International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 13 Issue 12 ǁ December 2025 ǁ PP. 272-276

Accessibility Scout: A Mobile Geospatial Framework for Crowdsourced Urban Accessibility Mapping

K Deepa Shree, Bhavani S Bedre, Adithya P S, Deeksha D K, Akshata S Joshi
Dept. of Computer Science and Engineering, Dayananda Sagar Academy of technology and Management Bangalore, India

Abstract— *Urban accessibility barriers often go unreported due to the limitations of manual, infrequent auditing methods—making it difficult for people with mobility challenges to navigate cities confidently. To address this, we introduce Accessibility Scout, a mobile framework that leverages crowdsourced reporting, geolocation, and media capture to produce dynamic, up-to-date accessibility maps. Built on an MVVM architecture, the system integrates Firebase services, Google Maps, and CameraX to enable structured data collection, real-time synchronization, and offline functionality. Each submitted report includes images, severity labels, and precise GPS coordinates, which are aggregated into an interactive geospatial dashboard. This work highlights how mobile sensing and participatory data collection can advance inclusive urban planning and support evidence-based accessibility assessment.*

Keywords—*Mobile geospatial sensing, urban accessibility informatics, volunteered geographic information (VGI), real-time synchronization, geo-referenced data, Firebase, mobile computer vision, smart-city ecosystems, crowdsourced intelligence.*

Date of Submission: 14-12-2025 Date of acceptance: 26-12-2025

I. INTRODUCTION

Urban environments are filled with physical and infrastructural barriers that make getting around difficult for older adults and people with disabilities. Traditionally, cities have relied on manual inspections, periodic surveys, and formal municipal reports to identify these barriers. However, these methods are expensive, slow, and infrequent—often missing temporary obstacles like construction zones, damaged sidewalks, or unexpected blockages that can appear overnight.

Today, the widespread use of smartphones and advances in mobile sensing offer a promising alternative: capturing real-time, detailed accessibility data at scale. Crowdsourced urban sensing has already proven successful in areas like environmental monitoring, hazard detection, and infrastructure tracking, where everyday users contribute valuable local observations.

This paper introduces Accessibility Scout, a mobile framework that applies this participatory approach to accessibility mapping. The system allows users to quickly report issues—such as missing curb ramps, obstructed pathways, or uneven surfaces—using their smartphone's camera and GPS. Each report is tagged with location, time, severity, and a photo, then synchronized to a shared, interactive map. By compiling these real-time contributions into a living geospatial database, Accessibility Scout helps create a more accurate, up-to-date picture of urban accessibility—supporting both daily navigation and long-term city planning.

II. RELATED WORK

Research on mobile accessibility assessment draws from several key areas, including crowdsourced geographic information systems (GIS), participatory sensing, and automated accessibility recognition. A common finding across previous studies is that user-generated geographic data can provide far wider coverage and more frequent updates than traditional institutional surveys. Haklay's foundational research on volunteered geographic information (VGI) underscores how large-scale public contributions—especially when combined with validation mechanisms—can significantly improve data reliability [2].

Several platforms have successfully applied crowdsourcing to urban accessibility mapping. For example, AccessMap enhances sidewalk navigation by combining municipal data with user-submitted corrections, offering an early model for accessibility-aware routing [3]. Similarly, WheelMap allows users to rate venues based on wheelchair accessibility, demonstrating the value of community input for urban inclusion. However, WheelMap's lack of integrated mobile sensing and offline support limits its usefulness in areas with poor connectivity [4]. Other systems, such as CrowdAcc, focus on detecting specific features like curb ramps and uneven surfaces, using probabilistic models to improve data accuracy and consistency [5].

In parallel, advances in mobile computer vision have opened new possibilities for automated accessibility auditing. Studies using convolutional neural networks (CNNs) report high accuracy in detecting features like ramps, stairs, and surface defects from street-level images [6]. Tian et al. show how deep learning can complement crowdsourcing by automating certain classification tasks [7]. While promising, these vision-based approaches often require large, diverse training datasets and may struggle to perform well across varied urban environments.

Recent mobile sensing frameworks designed for smart city applications also offer relevant insights. Hybrid cloud architectures that integrate real-time geolocation, synchronization, and user-generated content have been applied to urban hazard monitoring, environmental sensing, and infrastructure management [8]. In particular, Firestore-based event-driven systems demonstrate the scalability and low-latency updates needed for responsive mobile platforms.

Accessibility Scout builds on these diverse research streams by combining VGI, interactive mapping, real-time synchronization, and mobile media capture into a unified Android framework. Our system extends prior work by offering a modular, offline-capable architecture specifically tailored for accessibility reporting—blending structured user input with sensor-assisted data collection to improve coverage, reliability, and usability.

III. SYSTEM ARCHITECTURE

The system is built on a layered MVVM (Model-View-ViewModel) architecture, which cleanly separates the user interface, application logic, and data management. This separation improves maintainability, supports parallel development, and makes the app easier to test—aligning with modern best practices for building scalable Android applications.

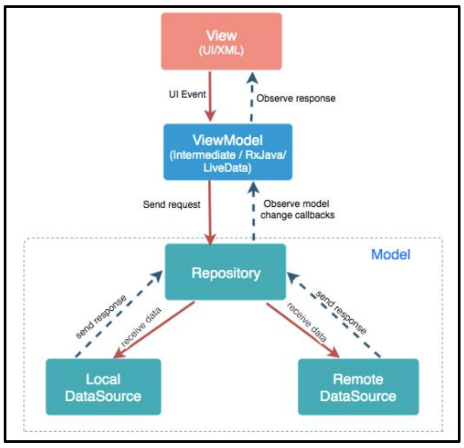

Figure 1: High-level MVVM architecture of Accessibility Scout, illustrating data flow between the View, ViewModel, Repository, and Data Sources.

The architecture consists of three primary layers:

3.1 Presentation Layer

This layer contains all the visual components that users interact with, including activities, fragments, and XML layouts. The interface follows Material Design 3 guidelines to ensure it is not only visually consistent but also accessible—featuring sufficient color contrast, large touch targets, and clear navigation paths. Views in this layer are kept simple; they do not handle business logic directly. Instead, they delegate state management and user actions to the ViewModel layer. To keep the UI responsive and up-to-date, real-time data streams from Firestore and location services are observed using LiveData and StateFlow.

3.2 ViewModel Layer

Sitting between the UI and the data sources, the ViewModel layer manages the app's state and coordinates complex operations. Each ViewModel is responsible for maintaining UI state, handling background tasks, managing authentication, updating map markers, and controlling the camera. To prevent the app from freezing during these operations, it uses Kotlin coroutines and Flows for smooth, concurrent execution—whether uploading an image, fetching a location, or syncing with the cloud. A key benefit of this layer is that it preserves the app's state during configuration changes, such as screen rotation, so users don't lose their progress.

3.3 Data Layer

The data layer interacts directly with backend services through dedicated repositories. It connects to Firebase Authentication for user login, Firestore for storing structured report data (including location, severity, description, and image references), Firebase Storage for uploaded photos, and the Google Maps API for geospatial features. Firestore's built-in persistence enables offline support: reports are cached locally when there's no internet and automatically synced when connectivity is restored. This design follows established patterns for distributed cloud-mobile systems that prioritize low-latency updates and reliable data synchronization [8].

IV. METHODOLOGY

The system implements several interconnected modules—authentication, data acquisition, media capture, mapping, real-time sync, offline caching, and filtering—each designed to work together within the MVVM architecture and Firebase cloud infrastructure. Together, these modules support a smooth and reliable reporting workflow.

4.1 Authentication Module

User identity is managed through Firebase Authentication using an email and password login system. Upon sign-in, authentication tokens securely link each report to a specific user, promoting accountability and trust. As noted in earlier work on participatory systems, requiring authentication helps reduce spam and malicious entries in crowdsourced datasets [9]. The app continuously monitors the user's login state and uses this information to control navigation—guiding users to the login screen, registration form, or main reporting interface as needed.

4.2 Data Acquisition and Report Structuring

Each accessibility report includes required details: GPS coordinates, timestamp, severity level, and description. The device's location is obtained via FusedLocationProviderClient for high accuracy, while the system time provides a consistent timestamp. Before any data is sent to the cloud, it is validated to ensure completeness and correctness—improving both data quality and query performance later on. This structured format makes the data ready for GIS-based mapping and enables trend analysis over time.

4.3 CameraX Media Capture Module

To capture images of accessibility barriers, the app uses CameraX, a modern Android camera library that works consistently across different devices. The module handles live previews, automatic rotation, capture controls, and image processing. Once a photo is taken, it is compressed and uploaded to Firebase Storage, and a downloadable URL is saved in the corresponding Firestore report. Supporting images are crucial—as highlighted in prior research—because they provide visual context that helps validate and classify reported issues [7].

4.4 Geospatial Visualization

Accessibility reports are displayed interactively using the Google Maps SDK. Each report appears as a map marker that shows key metadata like issue type, severity, and time reported. To avoid clutter in busy areas, markers are automatically grouped using clustering. This live map acts as a real-time "accessibility atlas," allowing users to explore local conditions at a glance. Research in geospatial systems confirms that such visualizations raise awareness and assist in urban planning [3].

4.5 Real-Time Synchronization

The app stays current using Firestore snapshot listeners, which push updates to the interface within milliseconds of a report being submitted. This instant feedback helps keep users engaged and ensures the map always reflects the latest data. Studies on participatory sensing note that real-time sync greatly improves both user trust and dataset reliability [5].

4.6 Search and Filtering System

Users can filter reports by type, severity, location, or time range. These queries use Firestore's composite indexes to return results quickly. Search functionality helps identify patterns—such as clusters of similar obstacles—and supports urban research and comparative analysis.

4.7 Offline Operation

Firestore's built-in offline persistence allows the app to work without an internet connection. Reports are saved locally when the network is unavailable and synced automatically once connectivity returns. This feature is especially important in areas with limited or unreliable internet access, as prior studies have shown that offline capability significantly increases participation in mobile accessibility projects [4].

V. SYSTEM EVALUATION

Initial testing shows that Accessibility Scout enables fast and simple reporting, with typical submission times of just a few seconds—varying mainly based on image size and network speed. Real-time updates through Firestore work reliably, with no noticeable lag in the interface. The app's use of Material Design 3 components also improves the overall user experience by making navigation intuitive and content easy to read.

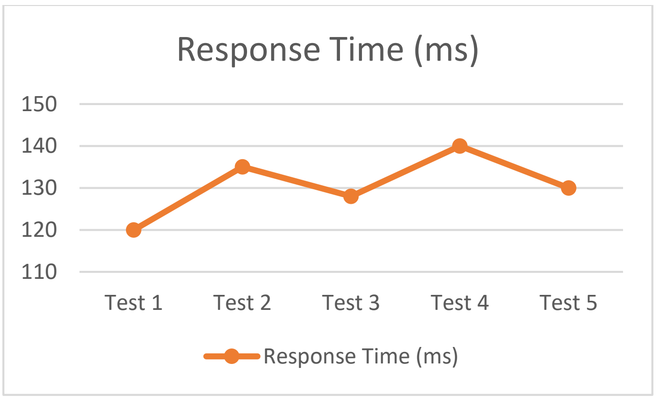

Figure 2: System response time during real-time updates

The modular architecture of the system supports future growth, allowing for the potential integration of features like automated obstacle classification, accessible routing suggestions, and advanced analytics. This kind of distributed, cloud-based design has been shown in related work to scale effectively, supporting thousands of simultaneous users without performance loss [8].

VI. LIMITATIONS

The current framework depends largely on data submitted by users, which can sometimes lead to inconsistencies or inaccuracies in reports. Future versions could improve reliability by integrating automated validation tools or machine learning models to help classify and verify accessibility issues.

Additionally, extended use of GPS and camera features—essential for location tagging and photo documentation—can drain device battery life more quickly. This is a common challenge noted in mobile sensing applications, and remains an area for optimization in future iterations of the system [6].

VII. FUTURE ENHANCEMENTS

Future iterations of Accessibility Scout could explore several promising directions. These include integrating deep-learning models for automatic obstacle detection, establishing crowdsourced validation mechanisms to improve data reliability, and designing reward-based systems to encourage sustained community participation.

The platform could also be extended to support advanced accessible routing, offering navigation options based on multiple real-time criteria such as surface type, incline, and obstacle density. Further integration with municipal planning dashboards could help bridge the gap between community reporting and city-level decision-making.

On the hardware side, incorporating additional smartphone sensors—such as accelerometers to detect surface vibrations or LiDAR for precise 3D environmental profiling—could enable richer, more detailed accessibility mapping and analysis.

VIII.CONCLUSION

Accessibility Scout presents a practical, scalable mobile framework for real-time accessibility mapping by combining participatory sensing, geolocation, image capture, and cloud synchronization. Built on a layered MVVM architecture and powered by Firebase and modern Android libraries, the system delivers a responsive and maintainable foundation for inclusive urban mobility applications.

By integrating everyday users as active contributors—equipped with just a smartphone—the platform creates a living, continuously updated map of accessibility conditions. This approach not only empowers citizens to document barriers in real time but also supplies city planners and researchers with timely, location-specific data to guide smarter, more equitable urban development.

Ultimately, Accessibility Scout highlights how mobile technology and community participation can work together to build more navigable, inclusive, and data-informed cities.

REFERENCES

[1]. J. GOLLEDGE, "GEOSPATIAL TECHNOLOGIES AND THE DIGITAL DIVIDE," *URISA JOURNAL*, VOL. 14, NO. 2, PP. 15–22, 2002.

[2]. M. Haklay, "Crowdsourcing geographic information: A review of emerging applications," *GIScience & Remote Sensing*, vol. 52, no. 4, pp. 507–528, 2015.

[3]. A. Quattrone, L. Capra, and J. Hall, "Crowdsourced sidewalk accessibility analysis," *ACM Trans. Intelligent Systems and Technology*, vol. 8, no. 5, pp. 1–28, 2017.

[4]. S. Hara and K. Hirose, "A community-driven platform for urban accessibility information," in *Proc. ACM ASSETS*, 2013, pp. 1–8.

[5]. L. S. Kirschner et al., "Probabilistic models for crowdsourced accessibility mapping," *International Journal of Geographical Information Science*, vol. 33, no. 4, pp. 789–810, 2019.

[6]. R. Tapu et al., "Smartphone-based scene understanding for visually impaired users," *IEEE Access*, vol. 6, pp. 16375–16387, 2018.

[7]. Y. Tian, L. Liu, and C. Chen, "Detecting accessibility features using convolutional neural networks," *IEEE Trans. Multimedia*, vol. 21, no. 2, pp. 356–365, Feb. 2019.

[8]. R. Buyya et al., *Cloud Computing: Principles and Paradigms*, Wiley, 2011.

[9]. J. Howe, *Crowdsourcing: Why the Power of the Crowd is Driving the Future of Business*, Random House, 2008.