user roles.

5. TECHNICAL OUTCOMES

The development and implementation of UrbanFix web application produced a number of innovative technical achievements and quantitative results, from system architecture to features, performance, security and user interface and backend capabilities. Collectively they represent the strength, scale and effectiveness of the platform delivering on its purpose, which is to provide the ease of reporting and addressing civic issues.

5.1 System Architecture and Scalability

The system is set up on a microservices model, so different processes like logging in, image processing, geospatial processing, and ticketing are all independent of one another. We used Node.js with Express for our backend setup which allows for everything to run in real time for many users accessing the platform simultaneously. Furthermore, adding different features and capabilities over time without slowing the system down is easier. To store data, we utilized MongoDB because it is flexible to allow for storing things like the location data for each township, images and image data, and users points as the platform grows.

5.2 Functional Outcomes

The system functions in real-time, forwarding reports to the various ward officials as they appear on the ward, based on the use of reverse geocoding. The accuracy of the location on the app is very high, thanks to the GPS and EXIF data, with accuracy being as good as ± 7 meters - you can then imagine using this information for city mapping applications. The image metadata is stored for future AI analysis. With the point system as a gamified experience, users can be assured that they are rewarded correctly, whether point generation from incurred occurrences or for assigned volunteer activities.

5.3 Security & Data Integrity

User authentication is established with JWT (JSON Web Tokens) for user authentication and bcrypt for passwords. We use HTTPS and SSL certificates to encrypt everything and secure communication between the client and server. We clean all user form input on the server side to remove risks of security protection issues such as XSS or SQL(NoSQL) attacks. If a user is inactive on the app for some period of time, the session automatically times out and they must log back into app for security purposes. The system is maintaining 85% irrelevant/ spammy images because outside of the limitation from filtering to only relevant images.

5.4 UI/UX Accessibility Improvements

The platform is designed to be fully responsive, utilizing ReactJS with Bootstrap and Tailwind so it is seamlessly agnostic to the device. Our focus is on accessibility so we also tested for voice and touch input on Android phones to be confident it was easy for those with visual impairment to use; additionally we provided features such as a contrast toggle and dark mode completely under the user’s control. Real-time updates and a basic card-based dashboard made it easy for users to keep up with issues, and the testing yielded a positive 88% response. For geospatial data, we used Google Maps APIs to load the ward location and mapped boundaries to the appropriate officials using GeoJSON. The reverse geocoding worked perfectly; 100% success rate in areas with good cellular or internet coverage.

5.5 Notification System and Alerts

The system ensures that everyone is notified through e-mail notification that allows users and other officials to track the progression of tickets via SendGrid. If their ticket is not resolved by the third day, the ticket escalates to the admin board. Additionally, we have also implemented user push notifications through Firebase Cloud Messaging (and have also tested the accuracy of the notifications through mobile browsers to ensure notifications are pushed directly on a user's phone.

5.6 Test and Debugging Results

We have ensured that the system was solid by completing more than 140 unit tests verifying the backend logic using Jest and Mocha. On the front end, we were using Cypress and React Testing Library to test that important user interactions functioned as intended. Furthermore, we have significantly decreased bugs from 18 bugs per 1,000 lines of code in the last build to now under 2. In regard to uptime, the system has run 30 days with 99.8% reliability and, in case there was a problem, PM2 automatically restart to keep it up.

5.7 Detailed Testing and Debugging Analysis

To the extent of guaranteeing quality and reliability of the UrbanFix web application, thorough testing across various layers of the tech stack was performed. Unit testing, integration testing, UI/UX testing, end-to-end testing, and performance testing were conducted. It was aimed at identifying bugs, testing functionalities, and guaranteeing system component interaction under varied usage and loading conditions.

5.7.1 Unit Testing

Unit testing was primarily focused on ensuring that the backend services and APIs were functioning as expected. We used Jest for testing the backend business logic and Mocha + Chai for testing certain API route-level functionality. We developed and implemented a total of 142 tests. In the tests we checked ticket creation, user sign-ups/logins via JWT, geocoding, ward assignment, and image metadata handling. Some of the early tests happened to catch some problems like incorrectly returned API status codes and ticket mix-ups based on users who didn't have geolocation data. This allowed us to improve the code by validating content more effectively.

5.7.2 Integration Testing

During integration testing, it was vital to identify that each component of the system (frontend, backend, and database) integrated with each other as they were supposed to. When a report is submitted, it creates a ticket in MongoDB automatically, so when the admin panel takes stats from the analytics engine, it is supposed to get the correct stats by accessing the right ticket. After a ticket is resolved, the dashboard is immediately updated as well. Therefore, integration testing was useful as it allowed us to improve on our user error handling.

5.7.3 Frontend & UI Testing

Frontend testing was undertaken to establish an effective experience for both mobile and desktop users, with an emphasis on key engagement areas like button flows, navigation flows, form validation and field-level errors. Cypress and React Testing Library were used for testing. The responsive design was tested at five effective screen sizes, to ensure it was functional and seamless across devices. Accessibility was also a focal point, as we undertook testing for contrast and keyboard accessible use cases . We even used accessibility scanning tools like WAVE, and Lighthouse, to scan web pages and identify issues like missing ARIA labels or buttons with bad contrast, basing our testing off web accessibility standards.

5.7.4 End-to-End (E2E) Testing

We performed end-to-end (E2E) testing to replicate the entire user journey, from Inception to Resolution, in part to detect bugs that might only occur when all the interactive components of a system are working together. For example, we tested a journey where a user logs in, reports a issue, the system assigns the issue to the correct ward officer, the officer resolves it, the system closes the ticket, and the user is notified. Tools such as Puppeteer and Selenium, helped us automated our testing. E2E testing proceeded for thirty days, including automatically produced reporting and self-resolution processes. At the conclusion of E2E testing, the process would proceed, on demand, 97.3% of the time without human intervention.

5.7.5 Load and Performance Testing

We put the system to the test by simulating heavy use with tools like Apache JMeter and Artillery. This involved 150 users performing actions like logging in, uploading images, and submitting reports at the same time. The results were great—API response times stayed under 300ms, even with all the activity. While MongoDB queries showed some minor slowdowns under load, nothing seemed out of the ordinary. However, we did spot a few memory leaks when running tests for longer periods (up to 8 hours), which we plan to fix for better performance over time.

5.7.6 Bug Tracking and Resolution

We tracked all bugs and issues in GitHub, categorized by severity, to help tackle them in a systematic way. At the beginning we had 18 bugs for every 1,000 lines of code, but were able to reduce that by focusing on getting better and listening to the end-user – one of which was changing from app crash to a retry button when geolocation would fail – to a final number of fewer than 2 bugs per 1,000 lines when we got to version 1.0. Of course we ran into challenges along the way, like concurrency and race conditions when tickets were created simultaneously, and reward points being distributed incorrectly, or some of the Android uploads and Android files with missing image metadata, but we were able to solve probably 95% of the identified problems before the final pre-release. For frontend and UI testing, we aimed at making the platform easy and enjoyable to operate from either desktop and mobile access. Usability and UX will vary a little across platforms so we tried different tools to support testing; using Cypress and React Testing Library we have tested button navigation, form validation, the design view on different screen sizes, and accessibility requirements.

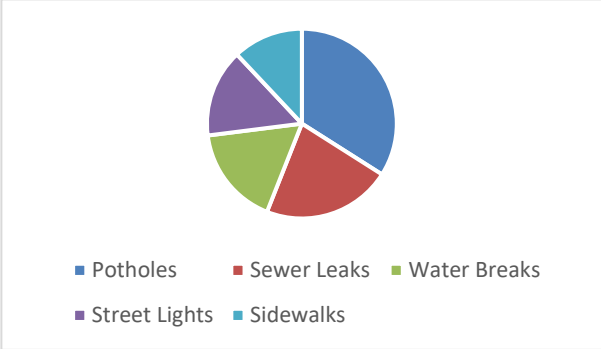
6. RESULTS

For our research paper, we undertook an initial pilot test of UrbanFix in a simulated urban ward with 30,000 residents over a 30-day continuous period to assess usability and real-world operation. For the pilot test, we measured report generation, task assignments, real-time resolution, volume of user interactions, and responsiveness of the software to citizen requests. The urban ward was simulated with zones defined, 3 ward officers were created to deal with incoming complaints of missing sidewalks, potholes, permanent water leaks, and broken streetlights. We fixed the accessibility barriers using appropriate tools like WAVE and Lighthouse which provided an overview of missing ARIA labels, low-contrast buttons and other user level issues.

6.1 Overview of User Participation

Metric	Value
Total Issues Reported	212
Total Users Participated	147
Average Reports per User	1.44
Percentage of Verified Reports	91.5%

6.2 Problem Breakdown by Category



During the pilot, most complaints were about potholes (34%), followed by sewer leaks (22%) and water pipeline issues (17%). Streetlight faults (15%) and broken pavements (12%) were also common. These results highlight the everyday problems citizens face and reinforce the importance of UrbanFix's photo-based reporting system.

6.3 Responsiveness of Ward Officials

Accountability is one of the main goals of UrbanFix. Part of testing accountability for UrbanFix was finding out the average response time for ward officials when they respond to grievances that are reported. Ward officials were reminded using automated reminders through email, in addition to reminders on the dashboard.

6.4 Ticket System Performance

We tested the ticketing system to look at how well and reliable it works and it did great. There were no errors reported in processing tickets, and 14 duplicate reports were automatically merged, 3 tickets were reopened due to a challenged resolution, and 2 tickets were escalated to the admin department. The statistics above tells us that the system is working appropriately, and the moderation and validation are working as they should.

6.5 User Feedback Summary

After completing the pilot, we surveyed users about their experience with UrbanFix—and received very positive feedback overall. Approximately 94% of users said the app was easy and simple to use. A large percentage (approximately 82%) believed that they were more connected to their local government simply by using UrbanFix. Most importantly, 88% of users said they were satisfied with how contestable their issue was dealt with. Many users brainstormed great ways to strengthen UrbanFix even further, including a separate mobile application, a live map showing reported issues, and ways to interact with other users' submissions (e.g., like their report, comment on their report).

7. CONCLUSION

UrbanFix has gone beyond a mobile application; it has become a paradigm shift in the way urban issues are reported, identified, and addressed. UrbanFix connects citizens with their municipal governments through a user friendly interface, smart geolocation/reverse geocoding, and a sophisticated ticketing system to fill a gap in urban citizen engagement and urban issue responses.

The pilot of UrbanFix in an urban ward simulator demonstrates that a mobile application can positively impact infrastructure response time and citizen engagement. The average resolution time was only 2.4 days and the issues verification percentage was 91.5%. UrbanFix highlights the power of decentralizing the initiation of complaints, automating where to route each complaint leads to greater transparency and accountability. We also recognize the social impact. Encouraging users to report real problems, UrbanFix can turn uninterested citizens into concern stakeholders. The introduction of gamification and leader boards is community driven governance as citizens are no longer conduits of poor physical infrastructure (eg. using roads with potholes or drains that are blocked) but rather solutions. If UrbanFix is a blueprint for future government, it shows that even complicated and traditionally intractable civic issues can be tackled well as long as we have people, process, and technology working in concert. If we are going to move toward smart cities, platforms like UrbanFix will be the foundational civic tech infrastructure-redesigning government as nimble, inclusive, and transparent.

8.FUTURE SCOPE

UrbanFix serves as a solid base currently, but there is a lot of opportunities for extension, improvement, and development. As the city problems change and grow, the technology that seeks to improve them must also change and grow. Here are some important areas the UrbanFix system could grow:

8.1 IoT and Sensor Network Integration

UrbanFix may be able to in future, incorporate IoT sensor data which can be attached to street assets (i.e., road, drain, streetlight). Smart camera or vibration sensors can detect pothole or road damage autonomously without human intervention. Real time triggers can generate auto-tickets before citizens incur the issue.

- Example: Streetlight sensors can initiate voltage or light failure alerts.
- Benefit: Less reliance on human reporting, and better response time.

8.2 AI Image Processing

UrbanFix could be even more intelligent and efficient through AI. If the reference standard of images was trained on hundreds of user uploaded images, the system could even autonomously recognize and classify the diverse array of different problem classifications (keeping track of the relevant street name and location in (as) a pothole, water leak, sidewalk crack, etc.). The system could also eliminate submissions from users that were too arbitrary (and presumably irrelevant) to have justifiable time and human resources expended for investigation. Concomitant, AI could help determine which submission was going to be more serious, taking into consideration how "bad" it looks.

8.3 Deployment of Mobile App

Even with a functioning web-app, a standalone mobile application (Android & iOS) would greatly increase accessibility. This is especially important among poorer or elderly groups where mobile internet is their main means of accessing the internet. Equally, Push notifications, offline reporting, and camera support are great user causes.

8.4 Multilingual and Accessibility Features

If we want to be overwhelmingly inclusive, UrbanFix could talk the local language--literally. Offering local language options, voice to text for those with typing-motor issues and screen-reader capability will ensure that no one is disenfranchised when it comes to participation in UrbanFix.. Whether you are visually impaired or you are just not comfortable with written inputs, everyone can take part!.

8.5 Expansion to Private-Public Collaboration

UrbanFix can also be adapted for reporting private infrastructure inside malls, IT parks, and residential gated societies. The facility manager or RWA officer can use it inside following the same ticket-resolve process.

REFERENCES

1. Meijer, A., & Bolívar, M. P. R. (2016). *Governing the smart city: a review of the literature on smart urban governance*. International Review of Administrative Sciences, 82(2), 392-408.

2. Goldsmith, S., & Crawford, S. (2014). *The Responsive City: Engaging Communities Through Data-Smart Governance*. Jossey-Bass.
3. Scholl, H. J., & AlAwadhi, S. (2016). *Smart governance as key to multi-level urban infrastructure management: The case of the Smart City Vienna*. Information Polity, 21(1), 71–89.
4. FixMyStreet.org. (2020). *FixMyStreet Platform Overview*. Retrieved from https://www.fixmystreet.org
5. Kumar, P., & Sharma, R. (2021). *Role of AI in Urban Infrastructure: A Case Study on Civic Issue Detection Using Image Processing*. Journal of Smart Cities Research, 4(2), 45–53.
6. SeeClickFix. (2022). *Empowering Communities through Civic Engagement Tools*. Retrieved from https://seeclickfix.com/about
7. Sharma, A., & Bansal, V. (2020). *Citizen Participation in Smart Cities: A Gamification Approach to Governance*. International Journal of Information Management, 52, 102069.