IT-314: Software Engineering

Prof. – Saurabh Tiwari

Project:

Crowd-Powered Complaint Management System

Group: 6

Name: Jainil Shailesh Jagtap

Student ID: 202301032

End-Users/Stakeholders:

1. Citizens-
    - Submit, track, and upvote urban issues via the app/web platform.
    - Expectations: Intuitive interface, real-time updates, multilingual support, privacy assurance.
2. Municipal Departments (Government Authority)-
    - Receive, analyse, and resolve reported issues using the centralized dashboard.
    - Expectations: Efficient routing of issues, data-driven prioritization, integration with existing systems.
3. System Administrator-
    - Managing the overall working of the application.

Elicitation Techniques:

Stakeholder Name	**Elicitation Techniques**	**Justification for Choosing the Technique**
Citizens (Primary End Users)	**Surveys** + **Prototype Testing**	Surveys quickly gather quantitative data on interface preferences, privacy concerns, and language needs. Prototype testing allows real-world feedback on usability and AI categorization accuracy.
Municipal Departments	**Interviews** + **W orkshops**	Interviews clarify workflow integration, AI categorization challenges, and training needs. Workshops align departments on

		issue prioritization logic and dashboard usability.
System Administrators	**Interviews, Document Analysis, Workshops**	Technical interviews ensure secure backend needs, document analysis identifies privacy and scalability constraints, and workshops elicit robust solutions for AI analytics, reliable maintenance, and federated learning implementation.

Functional Requirements:

Citizens

1. People can use a phone or web app to report problems like potholes or broken lights, adding photos, locations, and descriptions.
2. Users can check updates on their reports, like if the issue is being worked on or fixed, with expected finish times.
3. People can vote up issues to show they're important, helping the city decide what to fix first.
4. Users can rate and comment on how well the city fixed their reported issue.

Municipal Departments

1. City workers get a website showing all reported issues, sorted by AI based on type, place, or urgency.

2. The system sends issues to the right department, like potholes to road crews, using AI.
3. The system makes reports on common problems, like where potholes keep popping up, to plan better.
4. Workers can post updates and photos about fixes, which people can see right away.

System Administrators

1. Admins use a dashboard to watch how the system runs and get alerts if something's wrong.
2. Admins can tweak or retrain the AI to sort issues better, keeping data private with special tech.
3. Admins handle user accounts for people and city workers, setting roles or fixing issues.
4. Admins set rules to keep user data safe and follow privacy laws.
5. Admins schedule system fixes or updates without messing up the app for users.

Non-Functional Requirements:

1. **Easy to Use**: The app and dashboard are simple, so anyone can learn them in a few minutes.
2. **Handles Growth**: The system can manage thousands of reports without slowing down.
3. **Fast**: The app loads in under 2 seconds on phones, and AI updates finish in a day.
4. **Safe**: Only the right people see data, with strong passwords and locked-up info.

5. **Always On**: The system works 99% of the time, with quick fixes if it goes down.
6. **Easy to Fix**: The system is built in parts, so updates don't break everything.
7. **Works with Others**: It connects to city systems like maps or databases smoothly.

Domain Requirements:

1. The system uses a clear list of problem types (like roads or trash) for AI sorting.
2. Reports need exact locations (within 5 meters) to send to the right city team.
3. The system follows city and data privacy rules, like keeping info safe.
4. It knows which areas belong to which city department for correct routing.
5. The AI is super accurate (90%+ for sorting issues) and follows standard tech rules.
6. Only real users can vote on issues to keep things honest.

User Stories:

Citizens

User Story 1

Front: As a citizen, I want to report potholes with photos so that the city fixes them quickly.

Back:
Acceptance Criteria:

- Success: User submits report with photo, geofencing adds location, system confirms receipt, routed to public works.
- Failure: Submission fails if photo exceeds 10MB or geofencing is disabled.

User Story 2

Front: As a citizen, I want to track my report's status so that I know when it's fixed.
Back:
Acceptance Criteria:

- Success: User views real-time updates (e.g., "in progress") with estimated completion time.
- Failure: Status doesn't update if department fails to log progress.

User Story 3

Front: As a citizen, I want to upvote urgent issues so that the city prioritizes them.
Back:
Acceptance Criteria:

- Success: Verified user upvotes issue, increasing its priority score in the system.
- Failure: Unverified users or spam accounts cannot upvote, showing an error.

User Story 4

Front: As a citizen, I want the app in my language so that I can use it easily.
Back:
Acceptance Criteria:

- Success: User selects language (e.g., Spanish), and app interface fully translates.

- Failure: Interface remains in default language if selected language isn't supported.

User Story 5

Front: As a citizen, I want to rate fixed issues so that I can share feedback.
Back:
Acceptance Criteria:

- Success: User submits 1-5 star rating and comment after issue resolution.
- Failure: Rating fails if issue isn't marked resolved by department.

Municipal Departments

User Story 1

Front: As a city worker, I want a dashboard of issues so that I can manage them.
Back:
Acceptance Criteria:

- Success: Dashboard shows AI-sorted issues by type and urgency, filterable by status.
- Failure: Dashboard fails to load if server is down or filters are broken.

User Story 2

Front: As a city worker, I want issues routed to my team so that we can act fast.
Back:
Acceptance Criteria:

- Success: AI routes pothole reports to public works using geofencing location data.
- Failure: Routing fails if AI misclassifies issue or geofencing data is inaccurate.

User Story 3

Front: As a city worker, I want to assign workers to issues so that we fix them efficiently.
Back:
Acceptance Criteria:

- Success: Worker assigns team and timeline to issue, updating status.
- Failure: Assignment fails if worker lacks permission or system crashes.

User Story 4

Front: As a city worker, I want reports on issue trends so that we can plan better.
Back:
Acceptance Criteria:

- Success: System generates report showing frequent pothole locations monthly.
- Failure: Report is inaccurate if geofencing data or AI analysis fails.

User Story 5

Front: As a city worker, I want to post updates so that citizens know progress.
Back:
Acceptance Criteria:

- Success: Worker posts status update and photo, visible to users instantly.
- Failure: Update doesn't show if photo upload fails or system is offline.

System Administrators

User Story 1

Front: As an admin, I want to monitor system health so that I can fix issues.
Back:
Acceptance Criteria:

- Success: Dashboard shows server load and alerts for high latency (>2s).
- Failure: Alerts don't trigger if monitoring system is misconfigured.

User Story 2

Front: As an admin, I want to retrain AI models so that issue sorting improves.
Back:
Acceptance Criteria:

- Success: Admin retrains AI with federated learning, improving accuracy to >90%.
- Failure: Retraining fails if geofencing data is insufficient or privacy settings are violated.

User Story 3

Front: As an admin, I want to manage user accounts so that only authorized people access.
Back:
Acceptance Criteria:

- Success: Admin suspends or assigns roles to user accounts successfully.
- Failure: Action fails if admin lacks permission or database is locked.

User Story 4

Front: As an admin, I want to set privacy rules so that user data stays safe.
Back:
Acceptance Criteria:

- Success: Admin sets data anonymization for geofencing, complying with GDPR/CCPA.
- Failure: Settings don't apply if configuration conflicts with existing rules.

User Story 5

Front: As an admin, I want to schedule updates so that the system stays reliable.
Back:
Acceptance Criteria:

- Success: Admin schedules update, notifies users, and system runs smoothly.
- Failure: Update disrupts service if scheduling conflicts or notification fails.