iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** IX **Month of publication:** September 2025

DOI: https://doi.org/10.22214/ijraset.2025.73975

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue IX Sep 2025- Available at www.ijraset.com

Predictive Visualization of Community Municipal Challenges

Somaraju Tharun[1], G. Praveen Babu[2]
[1]*Post Graduate Student, MCA, Department of Information Technology, Jawaharlal Nehru Technological University, Hyderabad, India*
[2]*Associate Professor, Department of Information Technology, Jawaharlal Nehru Technological University, Hyderabad, India*

Abstract: ***The system is an advanced web-based solution designed to simplify the process of reporting, analyzing, and predicting urban civic problems. It addresses recurring challenges such as road damage, drainage leakage, water overflow, garbage accumulation, electricity outages, and borewell malfunctions. Citizens can submit detailed complaints through an intuitive interface that supports descriptive text, geolocation data in the form of latitude and longitude, and image uploads. The backend is developed using the Flask framework with MySQL as the database, ensuring secure authentication and organized data storage.***
For administration, the platform provides a comprehensive dashboard to manage complaints, monitor resolution progress, and support informed decision-making. A machine learning model based on the Random Forest Regressor, trained on historical complaint records and optimized with RandomizedSearchCV, is integrated to predict the estimated time required to resolve reported issues. Spatial patterns are analyzed using K-Means clustering to detect areas with high complaint density. Visual insights are generated in the form of bar chart, heatmap, and cluster map using Matplotlib and Seaborn.
Through its combination of real-time complaint collection, predictive modeling, and spatial analytics, the system improves transparency, speeds up municipal responses, and supports data-driven governance. Future developments may involve mobile application integration, real-time status tracking, and intelligent allocation of resources for issue resolution.
Keywords: Advanced web-based solution, Urban civic problems, Road damage, Drainage leakage, Water overflow, Garbage accumulation, Electricity outages, Borewell malfunctions, Citizen complaints, Intuitive interface, Descriptive text, Geolocation data, Latitude, Longitude, Image uploads, Flask framework, MySQL database, Secure authentication, Organized data storage, Administration dashboard, Complaint management, Resolution monitoring, Informed decision-making, Machine learning model, Random Forest Regressor, Historical complaint records, RandomizedSearchCV optimization, Prediction of resolution time, Spatial pattern analysis, K-Means clustering, Complaint density, Visual insights, Bar chart, Heatmap, Cluster map, Matplotlib, Seaborn, Real-time complaint collection, Predictive modeling, Spatial analytics, Transparency, Municipal responses, Data-driven governance, Mobile application integration, Real-time status tracking, Intelligent resource allocation.

I. INTRODUCTION

Urban areas frequently encounter recurring civic issues such as road deterioration, drainage leakage, water overflow, power outages, garbage accumulation, and borewell malfunctions. If not addressed promptly, these problems can disrupt daily activities, compromise public safety, degrade environmental hygiene, and reduce overall quality of life. Although some digital platforms are available to facilitate citizen engagement, most lack the specialized, structured workflows required for streamlined complaint submission, efficient tracking, and intelligent prediction of resolution timelines. As a result, response times are often delayed, and municipal resources are not utilized effectively.

This system offers a focused, data-driven, and scalable web application to address these challenges. Developed using the Flask framework for backend services and MySQL for persistent storage, it enables citizens to submit detailed complaints that include descriptions, image uploads, and precise geolocation data in terms of latitude and longitude.

The platform incorporates machine learning and visual analytics to assist administrators in making informed decisions. A Random Forest Regressor, optimized through RandomizedSearchCV, predicts the estimated number of days required to resolve a reported issue based on its type and location. K-Means clustering is applied to identify complaint hotspots, allowing municipal authorities to detect priority areas for intervention.

©IJRASET: All Rights are Reserved

To enhance monitoring and transparency, the system provides interactive visualizations such as bar chart for issue frequency, heatmap for geographic distribution, and cluster map for hotspot identification. These visuals are displayed in a secure, role-based administrative dashboard that also supports complaint management, prediction functions, image handling, and resolution tracking.
While potential future developments include mobile application integration, real-time complaint tracking, and automated resource allocation, the current version delivers a reliable foundation for participatory and data-driven urban governance. By combining citizen input with predictive modeling and spatial analytics, the platform enables faster response, greater transparency, and more efficient civic management.

A. Objective

The *Predictive Visualization of Community Municipal Challenges* project aims to create a web-based platform for efficient and transparent reporting of urban civic issues such as road damage, drainage leaks, garbage accumulation, power outages, water-overflow and borewell failures. Built using Flask and MySQL, the system allows citizens to submit complaints with descriptions, images, and geolocation data, while administrators leverage machine learning models like Random Forest Regressor and K-Means clustering, along with visual analytics, for informed decision-making. The goal is to deliver a data-driven solution that simplifies reporting, improves issue resolution, and enhances urban governance through the following key objectives:

1) Simplify citizen complaint reporting through an intuitive interface
2) Predict resolution timelines using Random Forest-based ML models
3) Identify high-density issue areas using K-Means clustering
4) Visualize complaint trends via bar-chart, heatmap, and cluster map
5) Ensure secure and scalable system architecture using Flask and MySQL

This work aims to support smarter urban governance by integrating citizen participation with predictive analytics and visual insights. It demonstrates how machine learning and web technologies can drive efficient, transparent, and data-informed municipal management

II. LITERATURE SURVEY

Existing civic complaint platforms like Swachhata-MoHUA and MyGHMC offer partial solutions but exhibit clear limitations. Swachhata-MoHUA, developed under the Ministry of Housing and Urban Affairs, mainly supports cleanliness-related categories like garbage dumps, public toilets, and sweeping. However, it lacks broader civic issue coverage such as road damage, drainage blockages, power failures, and borewell problems. It also suffers from major usability complaints such as false complaint closures, slow responses, OTP failures, and a rigid category system. Additionally, its success heavily depends on public awareness and repeated complaint filing, which limits long-term effectiveness.
On the other hand, MyGHMC is an integrated utility app offering services like property tax payments, trade license status, weather alerts, and limited civic issue reporting. Its interface, however, is not streamlined for municipal complaint tracking alone. Many users struggle to locate the right categories for issues like water overflow, road potholes, or borewell damage. Some issues are entirely missing (e.g., “Street Power Supply” or drainage leakage), and others are limited in scope (e.g., electricity issues restricted to public toilets). The mixture of non-civic services with complaint reporting makes the app bulky and confusing for users who only wish to report urban faults.
To address these gaps, our project proposes a dedicated, user-friendly web-based platform focused solely on key municipal issues—namely road damage, drainage leakage, water overflow, garbage accumulation, power supply failure, and borewell problems. By eliminating unrelated features and providing clear complaint categories, geolocation tracking, and image uploads, the platform simplifies civic participation. Machine learning models (Random Forest Regressor) are used to predict issue resolution times, while K-Means clustering highlights complaint hotspots. The goal is to create a responsive, transparent, and intelligent municipal reporting system that empowers citizens and enables authorities to take timely, data-driven action—ultimately improving urban life and public trust.

III.METHODOLOGY OF PROPOSED SYSTEM

A. Proposed System

The Predictive Visualization of Community Municipal Challenges is a web-based platform developed using Flask and MySQL that allows citizens to report issues such as road damages, drainage leakages, water overflows, garbage accumulation, power supply failures, and borewell problems. Users can submit reports with descriptions, images, and geolocation through a secure, easy-to-use interface supported by Werkzeug authentication.

©IJRASET: All Rights are Reserved

The system uses a Random Forest Regressor, optimized via RandomizedSearchCV, to predict resolution times based on issue type and location. It employs K-Means clustering to detect frequently affected areas, helping administrators identify hotspots. Visual tools like heatmap, cluster map, and bar-chart—created using Matplotlib and Seaborn—allow administrators to understand issue patterns and prioritize responses effectively.

This platform streamlines issue reporting for citizens and equips municipal administrators with powerful data-driven tools for faster, smarter urban problem-solving. The goal is to enhance transparency, accelerate response, and improve the quality of life through timely maintenance and better infrastructure management.

B. Dataset Description for Predicting Resolution Time

This project utilizes the dataset "hyderabad_issues_dataset_days_1_to_14 (2).csv" to train a Random Forest Regressor for predicting municipal issue resolution times in Hyderabad, India. Comprising 10000 rows, it includes Latitude, Longitude, and Issue Type columns, capturing essential features for accurate forecasting. This dataset is pivotal for enabling precise machine learning predictions within the Flask-based platform.

Key Dataset Features and Importance

1) Structure and Content: The dataset contains 10000 records with Latitude (float, 17.2–17.6), Longitude (float, 78.2–78.7), and Issue Type (categorical, e.g., road damage, drainage leak, garbage accumulation), specifically designed to train the Random Forest Regressor for resolution time prediction.

2) Geospatial Accuracy: Coordinates are constrained to Hyderabad's geographic bounds (centered at 17.3850, 78.4867), ensuring location-specific relevance for modeling municipal issue patterns.

3) Data Quality Assurance: The load_and_validate_data() function validates required columns and applies OneHotEncoder to Issue Type, ensuring data consistency for robust Random Forest performance.

4) Predictive Modeling Role: Exclusively used to train the Random Forest Regressor with RandomizedSearchCV, the dataset enables accurate estimation of resolution times based on issue type and location.

5) Technical Significance: By providing structured, high-quality data, the dataset supports the development of a scalable predictive model, enhancing the platform's ability to deliver precise resolution forecasts

C. System Architecture

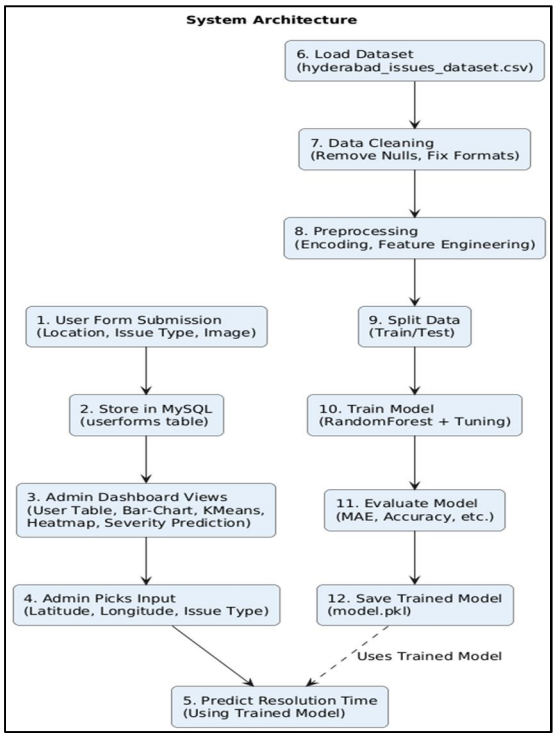


Figure-1: System Architecture of Predictive Visualization of Community Municipal Challenges

The system is architecturally designed to facilitate user issue reporting and provide an administrative interface with predictive analytics capabilities. The workflow is logically divided into two interconnected streams: the User and Administration Workflow and the Machine Learning Model Lifecycle.

1) User and Administration Workflow

a) User Form Submission: The process begins with a user reporting a civic issue via a form. The form captures critical data points, including the user's location (latitude and longitude), the issue type, a textual description, and an optional image.

b) Data Storage: The submitted data is securely stored in a userforms table within a MySQL database. This table serves as the central repository for all reported issues, ensuring data persistence and integrity.

c) Administrative Dashboard Views: Administrators can access a comprehensive dashboard to view and analyze the reported issues. The dashboard presents a user table of all submissions and visualizes the data through a bar-chart of issue frequencies, a heatmap of issue distribution by area, and a K-means clustering plot to identify geographical hotspots of issues.

d) Admin Picks Input: For predictive analysis, an administrator can manually input specific parameters: latitude, longitude, and an issue type. The system validates this input to ensure its accuracy before proceeding with the prediction.

e) Predict Resolution Time: The system uses a pre-trained machine learning model to predict the estimated time required to resolve the issue based on the administrator's input. The prediction is returned in days, providing a data-driven estimate for resolution.

2) Machine Learning Model Lifecycle to Predict the Resolution Time

a) Load Dataset: The predictive modelling pipeline is initiated by loading a historical dataset from a CSV file, hyderabad_issues_dataset_days_1_to_14 (2).csv. This dataset contains the features necessary for training the model, including location and issue type, as well as the target variable: resolution days.

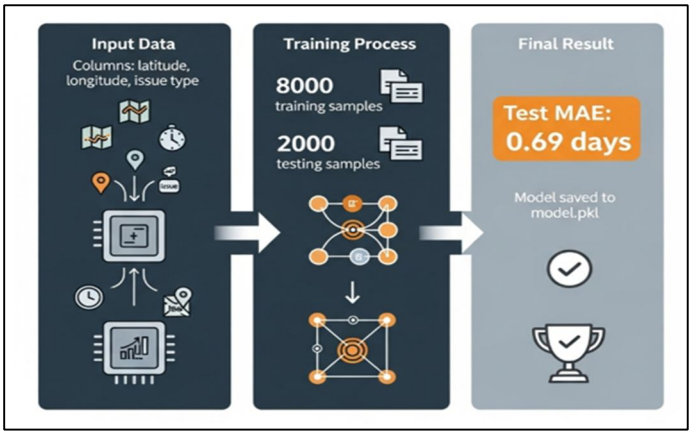
b) Data Cleaning: The loaded dataset undergoes a cleaning process to prepare it for analysis. This involves standardizing column names and validating that all required columns (latitude, longitude, issue type, resolution days) are present.

c) Preprocessing: The data is pre-processed using a ColumnTransformer to handle different feature types. Numerical features (latitude, longitude) are passed through, while the categorical feature (issue type) is encoded using OneHotEncoder.

d) Split Data: The pre-processed data is partitioned into training and testing sets, with 80% used for training and 20% for testing. This partitioning is crucial for model validation.

e) Train Model: A RandomForestRegressor is trained on the training data. To optimize the model, RandomizedSearchCV is employed to tune key hyperparameters such as n_estimators and max_depth.

f) Evaluate Model: The trained model's performance is evaluated on the test set using the Mean Absolute Error (MAE) as the primary metric. MAE provides a clear measure of the average prediction error in days.




```
2025-08-26 12:32:06,416 - INFO - Loaded columns: ['latitude', 'longitude', 'issue type', 'resolution days']
2025-08-26 12:32:06,420 - INFO - Data split: 8000 training, 2000 testing samples
2025-08-26 12:32:06,420 - INFO - Starting model training...
2025-08-26 12:32:38,776 - INFO - Best parameters: {'regressor__n_estimators': 200, 'regressor__min_samples_split': 2, 'regressor__min_samples_leaf': 2, 'regressor__max_depth': 10}
2025-08-26 12:32:38,777 - INFO - Test set MAE: 0.69 days
2025-08-26 12:32:38,839 - INFO - Model saved to model.pkl
```

Figure-2 & 3: Result of Model's Performance by MAE

3) Save Trained Model

The best-performing model identified during the training phase is serialized and saved to a file named model.pkl. This persistent model file allows for its reuse in the prediction process without the need for retraining.

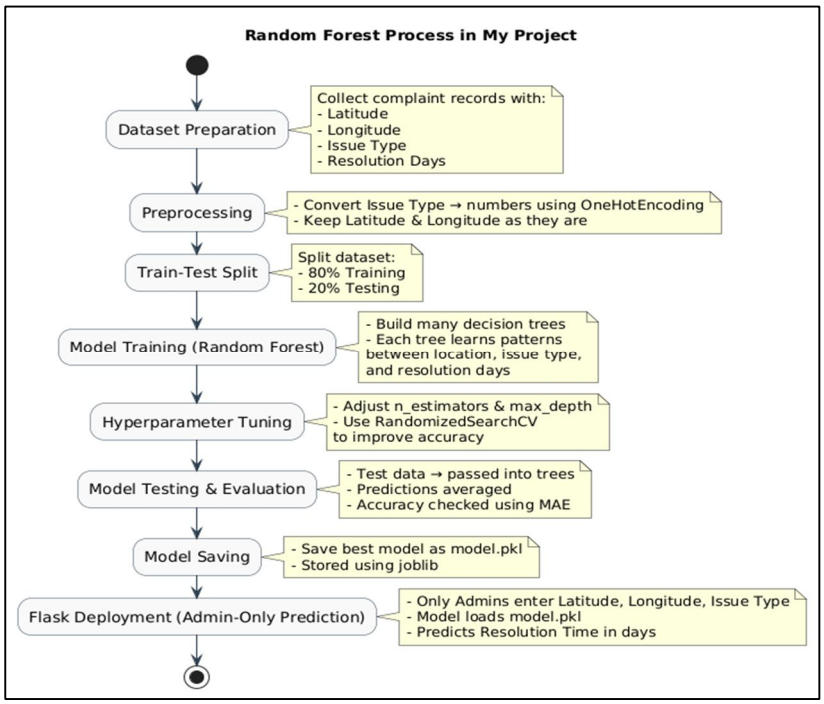


Figure-4: Random Forest Process to Predict the Resolution Time

D. Methodology

The development approach follows a structured pipeline consisting of three major stages: data preparation, machine learning model creation, and integration into the application. This multi-step strategy ensures the system can reliably estimate how long civic issues may take to be resolved.

1) Data Collection and Preparation

In the first step, raw data from the hyderabad_issues_dataset_days_1_to_14 (2).csv file is imported using the pandas library. A validation check ensures essential fields—latitude, longitude, issue type, and resolution days—are present; otherwise, the system halts with an error. The data undergoes cleaning by trimming spaces and converting column names to lowercase for consistency. Feature selection is then performed by choosing latitude, longitude, and issue type as inputs for the model. The categorical variable issue type is transformed using OneHotEncoder within a ColumnTransformer, while numeric features are used directly without changes.

2) Model Development and Optimization

To predict resolution times, the system employs a RandomForestRegressor, which is known for its effectiveness in regression tasks involving complex data relationships. Data is split into training (80%) and testing (20%) subsets. The entire pipeline—including preprocessing—is encapsulated using Pipeline. Hyperparameters such as the number of trees and depth of the forest are optimized using RandomizedSearchCV, improving model accuracy and reducing overfitting risks.

Once trained, the model's performance is measured using Mean Absolute Error (MAE), offering a simple interpretation of average prediction error. The final optimized model is then saved using joblib as model.pkl, enabling future predictions without retraining.

3) Application Integration

In the final stage, the trained model and data visualizations are connected to a web-based platform.

Users can submit civic issue reports through the interface, while administrators access a dashboard that includes visual analytics like bar-chart, heatmap, and K-Means cluster map generated from real-time data stored in a MySQL database.

When admins input new issue information (latitude, longitude, issue type) in prediction page, the system uses the preloaded model.pkl to instantly predict the estimated resolution duration. This prediction is displayed on the dashboard, supporting quicker decision-making and prioritization.

E. Database design (MySQL Schema)

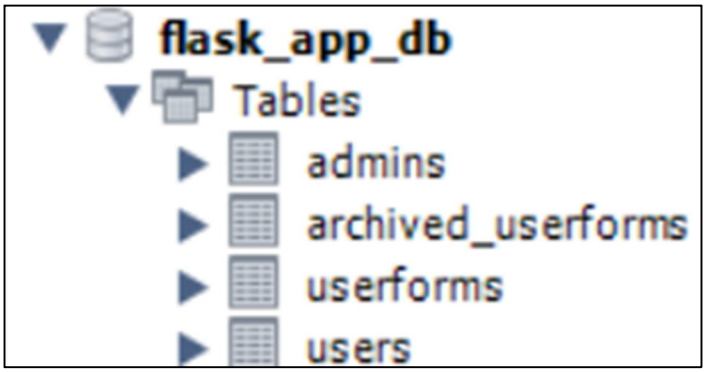


Figure-5: MySQL Database Tables List

There are 4 tables in this Project's MySQL Database: -

1) users table- Stores data of every user who gets Logged in.
2) userforms table – Stores the data of USER FORM Details, when User Submits the Form, here the Submitted data stores
3) archived_userforms table - only deleted or Issues Resolved users will go to this table for the Record Purpose
4) admins table - Store only admin details - we can add anyone if we want.

IV. RESULTS

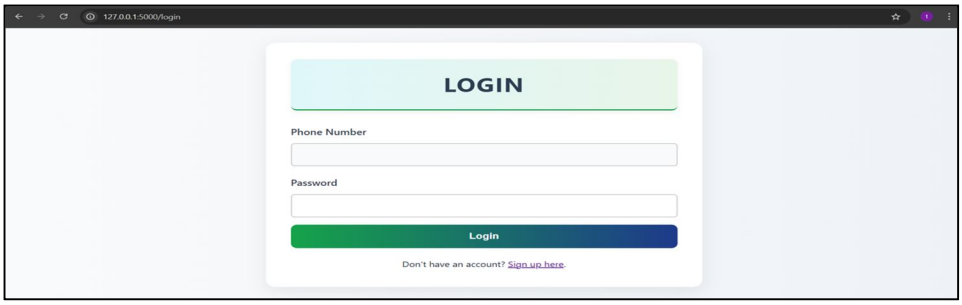


Figure-6: Login page for Users and Admins

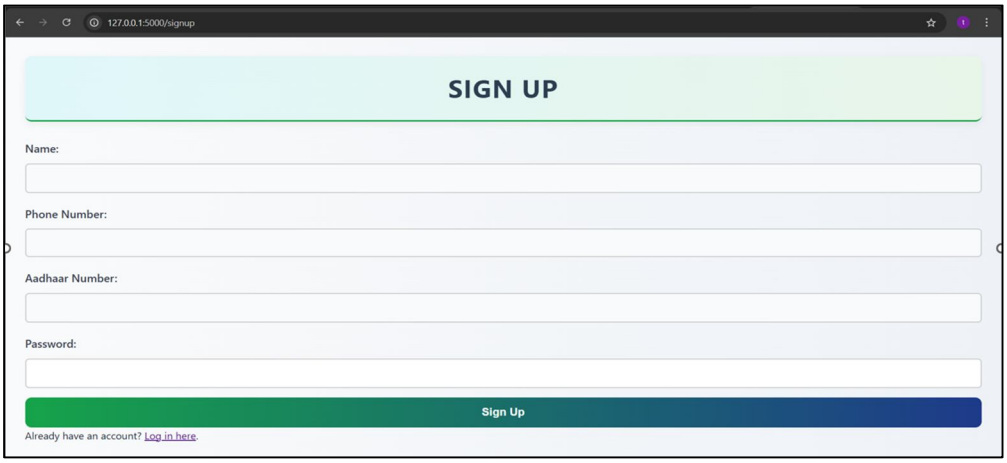


Figure-7: Sign-Up Page for New Users.

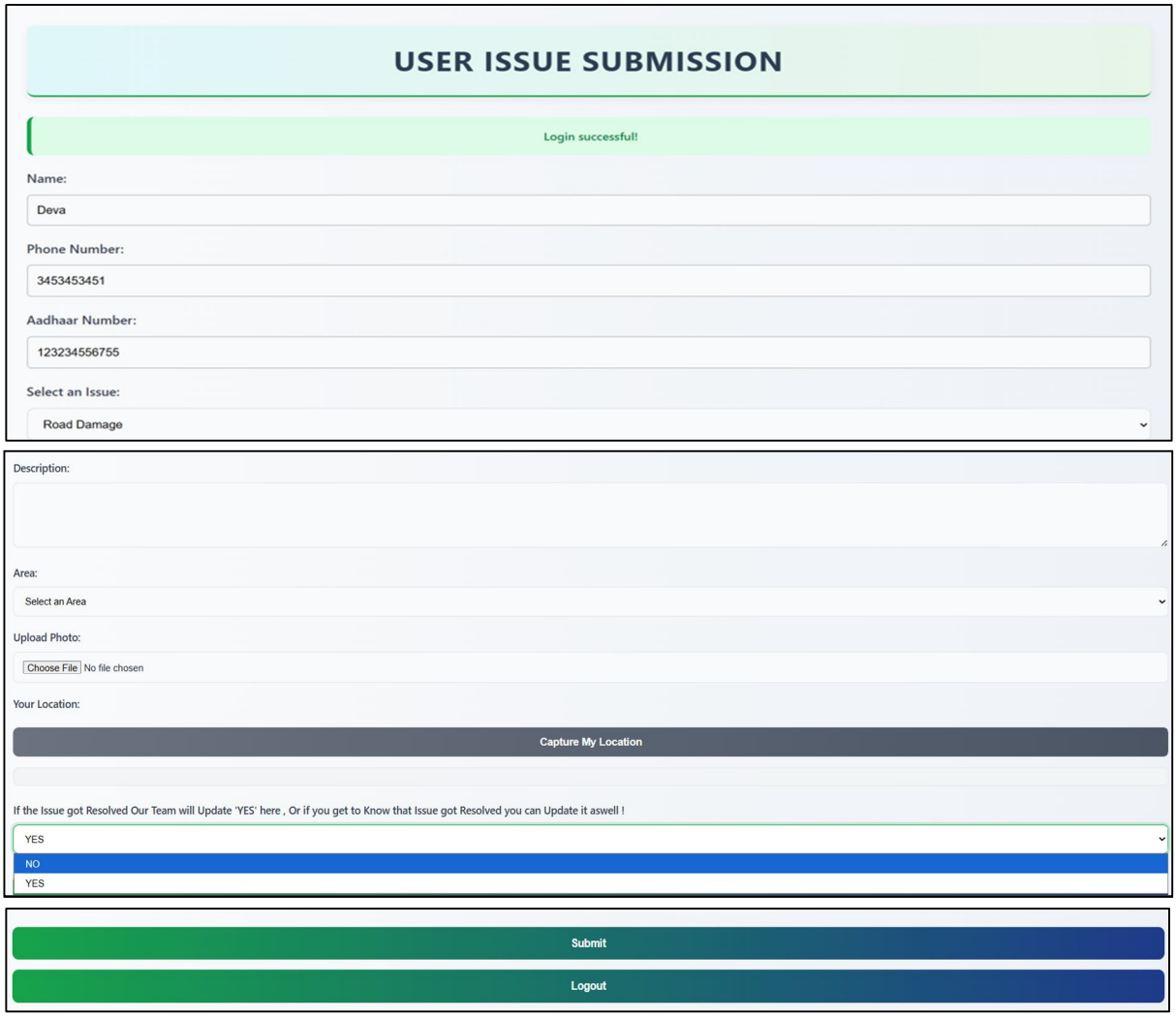


Figure-8: Form Submission Page for Users

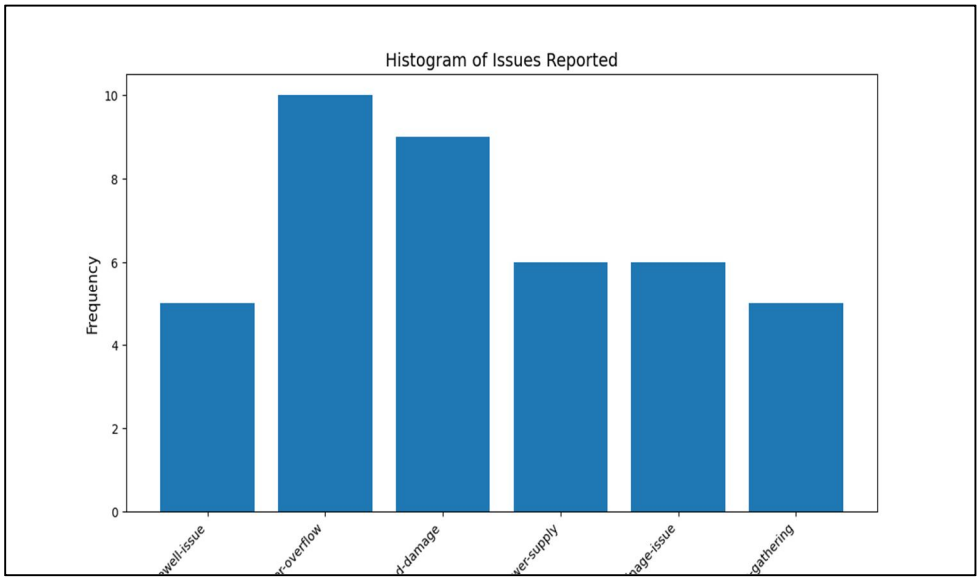


Figure-9: Admin Dashboard ‘s Bar-chart of Issues Reported

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue IX Sep 2025- Available at www.ijraset.com

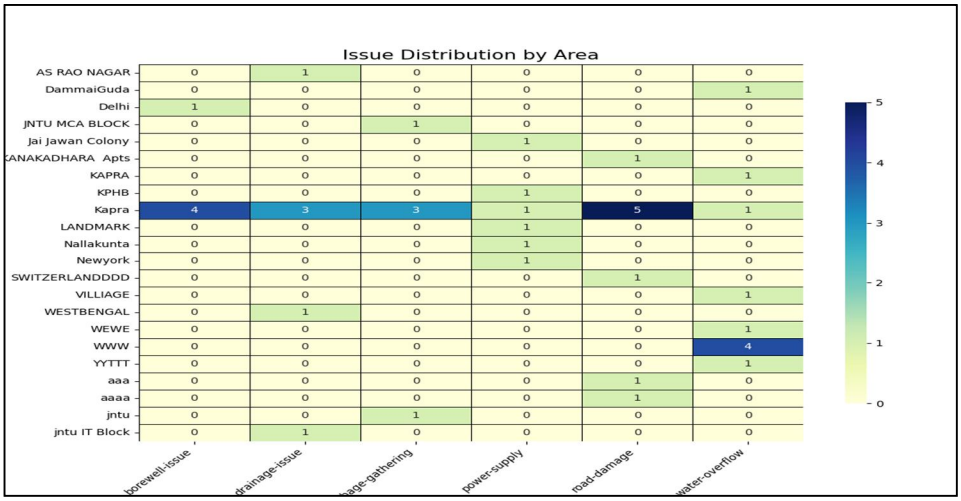


Figure-10: Admin Dashboard's Issue Distribution by Area

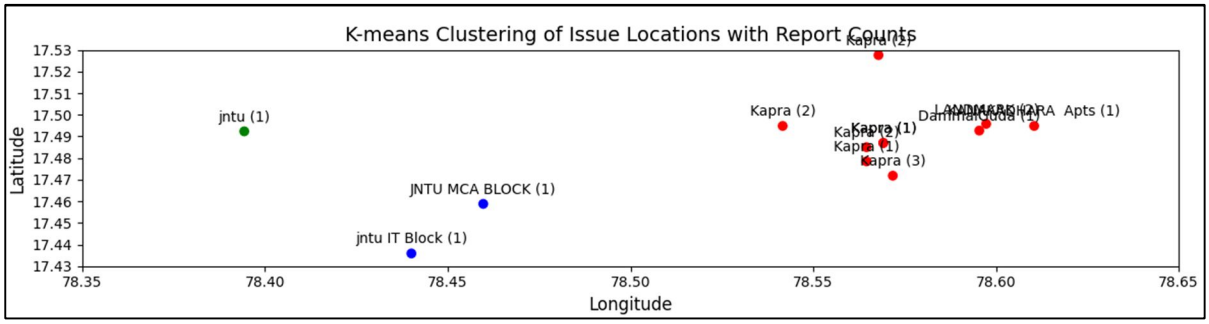
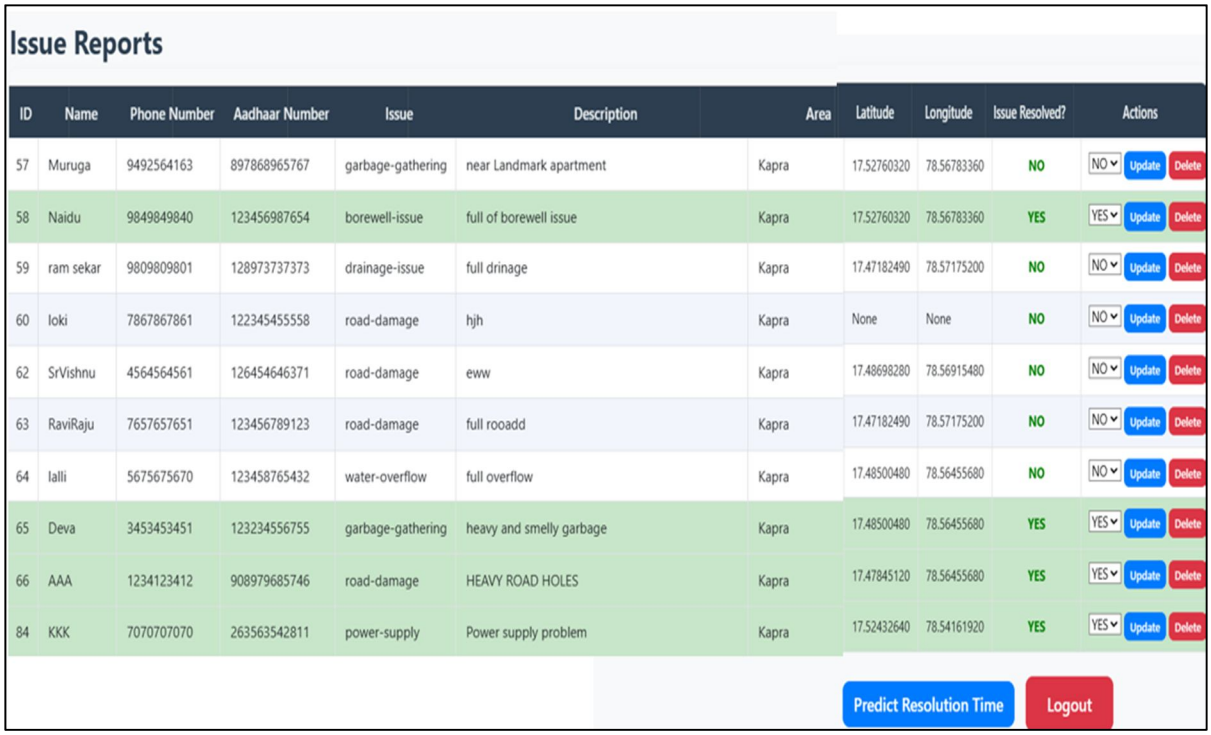


Figure-11: Admin Dashboard's K-Means Clustering of Issue Hotspots



ID	Name	Phone Number	Aadhaar Number	Issue	Description	Area	Latitude	Longitude	Issue Resolved?	Actions
57	Muruga	9492564163	897868965767	garbage-gathering	near Landmark apartment	Kapra	17.52760320	78.56783360	NO	NO Update Delete
58	Naidu	9849849840	123456987654	borewell-issue	full of borewell issue	Kapra	17.52760320	78.56783360	YES	YES Update Delete
59	ram sekar	9809809801	128973737373	drainage-issue	full drinage	Kapra	17.47182490	78.57175200	NO	NO Update Delete
60	loki	7867867861	122345455558	road-damage	hjh	Kapra	None	None	NO	NO Update Delete
62	SrVishnu	4564564561	126454646371	road-damage	eww	Kapra	17.48698280	78.56915480	NO	NO Update Delete
63	RaviRaju	7657657651	123456789123	road-damage	full rooadd	Kapra	17.47182490	78.57175200	NO	NO Update Delete
64	lalli	5675675670	123458765432	water-overflow	full overflow	Kapra	17.48500480	78.56455680	NO	NO Update Delete
65	Deva	3453453451	123234556755	garbage-gathering	heavy and smelly garbage	Kapra	17.48500480	78.56455680	YES	YES Update Delete
66	AAA	1234123412	908979685746	road-damage	HEAVY ROAD HOLES	Kapra	17.47845120	78.56455680	YES	YES Update Delete
84	KKK	7070707070	263563542811	power-supply	Power supply problem	Kapra	17.52432640	78.54161920	YES	YES Update Delete

Figure-12: User's Table in the Admin Dashboard

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

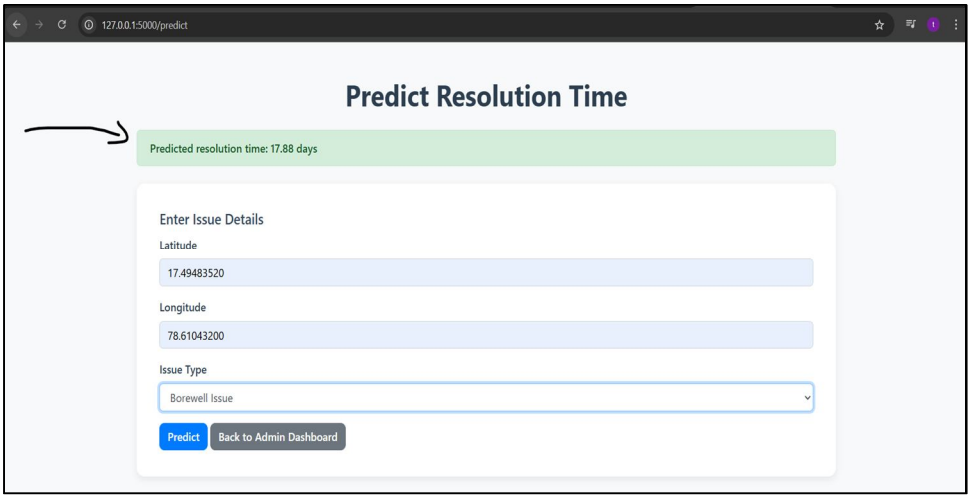


Figure-13: Prediction tab opens when the Admin Clicks on "Predict Resolution Time" Button in the Admin Dashboard

A. Description About the Results

Both Admins and Users can log in through the Login Page. Admins log in using their provided credentials, while Users can log in if they already have an account or have signed up earlier. The Sign-Up Page is meant for new users to create an account, after which they are redirected to the Login Page. Once logged in, Users are redirected to the User Form Page.

On the User Form Page, Users can enter the issue details such as location, issue type, and whether the issue is already resolved (YES/NO). By clicking the "Capture My Location" button, the system uses the Geolocation API from the backend to automatically fetch the user's current location. After filling in the details, users can submit the form.

Submitted data is visualized on the Admin Dashboard through various charts and tables, including a Bar-Chart, Area-Wise Severity Plot, K-Means Clustering Plot, and a User Table. In the User Table, a row marked in green indicates that the issue has been resolved—either by the user or updated by the admin. Admins can also update the resolution status and delete users whose issues are resolved. Deleted users' data is securely moved to an archived_table in the backend MySQL database. Finally, Admins can click the "Predict Resolution Time" button, which redirects them to a new page where they can manually input a user's Latitude, Longitude, and Issue Type to predict the expected time for resolution.

V. LIMITATIONS AND FUTURE SCOPE

The project, while functionally robust for user reporting, data visualization, and machine learning-based resolution prediction, is constrained by several significant limitations. Firstly, the system relies on a local MySQL database, restricting scalability and making it less suitable for high-volume or widely distributed deployments. Secondly, the machine learning model—used for predicting resolution days—is trained on a limited, location-specific dataset from Hyderabad, which greatly reduces its generalizability to other regions or timeframes. Thirdly, while user passwords are securely hashed, administration passwords are stored and checked in plain text, presenting a major security risk. Additional limitations include basic error handling, a relatively simple user interface, static handling of issue types, minimal advanced geospatial analytics, and hard-coded configurations that further hamper maintainability and extensibility.

To address these shortcomings, there is substantial scope for future enhancements. The database design can be migrated to cloud-based or distributed systems to improve scalability and reliability. Expanding the dataset to cover broader geographies and time periods, as well as integrating data from multiple sources, will make the machine learning model more robust and generalizable. Implementing hashed storage and verification for admin passwords, along with stronger session and access control measures, will secure the platform against common vulnerabilities.

Upgrades to the user interface—such as the introduction of interactive maps, real-time updates, multilingual support, and richer analytics—will foster greater engagement. Moving configuration parameters out of source code and supporting environment-based management will streamline deployment. Lastly, integrating APIs for real-time government or third-party responses can turn the platform into a scalable and secure smart civic solution

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

VI. CONCLUSION

The project titled "Predictive Visualization of Community Municipal Challenges" successfully delivers an intelligent, user-centric platform that simplifies municipal issue reporting while empowering city administrators with predictive insights and data-driven visual analytics.

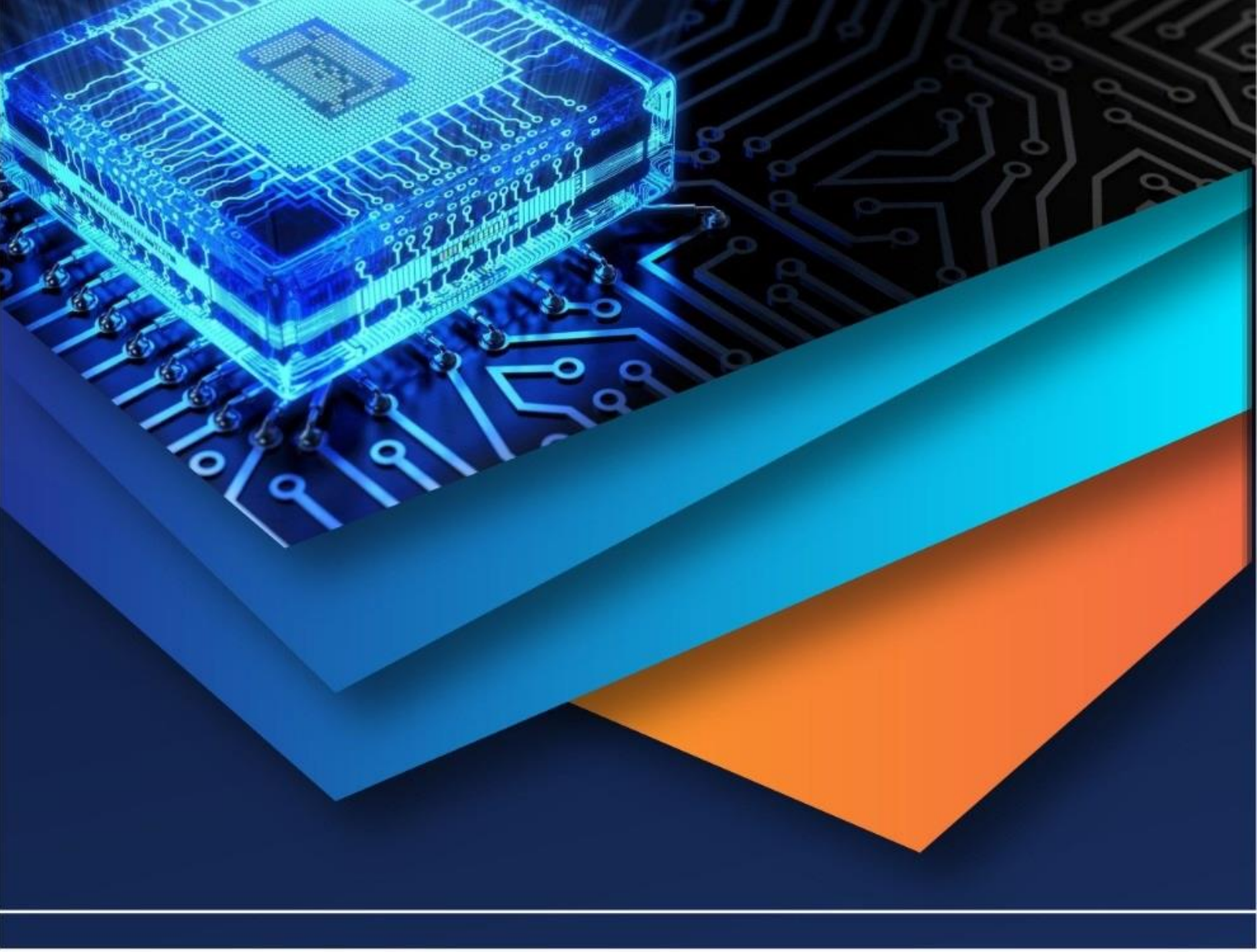
Through the seamless integration of Flask (Python), MySQL, and machine learning (Random Forest Regressor), the system enables users to report issues like road damage, drainage leaks, power outages, and more—along with accurate geolocation and image evidence. On the administrative side, the use of K-Means Clustering, heatmap, and bar-chart allows for smart visualization of complaint patterns and identification of urban hotspots.

Most notably, the system introduces a predictive module capable of estimating the resolution time for reported issues based on historical data. This innovative helps administrators prioritize complaints more effectively.

The platform is simple to use, scalable for broader deployment, and designed with modularity, security, and extensibility in mind. It addresses the limitations of generic civic apps like MyGHMC by offering a focused, specialized, and intelligent solution for community-driven problem reporting and resolution.

REFERENCES

[1] Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. Retrieved from: https://scikit-learn.org/
[2] McKinney, W. Pandas: Python Data Analysis Library. Retrieved from: https://pandas.pydata.org/
[3] Harris, C. R., Millman, K. J., van der Walt, S. J., et al. (2020). Array programming with NumPy. Nature, 585, 357–362. Retrieved from: https://numpy.org/
[4] Hunter, J. D. Matplotlib: Visualization with Python. Retrieved from: https://matplotlib.org/
[5] Waskom, M. Seaborn: Statistical Data Visualization. Retrieved from: https://seaborn.pydata.org/
[6] Ronacher, A. Flask: A Lightweight Web Application Framework. Retrieved from: https://flask.palletsprojects.com/
[7] Oracle Corporation. MySQL Documentation. Retrieved from: https://dev.mysql.com/doc/
[8] Joblib Development Team. Joblib: Python Object Serialization and Caching. Retrieved from: https://joblib.readthedocs.io/
[9] MacQueen, J. (1967). Some Methods for Classification and Analysis of Multivariate Observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability.
[10] Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32.
[11] Greater Hyderabad Municipal Corporation (GHMC), Telangana Government. MyGHMC App Overview. Retrieved from: https://www.ghmc.gov.in/
[12] W3C / Mozilla Developer Network (MDN). HTML5 Geolocation API Documentation. Retrieved from: https://developer.mozilla.org/en-US/docs/Web/API/Geolocation_API
[13] Stack Overflow Developers' Community. Technical Discussions and Code Snippets. Retrieved from: https://stackoverflow.com/
[14] GitHub Developer Community. Open-source Contributions and Repositories. Retrieved from: https://github.com/
[15] Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.
[16] Van Rossum, G., & Drake, F. L. (2009). The Python Language Reference Manual. Python Software Foundation. Retrieved from: https://docs.python.org/
[17] Pedregosa, F., & Varoquaux, G. (2024). Scikit-learn User Guide. Retrieved from: https://scikit-learn.org/stable/user_guide.html
[18] Google Developers. (2024). Google Maps JavaScript API. Retrieved from: https://developers.google.com/maps/documentation/javascript
[19] Lantz, B. (2013). Machine Learning with Python for Everyone. Addison-Wesley.
[20] McKinney, W. (2022). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and Jupyter. O'Reilly Media.
[21] Python Software Foundation. (2024). Werkzeug Documentation. Retrieved from: https://werkzeug.palletsprojects.com/



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)