TRIBHUVAN UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY
CENTRAL CAMPUS OF TECHNOLOGY (CCT), HATTISAR, DHARAN

A Project Proposal On

UrbanFix: An Intelligent E-Governance System for Civic Issue Management

Submitted To:
DEPARTMENT OF CSIT/IT
CENTRAL CAMPUS OF TECHNOLOGY (CCT), HATTISAR, DHARAN, NEPAL

In partial fulfilment of the requirements for the degree of Bachelor's of Science in Computer Science and Information Technology (B.Sc. CSIT)

Submitted By:
Suyash Dhakal (29617)
Salan Katuwal (29607)
Jitendra Shah (29595)

June, 2025

TABLE OF CONTENTS

List of Figures

List of Tables

1. Introduction

In today's digital age, efficient communication between citizens and local government bodies is crucial for addressing community-level issues and ensuring good governance. Traditional reporting mechanisms often suffer from delays, slow, and lacks transparency. Limited accessibility, and lack of transparencyfactors that hinder citizen engagement and administrative accountability [1].

This project proposes the development of a web-based E-Governance System UrbanFix-designed to streamline the issue reporting process and enhance civic participation. The platform empowers registered users to report local problems to their respective ward offices by providing a title, description, and location, attaching images, and either selecting or automatically predicating a category using a multi-class classification algorithm [2]. To maintain integrity and prevent duplicate submissions, a cosine similarity algorithm compares new reports against existing ones within the same ward and category [3].

Submitted issues enter a 'verifying' stage, where ward head admins review and validate the report before marking it as 'verified'. Guests (unregistered users) can still browse reported issues, view details, and explore a map that displays all verified issues to ensure public transparency. UrbanFix Features interactive dashboards for both users and administrators, showing real-time issue statues, visual analytics, and a "Hall of Fame" to recognize active contributors. By leveraging machine learning, similarity detection, and real-time analytics for both administrators and citizens, the system enhances transparency, accountability, and data-driven governance—supporting SDG 11 by enabling safer, more inclusive, resilient, and sustainable cities through improved civic issue reporting and active citizen participation [4].

2. Problem Statement

Urban communities often face persistent issues like potholes, unmanaged waste piling up in public areas, waterlogged streets during monsoon, yet traditional reporting systems are slow, non-transparent, and offer minimal citizen involvement. The lack of a centralized digital platform leads to delays, poor tracking, and limited accountability from local authorities. Citizens are often unaware if their complaints are acted upon, and there is no way to engage in follow-up or community discussion [1].

Moreover, there is typically no system in place to check whether a reported issue already exists, leading to duplicate entries that burden administrative workflows [3].There is also a noticeable absence of features that promote civic dialogue and community participation in verified problems. These limitations call for an intelligent, transparent, and interactive solution that not only improve reporting but also encourage feedback and collaboration among users and local authorities.

3. Objectives

1. To develop a web platform that enables citizens to report urban issues with title, description, ward, category, image, and geo-location (pinpointing the location on a map).
2. To detect and block duplicate reports using semantic similarity to reduce redundancy.
3. To implement a multi-class text classification model using SpaCy embeddings and Logistic Regression to automatically suggest appropriate issue categories.
4. To design separate dashboards for users and ward admins that provide real-time issue statuses, verified problems on maps, and analytics for transparency.

4. Methodology

4.1. Requirement identification

4.1.1. Literature Review

E-participation platforms facilates direct engagement between citizens and local governance bodies. Platforms like "FixMyStreet" in the United Kingdom allows citizens to report local issues, such as pot-holes or broken streetlights, directly to their local authorities. This real-time reporting improves citizen participation and streamlines the resolution process, showcasing the potential of technology to address everyday concerns [5].Other platforms like SeeClickFix (USA), and 311 NYC (USA) also allow citizens to report local problems such as potholes, graffiti, and broken streetlight via web or mobile apps. This system often incorporate geolocation to tag issues and provide updates to users. SeeClickFix offers these functionalities along with analytics dashboards for city administrators, enabling them to track and respond complaints more effectively [6].The 311 NYC system integrates multiple city services into a unified portal, allowing citizens to report various problems and track their resolution status online [7]. However, many of these systems still rely on manual classification of issue types. Based on reviewed implementations, most do not incorporate advanced machine learning techniques for categorizing complaints. Additionally, many lack automated tools for detecting duplicate reports—features that could significantly enhance the accuracy and efficiency of issue tracking [3].

Duplicate submissions in civic issue reporting platforms burden municipal systems and reduce efficiency. Unlike exiting platforms, our project uses a multi-class machine learning classifier to predict the category of reported problems automatically, saving administrative time and improving classification accuracy. Cosine similarity has shown effectiveness in identifying duplicates by comparing textual records [3]. In this system, a similarity threshold (e.g., 70%) is applied to complaint descriptions to prompt user confirmation and prevent redundant submissions. Our system enforces email-based user registration, preventing spam and ensuring accountability, while guests can only view reports and statistics.

4.1.2. Requirement Analysis

- Functional Requirements
  - User Authentication: User register and log in via email verification.

- Role-Based Access Control: Roles include Guest, Registered User, and Ward Admin, each with defined permissions.
- Issue Reporting: Authenticated users submit issues with title, category (manual or predicted), description, image, ward, and location.
- Category Prediction: Spacy-based multi-class model suggests relevant categories during submission.
- Duplicate Detection: Cosine similarity checks compare submissions with existing issues (same ward and category), showing top 3 similar issues ($> 70\%$).

- Non-Functional Requirements
  - Usability: Intuitive and accessible UI for users with basic digital literacy.
  - Performance: Supports concurrent user activity with minimal latency.
  - Security: Ensure data protection and access control for sensitive operation.
  - Scalability: Designed to support additional wards, users, and data volume growth.
  - Availability: Ensures 99% uptime with minimal disruptions.
  - Transparency: Status indicators, public access to verified issues, and visible moderation workflow.

- Software Requirements
  - Frontend: React.js, Tailwind CSS
  - Backend:
    1. Main API: Node.js, Express.js
    2. Machine Learning Microservice: Python, Flask (using SpaCy + Logistic Regression)
  - Database: MongoDB
  - Machine Learning: SpaCy, Logistic Regression, OpenAI Embedding API
  - Other Tools: GitHub, Postman, Map API (like Leaflet or Google Maps),
  - IDE: VS code

- Hardware Requirements
  - Client-side:
    * Minimum 2 GHz dual-core processor

    - 4 GB RAM
    - Internet-enabled device (PC/Laptop/smartphone)
  - Server-Side:
    - Minimum 2-core CPU
    - 4 GB RAM
    - Cloud hosting or local server for deployment and testing
    - Minimum 50 GB storage
    - Internet connectivity with high Uptime

4.2. Feasibility Study

4.2.1. Technical Feasibility

The proposed system, is technically feasible. The development leverages modern, open-source technologies such as React.js and Tailwind CSS for the frontend, Node.js with Express.js for the backend, and MongoDB for database management. Machine learning features are supported using Python, Spacy, and OpenAI's Embedding API for classification and similarity detection. For geolocation and map features, Leaflet.js integrated with OpenStreetMap provides a free and customizable mapping solution. These tools are widely supported, cost-effective, and suitable for scalable web applications. The development team possesses the necessary progamming skills and access to relevant development environments (e.g., VS Code, Postman, GitHub). Thus, based on the availability of tools, platforms, and expertise, the system is considered technically viable.

4.2.2. Operational Feasibility

The proposed system is designed with a user-friendly interface suitable for citizens with basic digital literacy and ward officials familiar with basic web tools. The role-based access control ensures that each user interacts only with relevant features, enhancing usability and minimizing confusion. Organizational support is expected through the involvement of ward-level administrators, who are assumed to be trained or trainable in basic dashboard usage. The system does not rely on integration with legacy systems, making it easier to implement without compatibility issues. With appropriate orientation and support, the system is expected to be accepted and effectively used by both citizens and ward officials.

4.2.3. Economic Feasibility

The economic feasibility of the proposed system project assesses/evaluate whether the system can be developed and maintained at a reasonable cost while delivering significant value to its stakeholders. Since this is a student-led academic project, the development primarily leverages/use of open-source technologies and voluntary contributions from the development time. As a result, overall costs are relatively low, while the potential benefits-especially when scaled to real-world use by local governments-are considerable. The economic feasibility of proposed system is shown in table below.

Table 4.1: Cost-Benefit Analysis of the Proposed Project

Item	**Description**	**Cost (NPR)**	**Benefit (NPR)**
Development Costs	Software Development	50,000	-
Hardware Costs	Server/Hosting fees	30,000	-
Training Costs	User and admin training	15,000	-
Maintenance Costs	Annual Maintenance	10,000	-
API Usage Costs	OpenAI embedding API	218	-
Total Costs		**1,05,218**	-
Increased Efficiency	Time Savings	-	3,00,000
Total Benefits		-	**3,00,000**
Net Benefit			**1,94,728**

The economic feasibility of "UrbanFix" is strong. The low upfront and maintenance costs, coupled with meaningful social benefits and future scalability, make this system a viable and impactful solution for improving local problem reporting and governance in Nepal.

4.2.4. Schedule Feasibility

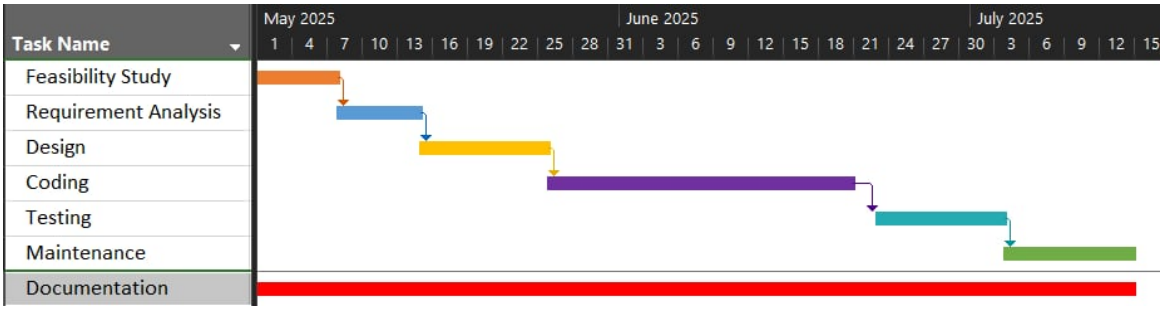

Figure 4.1: Gantt Chart Demonstrating Schedule Feasibility

4.3. High-Level Design of System

4.3.1. Methodology of the proposed system

The proposed system follows an Agile Prototyping Hybrid Model. Development begins with a basic working prototype to validate core functionality and collect feedback. After that, the team adopts Agile sprints to implement each feature, refine it based on feedback, and ensure continuous improvement. This hybrid approach promotes flexibility, early validation, and user-focused development.

The system architecture is monolithic, where the frontend and backend components are tightly integrated for simplicity and ease of deployment. The backend adheres to RESTful API principles, ensuring clear and standardized communication between the client and server using HTTP methods (GET, POST, PUT, DELETE).

While the core backend is built using Node.js with Express.js, a dedicated Flask-based microservice is deployed for handling category prediction. When the user fills the issue title, a POST request is triggered (onBlur) to a Flask service running a SpaCy + Logistic Regression model, which returns suggested categories based on the title.

4.3.2. Flow Charts/Working Mechanism of Proposed System

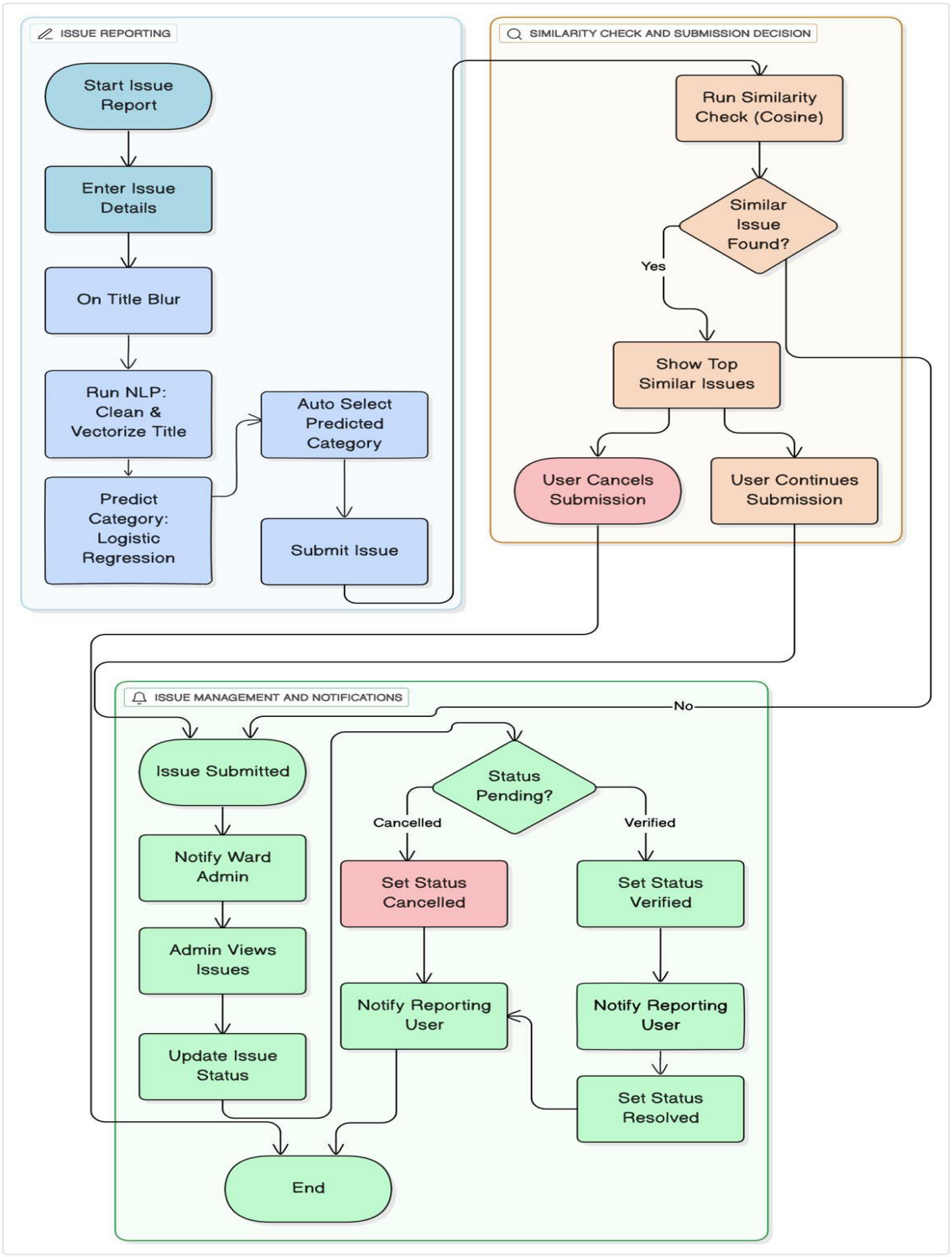

Figure 4.2: Flowchart of the proposed system

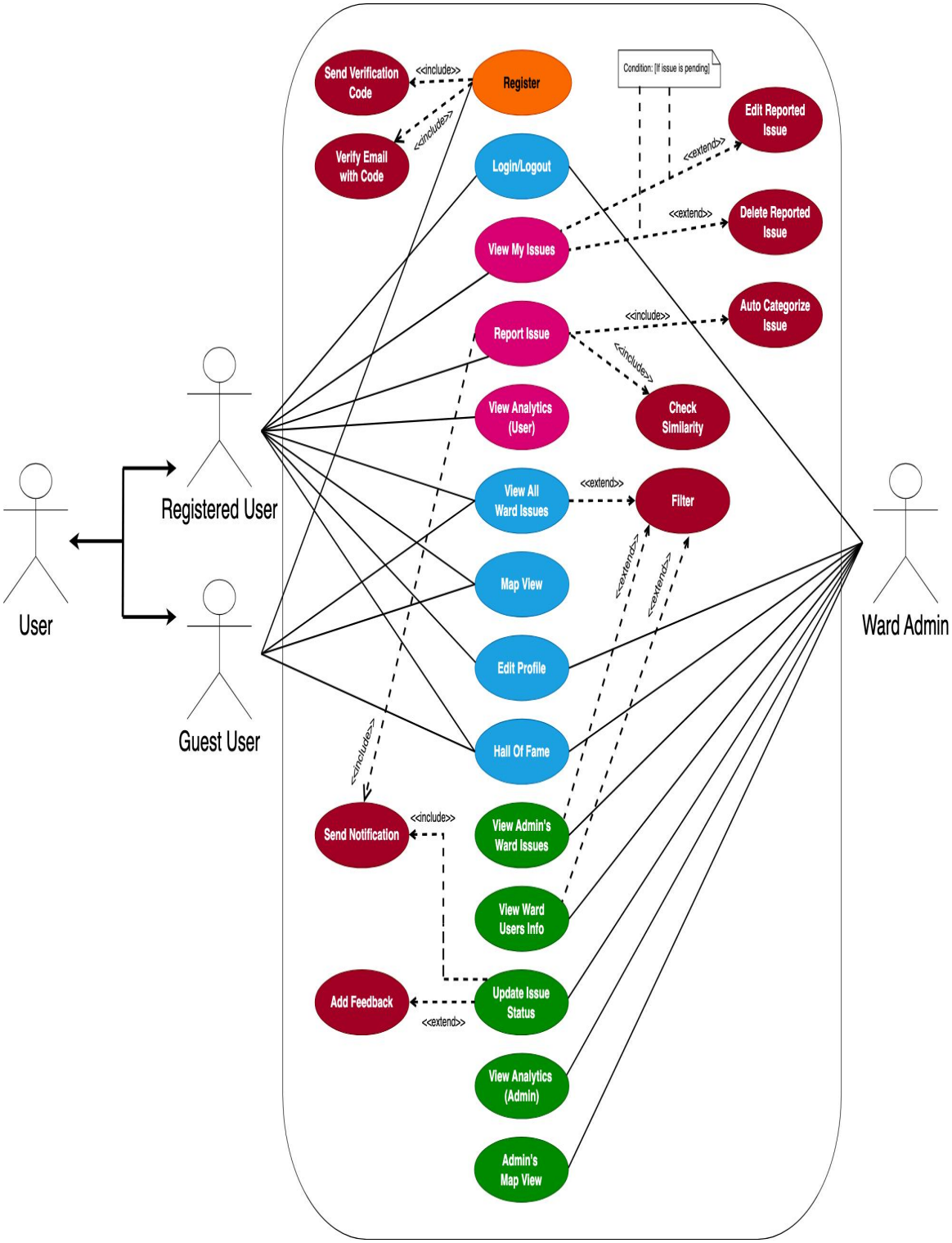

Figure 4.3: Use Case Diagram of the proposed system

4.3.3. Description of Algorithms

- Cosine Similarity for Duplicate Problem Cosine Detection

- Purpose: To prevent users from submitting issues that are already reported in the system, thereby maintaining data quality, reducing redundancy, and improving system efficiency.
- Mechanism:
  1. When a user submits a new issue (title + description), the text is passed to OpenAI's text-embedding-3-large model, which returns a high-dimensional vector representing its semantic content [8].
  2. The backend applies a blocking technique to filter existing issues based on the same ward and category, narrowing down the comparison scope.
  3. For each of these filtered issues, their embeddings are also obtained by passing their text (title + description) to the same OpenAI model.
  4. The system computes cosine similarity between the new issue's embedding and each existing one using the formula:
     "cosine similarity(A, B) = (A · B) / (‖A‖ * ‖B‖)"
  5. If any similarity score is greater than or equal to 70%, the user is prompted: "This issue appears similar to an existing report. Do you still want to continue?"
- Threshold Justification: A threshold of 70% was chosen as a balanced default. A higher value (e.g. greater than 90%) may fail to detect semantically similar issues described differently, while a lower value (e.g. less than 50%) risks false positives. This threshold can be fine-tuned over time based on user behavior and feedback, such as how often users agree with or reject the duplicate suggestions.
- Contribution: This algorithm significantly reduces duplicate submissions, improves report accuracy, and enhances the user experience by introducing intelligent issue intake.

• Category Classification Using Word Embeddings and Logistic Registration

- Purpose: Automatically categorize user-submitted issues (e.g., "road damage", "waste disposal", "lighting") to improve data labeling and assist administrators using word embeddings which have emerged as a powerful technique for text classification, offering advantages over traditional methods like Bag-of-Words.These embeddings capture semantic relationships between words and have shown state-of-the-art results in various natural language processing tasks [9].
- Mechanism:
  1. The titles of reported problems are extracted and preprocessed.

2. SpaCy's en_core_web_lg model is used to vectorize words in the titles of reported problems, enabling the creation of feature vectors that can be used for training a machine learning model.SpaCy can use complex neural network-based models to implement natural language processing components. These components achieve the most advanced results for many tasks and have the advantage of integrating word vectors [10].
3. The dataset is split into training and test sets.
4. A Logistic Regression model is trained using these averaged vectors.

– Contribution: This model helps reduce manual effort and ensures consistent tagging of issues. It enhances the efficiency of the admin dashboard and enables better analytics for governance performance.

- Bcrypt for Secure Password Storage

  – Purpose: To protect user passwords stored in MongoDB, ensuring password confidentiality even if the database is compromised.

  – Mechanism:

    1. During user registration, bcrypt generates a unique cryptographic salt and hashes the password using a configurable work factor producing 60 character hash.
    2. The resulting hash is stored in MongoDB.
    3. During login, the provided password is hashed using the same salt and compared with the stored hash for verification.

  – Contribution: Bcrypt significantly enhances credential security by preventing reverse engineering of passwords through brute-force or rainbow table attacks. This is a critical foundation for secure authentication and JWT issuance in systems managing sensitive data, reinforcing user trust and system integrity.

- JSON Web Token (JWT) for authentication

  – Purpose: To implement secure, stateless authentication and authorization across the platform for multiple user roles-citizens, ward administrators, and system administrators.

  – Mechanism:

    1. Upon successful login, the system issues a JWT (JSON Web Token) that includes:
       (a) Header: Specifies the signing algorithm (e.g., HS256).

(b) Payload: Encodes user-specific claims such as
    i. User_id
    ii. Role (e.g., citizen, ward_admin,sys_admin)
    iii. Exp (token expiration timestamp)
(c) Signature: A cryptographic hash generated using a secret key to ensure the token's integrity and authenticity.

2. The client stores the token (typically in local storage or cookies) and includes it in the Authorization header as a Bearer token for subsequent API requests.
3. Each protected API endpoint validates the JWT:
    (a) Verifies the signature using the server-side secret key.
    (b) Checks for token expiration and correct role claims.

– Contribution: JWT enables secure, role-based access control without maintaining session state on the server. It protects sensitive endpoints from unauthorized access, boosts system transparency, and enhances user trust by ensuring authenticated interactions.

5. Expected Output

The successful implementation of UrbanFix is expected to deliver a functional, user-friendly web-based e-governance platforms that:

- Enables citizens to report local issues with location, image, and descriptive details.
- Uses email-based authentication to ensure user accountability.
- Implements machine learning for automatic issue category classification to improve administrative efficiency.
- Integrates cosine similarity-based detection to prevent duplicate reports and maintain data integrity.
- Provides ward-level admin dashboards for verifying, updating, and resolving reported issues.
- Promotes transparency, civic participation, and data-driven governance at the local level.

References

[1] B. Aryal, B. Poudel, J. Karki, N. Bhattarai, and S. Bhandari, "Expectation of the communication mechanism and its gap in public administration: Approaches to communication between local government and public," *Nepal Journal of Multidisciplinary Research (NJMR)*, p. 109–126, 06 2024.

[2] C. Bishop, M, *Pattern Recognition and Machine Learning*. Springer New York, NY, 01 2007.

[3] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 1, pp. 1–16, 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4016511

[4] U. Nations, "Goal 11 — make cities and human settlements inclusive, safe, resilient and sustainable," 2025. [Online]. Available: https://sdgs.un.org/goals/goal11

[5] J. Bhanye and R. Shayamunda, *The Promise of Civic-Tech: Digital Technologies and Transparent, Accountable Governance*, 02 2025, p. 101.

[6] SeeClickFix. [Online]. Available: https://seeclickfix.com/

[7] N. 311. [Online]. Available: https://portal.311.nyc.gov/

[8] OpenAI, "Text embedding models," *OpenAI Platform Documentation*, 2024. [Online]. Available: https://platform.openai.com/docs/guides/embeddings

[9] M. N. Helaskar and S. S. Sonawane, "Text classification using word embeddings," *2019 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA)*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:220311033

[10] C. Hu, H. Gong, and Y. He, "Data driven identification of international cutting edge science and technologies using spacy," *PLOS ONE*, 10 2022. [Online]. Available: https://doi.org/10.1371/journal.pone.0275872