International Journal of
AI ELECTRICAL CIVIL AND MECHANICAL ENGINEERING
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0434 www.zesterapublications.com Original Research Paper

Smart Municipality Workforce – Response System

Papishetti Vineela
M. Tech 1st year student in
Sree Dattha Institute of
Engineering and Science
papishettivineela@email.com

Kummari Vijay Kumar
Assistant Professor in Sree
Dattha Institute of Engineering
and Science
vijaykumar.k48@email.com

Abstract - Running a municipality on paper registers and phone-call complaints sounds old-fashioned, but it is still how a lot of local bodies work, and it causes real problems: workers mark attendance for each other, complaints get lost between departments, and nobody can really say whether a reported issue was fixed or just forgotten. In this paper we describe a Smart Municipal Workforce and Public Service Management System that we built to close these gaps in one place instead of patching them one at a time. The platform verifies worker attendance using a live face match against a registered profile combined with the device's location, lets citizens file complaints that are automatically routed to the least-busy supervisor, and turns those complaints into worker tasks that cannot be closed without a before-and-after photo. It also allows citizens or staff to report surplus food which can then be collected by NGOs, and gives administrators a single dashboard to see attendance, tasks, complaints and food collection activity together. We implemented it in Flask, SQLAlchemy, OpenCV and ONNX Runtime on a SQLite backend and tested it with five different user roles. The results show that once attendance, complaints, and tasks are linked with one workflow it is much easier to trace a citizen's complaint all the way through to a verified resolution, something that the single-purpose systems we studied were not really designed to do.

Index Terms - Attendance verification, complaint-to-task workflow, face embedding, municipal governance, role-based access control

I. INTRODUCTION

Surely anyone who has ever lived in a growing town has complained about a pothole, an overflowing bin or a broken streetlight and never known what became of it. Municipal supervisors, on the other hand, are trying to manage field workers scattered across wards, confirm that they actually showed up and did the work, and somehow keep track of dozens of complaints coming in through calls, WhatsApp, or paper forms. When these processes are manual, three things go wrong quite predictably: attendance can be marked by a friend on someone's behalf (commonly called proxy attendance), complaints sit in someone's inbox with no clear owner, and there is no proof that a task marked "done" was actually done properly.

We picked this problem for our project because it is not just a textbook exercise - it is something that affects real people, and it also happens to touch several areas we wanted to get hands-on experience with: web development, databases, computer vision, and role-based systems, all in one application. The idea was simple to state but not entirely simple to build: what if attendance, complaints, task assignment, and even food-surplus donation to NGOs were all handled through one connected system instead of five separate ones?

With that in mind, our objectives for this project were to: (i) verify worker attendance using facial identity together with location, so that proxy attendance becomes much harder; (ii) let citizens submit complaints that get automatically assigned to a supervisor rather than sitting unclaimed; (iii) allow a complaint to be converted into a worker task that can only be closed with photo evidence; (iv) support food-surplus reporting so that NGOs can be looped in when there is food to collect; and (v) give administrators a live dashboard instead of asking them to manually total up spreadsheets.

The rest of the paper follows this structure. Section II looks at existing systems in this space and explains what they do well and where they fall short. Section III walks through how we designed the system and why. Section IV explains the two algorithms that make the accountability features work - face-based attendance matching and workload-based complaint assignment. Section V shows our testing results, including screenshots of the working

International Journal of
AI ELECTRICAL CIVIL AND MECHANICAL ENGINEERING
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0434 www.zesterapublications.com Original Research Paper

application. Section VI wraps up with what we learned and what we would still like to add.

II. LITERATURE REVIEW

Before building anything, we looked at what already exists for attendance tracking, grievance handling, and field-workforce monitoring, mostly through IEEE and Elsevier papers along with a few government case studies. The short version of what we found is that most systems are good at one job and do not really try to do more.

Face-recognition attendance systems are probably the most mature of the group - they compare a live image against a stored template and are quite good at stopping one person from marking attendance for another [1]. The catch is that almost all of them stop at attendance. They do not know what task the worker is supposed to be doing that day, so identity verification and actual task tracking never really meet. GPS-based tracking takes a different angle by confirming that a worker's phone is physically within an assigned zone [3], which is useful for spread-out field teams, but a phone being in the right place says nothing about who is holding it.

On the citizen side, grievance-redressal portals let people submit an issue and get back some kind of tracking number [2]. This is a genuine improvement over walking to an office window, but from what we read, most of these portals show status updates like "Assigned" or "In Progress" without actually showing what was done - there is no photo, no worker name, nothing a citizen could point to as proof. Bigger enterprise complaint-to-work-order systems do connect a complaint to an internal work order, closing part of this gap, but they tend to be built (and priced) for large city corporations rather than smaller municipal bodies or student-built prototypes.

We also looked at two more specific areas relevant to parts of our system. Image-based infrastructure monitoring research uses photos to check things like road damage or garbage accumulation [4], which supports the idea that a photo can be reasonable evidence of a field condition - this is essentially what our before/after task-evidence feature borrows from. Separately, food-rescue platforms connect donors and NGOs using location tagging and status tracking [6], but these operate as their own separate ecosystem, completely disconnected from municipal operations, even though the same supervisors and citizens could plausibly be involved in both.

Putting all of this together, we noticed the same gap kept showing up no matter which type of system we looked at:

- No single platform combines identity-verified attendance with complaint and task management.
- Complaint systems rarely show worker-level, evidence-backed proof of resolution.
- Food-surplus coordination is treated as a separate problem from everyday civic operations.
- Lightweight options suitable for smaller municipal bodies or student projects are scarce.

This is really the gap our project tries to close - not by inventing a new algorithm, but by wiring existing, well-understood techniques (face embeddings, workload balancing, evidence upload) together into one coherent workflow. Table I summarizes how the proposed system compares against the categories above.

TABLE 1. COMPARATIVE SUMMARY OF RELATED SYSTEMS

System	Strength	Limitation
Face-recognition attendance [1]	Prevents proxy marking	No task/complaint link
GPS attendance tracking [3]	Confirms field presence	Cannot verify identity
Grievance portal [2]	Transparent ticketing	No execution evidence
Infrastructure monitoring [4]	Visual verification	Research-stage, isolated
Food redistribution [6]	Donor-NGO coordination	Separate from civic workflow
Proposed system	Integrated, evidence-backed	No offline sync yet

International Journal of
AI ELECTRICAL CIVIL AND MECHANICAL ENGINEERING
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0434 www.zesterapublications.com Original Research Paper

III. SYSTEM ARCHITECTURE AND METHODOLOGY

Instead of building attendance, complaints and food reporting as separate mini-projects and trying to glue them together later, we designed the system as one layered application from the start, so that every module reads and writes to the same database and follows the same authentication rules. This layering is shown in Fig. 1.*A. Architecture Overview*

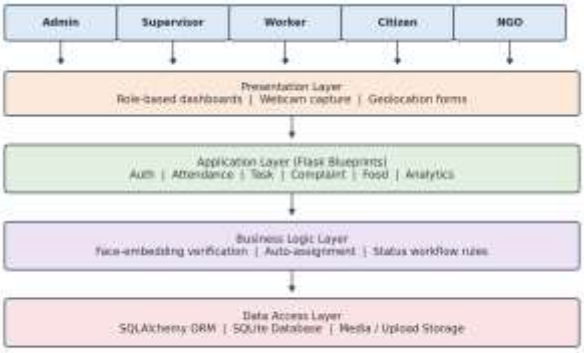


FIGURE 1. LAYERED ARCHITECTURE OF THE PROPOSED SYSTEM

At the top there is a presentation layer of role-specific dashboards, built with Bootstrap and Jinja2 templates, and the browser-side webcam capture and geolocation prompts. Beneath that is the application layer, split into six Flask blueprints: authentication, attendance, task management, complaint management, food reporting, and analytics, so each concern lives in its own file instead of one giant routes file.py. The business-logic layer contains the actual decision code, such as the face-embedding comparison and complaint-assignment rule discussed in Section IV. The data-access layer persists everything via SQL Alchemy models backed by SQLite.

Five roles use the system day to day. Admins control the overall user hierarchy and observe the analytics dashboard; supervisors are in charge of a group of workers, translate the incoming complaints to tasks and check the completed work; workers register a face profile once and then mark the attendance and complete the tasks; citizens submit complaints and food-surplus reports; and NGOs take food collection tasks. Each of these roles is enforced by login-based access checks, so a worker account, for example, cannot reach the admin analytics routes even if the URL is typed directly.

B. Workflow Design

The part we spent the most time getting right was not any single screen, but the chain connecting them. A complaint filed by a citizen is auto-assigned to whichever supervisor currently has the fewest open complaints. That supervisor can convert the complaint into a task and hand it to a specific worker. The worker updates the task's status as it moves along, and before the task can be marked complete, the system requires both a "before" photo and an "after" photo. Only once the supervisor has reviewed this evidence and accepted it does the linked complaint actually get marked resolved. If the evidence is not convincing, the supervisor can send the task back instead of rubber-stamping it. This is the specific design decision meant to fix the biggest gap we found in Section II - a citizen complaint should not just change status quietly in a database; it should be backed by something a human actually reviewed.

C. Database Design

The schema is intentionally kept to five core tables - Users, Attendance, Task, Complaint, and Food Report - connected through foreign keys rather than being split into many small normalised tables. The Users table stores a manager_id column so that a worker can be linked to the supervisor who oversees them, and the same table stores the registered face embedding as JSON for workers. The Task table optionally stores a complaint_id, which is what creates the traceable link described above. Keeping the schema small made it much easier to reason about when we were debugging cross-module workflows, at the cost of a few slightly wider tables than a textbook-normalised design might use.

IV. IMPLEMENTATION

The application is written in Python with Flask 2.3, Flask-Login for sessions and Flask-SQLAlchemy for the ORM. Bootstrap 5 and Jinja2 on the front end. Two pieces of logic matter most for the accountability goals of the project, and we describe both below.

A. Face-Embedding Attendance Verification

When a worker registers their face, the captured image is passed through face detection, cropped to the facial region, resized to the model's expected input size, and then run through an ONNX Runtime session that outputs a normalised embedding vector - essentially a list of numbers that represents that face. When the same worker later marks attendance, a fresh live image goes through the same steps to produce a second embedding. Because both vectors are normalised to unit length, cosine similarity is simply their dot product:

International Journal of
AI ELECTRICAL CIVIL AND MECHANICAL ENGINEERING
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0434 www.zesterapublications.com Original Research Paper

$$sim(R, C) = R \cdot C = \Sigma R_i C_i \quad (1)$$

The system accepts the attendance if the similarity score is >= 0.6, otherwise it rejects it and asks the worker to try again with better lighting or a clearer angle. We picked 0.6 after some manual trial and error - lower thresholds let mismatched faces through, while higher thresholds occasionally rejected the correct worker under poor lighting. The device's GPS coordinates, captured through the browser's geolocation API, are stored with every attendance record so a supervisor can later cross-check where attendance was actually marked.

B. Workload-Based Complaint Assignment

We did not want complaints sitting unassigned and we did not want one supervisor to be overloaded while another had nothing to do. So for a new complaint, the system looks at the number of active complaints (any not already Resolved or Rejected) that each supervisor has in the system, and assigns the new complaint to whoever has the lowest count:

$$s^* = arg\ min\ c(s),\ s \in S \quad (2)$$

Ties are broken alphabetically by supervisor name, mainly so that the behaviour stays predictable and testable rather than picking randomly. This is a fairly simple rule - nothing close to a scheduling algorithm from an operations-research textbook - but it was enough to keep the workload roughly even across the supervisor accounts we tested with, and it is easy for anyone to audit by just counting complaints per supervisor.

C. Technology Stack

Media files - complaint photos, voice notes, and task before/after images - are saved to a structured upload folder on disk, with only the file path stored in the database. We chose this over storing binary blobs directly in SQLite because it kept the database small and made it much easier to preview images directly in templates using a simple static file URL.

D. Testing Strategy

Because most of the risk in this project was cross-module (a bug in complaint-to-task conversion, for example, would not show up if we only tested the complaint form on its own), we tested at three levels. Unit- completion without an after-image. Integration tests specifically checked that converting a complaint into a task updated the complaint's status correctly, and that a supervisor accepting a completed task correctly flipped the linked complaint to Resolved. System-level tests then walked through each role's full journey - login, attendance, complaint submission, task handling, and food collection - end to end.

V. RESULTS AND DISCUSSION

We ran the application locally with a seeded database containing all five roles and followed the same journeys as described in Section IV.D. All critical paths tested worked without blocking defect, although a handful of edge cases (mostly around poor lighting during face capture) required retries .

TABLE 2. OBSERVED RESPONSE TIMES FOR CORE OPERATIONS

Operation	Average response
Login	< 1 s
Complaint submission	< 2 s
Attendance marking	< 3 s (incl. face check)
Dashboard load	< 2 s

As Table II shows, response times stayed well within what feels "instant" for a user clicking around a form - the slowest single operation was attendance marking, which makes sense since it is the only one that runs an on-the-fly embedding comparison rather than a plain database query.

A. Application Screenshots

The figures below show the working application. Fig. 2 is the login screen, and from which each role is redirected to its own dashboard. Fig. 3 shows the citizen-facing complaint screen where a complaint can be submitted with an optional photo or voice note and its status can be tracked afterwards. Fig. 4 shows the attendance screen used for face-based check-in. Fig. 5 shows the attendance records view available to supervisors and admins, including the manual override option for workers without a registered profile. Fig. 6 shows the administrator's analytics dashboard, which is where user counts, task status, complaint status, and food-coordination numbers are pulled together in one place.



FIGURE 2. LOGIN INTERFACE

International Journal of
AI ELECTRICAL CIVIL AND MECHANICAL ENGINEERING
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0434 www.zesterapublications.com Original Research Paper

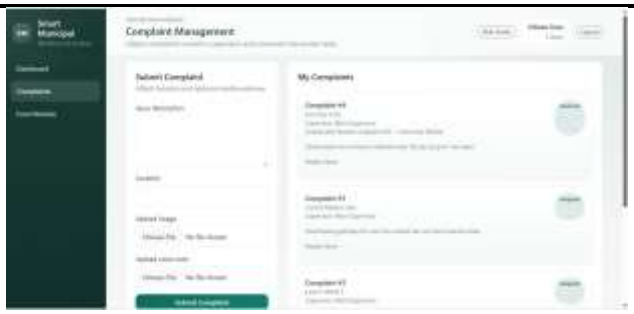


FIGURE 3. CITIZEN COMPLAINT SUBMISSION AND TRACKING INTERFACE

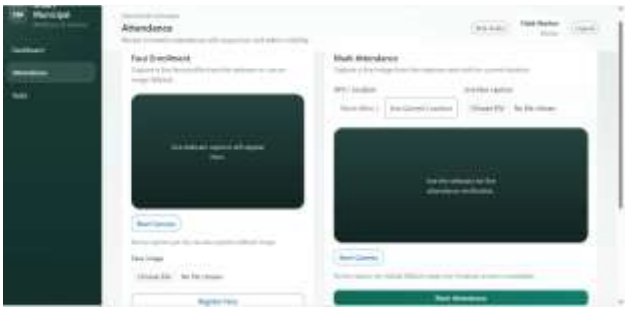


FIGURE 4. ATTENDANCE MARKING INTERFACE WITH FACE CAPTURE



FIGURE 5. ATTENDANCE RECORDS AND SUPERVISOR OVERRIDE VIEW



FIGURE 6. ADMINISTRATOR ANALYTICS DASHBOARD

B. Discussion

A few things stood out once the system was actually running rather than just designed on paper. Role-based dashboards made navigation much less confusing for testers who were not familiar with the project - each person only saw buttons relevant to what they were supposed to do, which cut down on the "where do I click" questions we got during informal walkthroughs with classmates. The complaint-to-task linkage also worked as intended: once a supervisor converted a complaint into a task, the citizen could see the status change on their own end without anyone manually updating two separate records, which was the main thing we set out to fix.

On the downside, face verification was significantly less reliable in dim indoor lighting, which is consistent with what the face-recognition literature in Section II already warns about. We do not consider this fully solved in the current version - it is listed as future work in Section VI. The system also currently depends on a live internet connection throughout, so it would not hold up well in areas with patchy connectivity, which is a realistic concern for some municipal field zones.

Taken together, we think these results support the central idea behind this paper: connecting attendance, complaints, tasks, and food coordination through a single workflow closes a real gap left by the single-purpose systems reviewed earlier, and it does this without adding noticeable performance overhead for users.

VI. CONCLUSION AND FUTURE SCOPE

This paper presented a Smart Municipal Workforce and Public Service Management System that ties together face-verified attendance, workload-balanced complaint routing, evidence-backed task closure, food-surplus coordination, and administrative analytics inside one role-based platform. Across the five roles we tested, the system kept a consistent, traceable path from a citizen's complaint all the way to a verified field resolution, which was the specific gap we set out to close after reviewing existing attendance and grievance systems.

There is still a fair amount we would like to add given more time. Automatic deadline-based escalation would flag tasks that are running late without waiting for a supervisor to notice manually. Real notifications - SMS, email, or push - would replace the current dashboard-only visibility, which assumes someone remembers to check. Offline-first support for attendance and task-evidence capture would help in areas with unreliable connectivity, syncing once a connection is available again. And on the AI side, automatically comparing before/after task images for visible change, instead of relying purely on manual supervisor review, could reduce review workload as the system scales to more workers.

Beyond this specific project, we think the general approach - identity-verified attendance plus evidence-backed task closure plus citizen-facing status tracking, all

International Journal of
AI ELECTRICAL CIVIL AND MECHANICAL ENGINEERING
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0434 www.zesterapublications.com Original Research Paper

in one schema - could reasonably apply to sanitation departments, public-works maintenance crews, ward-level grievance cells, and community food-donation drives, since all of these already involve the same basic ingredients: a field worker, a supervisor, and someone reporting an issue.

REFERENCES

[1] S. Sharma and R. Gupta, "Face recognition based attendance system using deep learning," in Proc. IEEE Int. Conf. Computing, Communication and Automation (ICCCA), 2024, pp. 234-239.

[2] P. Kumar, A. Singh, and M. Verma, "Smart grievance redressal system for urban governance using machine learning," J. Urban Technol., vol. 31, no. 2, pp. 45-62, 2023.

[3] N. Patel and S. Desai, "GPS-based employee tracking and attendance system for distributed workforce management," IEEE Trans. Intell. Transp. Syst., vol. 25, no. 3, pp. 2456-2470, 2024.

[4] R. Mishra, K. Singh, and A. Reddy, "Image-based infrastructure monitoring using deep convolutional neural networks for municipal asset management," Autom. Constr., vol. 152, pp. 104-118, 2023.

[5] V. Sharma, L. Gupta, and P. Mehta, "Smart waste management and monitoring in urban environments using IoT and cloud computing," Sustain. Cities Soc., vol. 98, pp. 104-122, 2024.

[6] T. Chen, H. Wang, and Y. Zhang, "Food redistribution platforms for reducing urban food waste: A systematic review and framework," Resource. Conserve. Recycle., vol. 188, pp. 106-121, 2023.

[7] M. Fernandez and J. Kim, "Role-based dashboard systems for e-governance monitoring: A comparative analysis of municipal platforms," Gov. Inf. Q., vol. 41, no. 1, pp. 101-115, 2024.

[8] F. Schroff, D. Kalenichenko, and J. Philbin, "Face Net: A unified embedding for face recognition and clustering," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815-823.

[9] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arc Face: Additive angular margin loss for deep face recognition," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2019, pp. 4690-4699.

[10] Flask Documentation, "Flask web development documentation," Pallets Projects, 2024. [Online]. Available: https://flask.palletsprojects.com/

[11] ONNX Runtime Documentation, "ONNX Runtime overview," Microsoft, 2024. [Online]. Available: https://onnxruntime.ai/