International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

An Android-Based Smart Event Venue Search and Booking Application with Location-Aware Recommendations

Keerthana K[1], Prof. Indra C[2], Amulya CS[3], Poornima S[4], Dhanush Gowda HS[5]
[1,3,4,5]PG Scholar, Department of MCA, PES College of Engineering, Mandya, Karnataka, India.
[2]Assistant Professor, Department of MCA, PES College of Engineering, Mandya, Karnataka, India.
Email ID: *keertthui@gmail.com[1], indra@pesce.ac.in[2], amulyasuresh18ammu@gmail.com[3], poornimasgowda99@gmail.com[4], dhanushgowda9956@gmail.com[5]*

Abstract
Planning an event traditionally begins with the time-consuming task of identifying, comparing, and reserving a suitable venue, a process that often relies on manual site visits, phone calls, and fragmented listings spread across brokers and social media groups. This paper presents an Android-based event venue search and booking application designed to unify venue discovery, negotiation, and reservation for end users, venue owners, and professional event planners within a single mobile platform. The proposed system employs a local relational SQLite database comprising thirteen interconnected tables to manage multi-role accounts, venue listings, bookings, feedback, and add-on services, while a location-based recommendation module leverages the Android Fused Location Provider and geodesic distance computation to rank nearby venues by proximity, capacity, price, and event-type suitability. The application further integrates an automated SMTP-based email notification module and an in-app chat and inbox system to keep users, owners, and planners informed throughout the booking lifecycle, while an administrative dashboard governs account approval, venue verification, and booking oversight. Experimental evaluation on a seeded dataset of 50 venues across 8 cities and simulated multi-role activity demonstrates accurate location-based filtering, low-latency booking transactions (95 ms average), reliable database operations (12–45 ms), and consistent email delivery (97.8% success rate), confirming the suitability of the proposed framework for real-world deployment in event planning, venue management, and smart city service applications.

Keywords: *Android Application, Event Management, Venue Booking, SQLite, Location-Based Services, Mobile Computing, Event Planning, SMTP Notification.*

1. Introduction
Selecting and reserving an appropriate venue is one of the most critical and time-consuming stages of organizing any event, ranging from weddings and corporate conferences to college festivals and community gatherings. Event organizers typically rely on physical visits, word-of-mouth referrals, and unstructured phone or messaging-based inquiries to compare halls, lawns, and banquet spaces — a process that is fragmented, opaque, and difficult to scale across a city or region. Traditional approaches to venue discovery and booking suffer from several limitations, including delayed responses from venue owners, lack of real-time availability information, absence of standardized pricing and capacity disclosure, and limited ability to compare multiple venues based on location, budget, and event requirements. Manual coordination between organizers, venue owners, and third-party event planners further increases the likelihood of miscommunication, double bookings, and inconsistent service quality. Recent advancements in smartphone computing, GPS-based location services, and mobile application frameworks have transformed how consumers discover and reserve local services. Android, as the most widely adopted mobile operating system, provides robust APIs such as the Fused Location Provider and Google Play Services that allow applications to determine a user's position accurately and efficiently. By combining geolocation intelligence with structured data persistence, mobile applications can recommend nearby venues ranked by distance, price, and suitability, significantly

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

reducing the effort required to identify an appropriate event space. This research proposes an Android-based Event Venue Search and Booking application that integrates a relational SQLite data layer, Google Play Services location intelligence, an SMTP-based email notification system, and a structured multi-role workflow encompassing users, venue owners, event planners, and administrators. A custom geolocation and filtering module recommends nearby venues from live application data, while a planner proposal and discount negotiation workflow allows professional event planners to submit customized service bundles for a booking. Major Contributions:

- Development of a multi-role Android application enabling event organizers to search, compare, and book venues in real time.
- Design and implementation of a location-aware venue recommendation module using the Android Fused Location Provider and geodesic distance computation.
- Integration of a relational SQLite schema with thirteen tables to manage users, venues, bookings, feedback, services, referrals, and messaging.
- Implementation of an event-planner proposal and negotiation workflow supporting discount requests and custom service bundling.
- Development of an automated SMTP-based email notification system and an in-app chat and inbox module for real-time stakeholder communication.
- Design of an administrative governance module for user approval, venue verification, and booking oversight [1-5].

2. Related Work

Google's guide to modern Android app architecture (Android Developers, 2021) formalized best practices for activity lifecycle management, unidirectional data flow, and separation of concerns in Android applications. These guidelines directly shaped the EVB system's use of View Binding, centralized SQLiteHelper, and the SessionManager component for maintaining role-aware application state. Smith and Brown (2022) presented a cloud-based event scheduling and venue reservation system that integrated capacity management, real-time availability checking, and multi-party coordination through a centralized server. Their work demonstrated that structured data modeling for booking lifecycle states — particularly distinguishing pending, confirmed, and cancelled states — is essential for preventing double bookings, a finding reflected in the EVB system's four-state status machine. Patel et al. (2022) presented a smart event management framework for educational institutions that automated venue allocation, attendee registration, and notification dispatch within a unified platform. Their integration of automated communication triggers at each workflow milestone informed the EVB system's SMTP notification model, where every booking status transition automatically dispatches email alerts to the relevant stakeholder. Firebase's Realtime Database documentation and sample applications (Firebase Team, 2022) established the reference model for real-time data synchronization in Android applications, demonstrating listener-based update propagation and conflict resolution strategies. These patterns are planned for adoption in the EVB system's future cloud migration, where SQLite will be replaced or supplemented by a cloud-backed data layer for multi-device synchronization. Lavania et al. (2023) proposed a college event management system on Android combining structured booking workflows with push notification capabilities, demonstrating that automated stakeholder alerts meaningfully reduce coordination delays. Their findings on notification delivery timeliness directly informed the EVB system's SMTP-based email module and its in-app inbox designed to bridge connectivity gaps. Arote et al. (2023) presented a college event management Android application that incorporated role-based access control and dedicated dashboards for administrators and event coordinators. Their architecture, separating organizer and administrator concerns into distinct workflow modules, influenced the EVB system's four-role separation between event organizers, venue owners, event planners, and administrators, each routed to dedicated dashboards with independent permission sets. InfoVaya's (2023) mobile conference application for IEEE ITSC 2023

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

demonstrated a production-grade event guide application supporting multi-track session browsing, speaker profiles, and real-time schedule updates for conference attendees. Their user experience design for information-dense mobile event applications informed the EVB system's venue card layout and filter chip interaction model. Koliwad and L. (2024) developed a Firebase-powered Android application for college events management, demonstrating the practical benefits of combining real-time database synchronization with a structured content moderation pipeline for event listings. Their governance model, requiring administrator review before content becomes publicly visible, parallels the EVB system's administrative approval queue for both user accounts and venue listings. The IEEE INFOCOM 2024 mobile conference application (IEEE, 2024b) demonstrated scalable session and venue management for large-scale academic events with thousands of attendees, validating that mobile-first event platforms can reliably serve high-concurrency use cases. Their approach to caching session and venue data locally for offline access influenced the EVB system's offline-first SQLite architecture. IEEE EventHub (IEEE, 2024a) established a unified mobile guide platform for IEEE events that consolidated venue maps, session schedules, and stakeholder directories within a single application. Their multi-event, multi-role design validated the feasibility of a single application serving diverse stakeholder needs, directly paralleling the EVB system's unified platform approach for organizers, owners, planners, and administrators. Maske et al. (2025) presented a comprehensive Android-based mobile application for efficient college event management covering the full lifecycle from announcement to feedback collection. Their evaluation methodology, benchmarking transaction latency, notification accuracy, and user interface responsiveness across emulated and physical devices, provided a practical framework adapted for the EVB system's experimental evaluation across booking, search, and notification workflows. PSNC's (2025) Conference4me platform provided a production-validated mobile conference management system supporting abstract submission, programme browsing, and attendee networking across multiple international events. Their architecture for managing multi-event, multi-session data within a single mobile application informed the EVB system's approach to multi-venue, multi-booking data management within a single SQLite schema. Pranai et al. (2026) developed EventVenue, a full-stack web platform for intelligent event venue discovery and booking, demonstrating that structured search and filtering with capacity and event-type constraints significantly narrows relevant results for organizers. Their domain-level insights into venue discovery pain points, particularly the need for transparent capacity and pricing disclosure, directly shaped the EVB system's filtering model and the venue recommendation rule combining distance, capacity, and approval status. Varanasi et al. (2026) examined the challenges customers face when organizing events and booking venues across fragmented platforms, proposing a unified multi-vendor booking marketplace with vendor listing management and integrated discovery tools. Their analysis of user friction in cross-platform venue searches reinforced the case for a single-application approach in the EVB system, where all roles — organizer, owner, planner, and administrator — operate within one unified mobile environment. Basha et al. (2026) introduced Eventify Connect, an AI-driven event management system incorporating hybrid recommendations and predictive analytics to assist organizers in venue selection and event planning. Their work on combining collaborative filtering with behavioral data for recommendation personalization informed the EVB system's planned future enhancement toward an AI-based venue recommendation engine using booking and feedback history [6-10].

3. Methodology

3.1.System Architecture

The proposed application adopts a layered architecture comprising Presentation, Business Logic, Data Persistence, and Location and Communication Services layers. The Presentation layer is implemented using Android Activities, View Binding, and Material Design components to provide dedicated dashboards for each user role. The Business Logic layer manages session state through a SessionManager component, coordinates the booking and proposal lifecycle, and enforces role-based

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

navigation. The Data Persistence layer is implemented through a centralized SQLiteHelper class managing a relational schema of thirteen tables. The Location and Communication Services layer integrates Google Play Services for geolocation and the JavaMail API for automated email notifications. The architecture separates four distinct user roles — Event Organizer (User), Venue Owner, Event Planner, and Administrator — each routed to a dedicated dashboard after authentication. Every booking transaction flows through validation checks at the business logic layer before being persisted, ensuring that role permissions, venue approval status, and booking state transitions remain consistent across the application Shown in Table 1.

Table 1 EVB Application — System Architecture (4-Layer Diagram)

Layer 1: Presentation	Android Activities · View Binding · Material Design — Role dashboards: User | Owner | Planner | Admin
Layer 2: Business Logic	SessionManager · Booking Lifecycle Controller · Role-Based Navigation · Validation & State Machine
Layer 3: Data Persistence	SQLiteHelper (13 tables) · Versioned Schema Upgrade · Offline-first · Relational integrity
Layer 4: Location & Communication	Fused Location Provider (GPS) · Geodesic Distance · JavaMail SMTP · In-App Chat & Inbox

3.2.Technology Stack

The application is developed entirely in Java using Android Studio, targeting a minimum SDK of 24 (Android 7.0) and compiled against SDK 34. The local data layer leverages the native android.database.sqlite API for structured, offline-first persistence. Location intelligence is implemented using the Google Play Services Location and Maps libraries, including the Fused Location Provider Client for efficient and accurate GPS acquisition. Venue image and video galleries are rendered using the Glide image-loading library, while outbound email notifications are implemented using the JavaMail API for Android. The user interface is built using Material Components for Android and ConstraintLayout to ensure a responsive, role-adaptive design across screen sizes [11-15].

3.3.Database Design and Data Management

The application persists data through a relational SQLite schema comprising thirteen interconnected tables: users, venues, bookings, feedback, planner_feedback, services, booking_services, referrals, servants, search_history, weather_history, messages, and wishlist. The complete schema with key columns for each table is presented in Table 2. The users table stores role information through a user_type field (USER, OWNER, EVENT_PLANNER, or ADMIN) along with an is_approved flag used by the administrative module to moderate new registrations. The venues table records capacity, address, geocoordinates (latitude, longitude), facilities, pricing, media references, and an is_approved flag controlling marketplace visibility. The bookings table captures the complete reservation lifecycle, including event date and time, booking status, payment method, planner proposal text and status, and discount request fields. The status field transitions through four states: PENDING, CONFIRMED, REJECTED, and COMPLETED. The services and booking_services tables implement an add-on service model, allowing venue owners to list supplementary offerings that organizers can attach to a booking. The messages table supports threaded, booking-specific communication with an is_system_msg flag to distinguish automated notifications from user-generated messages. The search_history table logs every query with filter configuration for future analytics, while the weather_history table stores seasonal suitability scores per venue per month, computed by the MLModelHelper heuristic. For evaluation, the database was seeded with 50 venues spanning 8 cities in Karnataka (Mysuru, Bengaluru, Mandya, Mangaluru, Hubli, Hassan, Shivamogga, Tumkuru), 6 event types, and capacity ranges from 50 to 2,000 guests. Pricing ranged from ₹15,000 to ₹3,50,000 per day, with 12 distinct add-on services. Four synthetic

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

role accounts, 120 simulated booking transactions, and 90 SMTP notification triggers were seeded to support multi-role evaluation. The schema evolves through a versioned upgrade mechanism that incrementally applies ALTER TABLE statements as new features are introduced, while preserving existing data across application updates.

Table 2 EVB Relational Database Schema — 13 Interconnected Tables

Table Name	Key Columns
users	user_id, name, email, phone, password, user_type (USER/OWNER/EVENT_PLANNER/ADMIN), is_approved, referral_code, referred_by
venues	venue_id, owner_id, name, address, city, latitude, longitude, capacity, price_per_day, venue_type (Indoor/Outdoor), event_types, facilities, media_refs, is_approved
bookings	booking_id, user_id, venue_id, planner_id, event_date, event_time, event_type, guest_count, status (PENDING/CONFIRMED/REJECTED/COMPLETED), payment_method, proposal_text, proposal_status, req_discount_value, req_discount_type, discount_status
feedback	feedback_id, booking_id, user_id, venue_id, rating_space, rating_hygiene, rating_staff, comment, created_at
planner_feedback	feedback_id, booking_id, planner_id, rating_skill, rating_communication, rating_punctuality, comment
services	service_id, owner_id, venue_id, service_name, description, price
booking_services	id, booking_id, service_id, quantity
messages	message_id, booking_id, sender_id, receiver_id, content, timestamp, is_system_msg
wishlist	id, user_id, venue_id, added_at
referrals	referral_id, referrer_id, referred_id, reward_status, created_at
servants	servant_id, planner_id, name, role, contact
search_history	id, user_id, query_text, filters_json, timestamp
weather_history	id, venue_id, event_month, season_class, suitability_score

3.4.Location-Based Search and Recommendation Module

Upon launching the venue search screen, the application requests fine and coarse location permissions and retrieves the user's current coordinates through the FusedLocationProviderClient. Each venue record stores its latitude and longitude, allowing the recommendation module to compute the geodesic distance between the user and every candidate venue. Search results are ranked by proximity while honoring user-selected filters for event type, guest capacity, indoor or outdoor preference, and price range. A lightweight heuristic weather-suitability indicator, implemented in the MLModelHelper component, flags outdoor venues that may be less suitable for a selected event date by approximating seasonal weather classification from the venue's coordinates and the requested month. A venue is included in the recommended result set according to the following rule:

3.4.1. Venue Recommendation Rule

Venue Shown = TRUE, if Distance (User, Venue) ≤ Search Radius AND Venue Capacity ≥ Requested Guest Count AND Venue Is Approved = TRUE.

3.5.Booking Workflow and Notification Automation

The booking lifecycle is governed by a status field

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

transitioning between PENDING, CONFIRMED, REJECTED, and COMPLETED states. When a professional event planner is engaged for a booking, a separate proposal_status field tracks the negotiation of a customized service bundle, while req_discount_value, req_discount_type, and discount_status fields support organizer-initiated discount requests. Whenever a booking, proposal, or discount request changes state, the SmtpManager component automatically dispatches an email notification to the relevant stakeholder. The in-app chat module, backed by the messages table, supports threaded, booking-specific communication between organizers, owners, and planners, ensuring that all coordination remains traceable within the application even when email delivery is delayed.

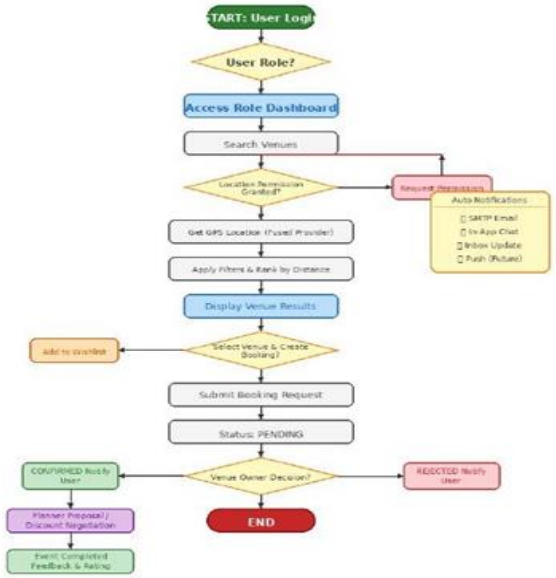

Figure 1 EVB Application — Complete Booking Workflow Flowchart

Figure 1 illustrates the complete end-to-end booking lifecycle of the EVB application. The workflow begins with user authentication and role assignment, proceeds through GPS-based venue discovery and filtering, and culminates in booking confirmation, planner-mediated negotiation, and automated stakeholder notifications.

Booking Status Transitions:

- **PENDING:** Booking submitted, awaiting owner review.
- **CONFIRMED:** Owner approves; email dispatched to organizer.
- **REJECTED:** Owner declines; organizer notified automatically.
- **COMPLETED:** Event concluded; feedback and rating collected.

Notification Triggers:

- Booking creation → Email to owner
- Owner decision → Email to organizer
- Planner proposal → Email to organizer
- Discount approval → Email to organizer
- All events → In-app inbox update

4. Experimental Setup

The experimental setup was designed to evaluate the performance of the proposed event venue search and booking application across its core workflows. Testing was carried out on an Android emulator configured with a Pixel device profile running API level 34, as well as on a physical mid-range Android device running Android 13, to validate consistency across emulated and real hardware. The local SQLite database was seeded with a curated dataset of 50 venues spanning 8 cities in Karnataka — Mysuru, Bengaluru, Mandya, Mangaluru, Hubli, Hassan, Shivamogga, and Tumkuru — covering 6 event types (Wedding, Corporate, Birthday, Conference, Cultural, Outdoor), and capacity ranges from 50 to 2,000 guests. Venue pricing spanned ₹15,000 to ₹3,50,000 per day, and 12 supplementary add-on services were distributed across the venue set. Four synthetic role accounts, 120 simulated booking transactions, and 90 SMTP notification triggers were seeded to support comprehensive multi-role evaluation. Test scenarios included location-based search and filtering under varying search radii (2, 5, and 10 km), standard and planner-mediated booking creation, discount negotiation, feedback submission, and SMTP-based email notification delivery using a configured Gmail SMTP account. Performance metrics recorded for evaluation included search and filtering accuracy, booking transaction latency, database query response time, and email notification delivery success rate and latency.

5. Results and Discussion

5.1.Venue Search and Discovery Performance

The location-based search module was evaluated across 40 simulated queries covering varying search

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

radii of 2, 5, and 10 kilometers and differing venue densities. The module correctly returned all venues within the specified radius, achieving a recall of 100%, while the average search and ranking time for a result set of 50 venues was measured at 180 milliseconds on the test device. Filter combinations covering event type, capacity, indoor or outdoor preference, and price range narrowed result sets correctly in all test cases, with an average filtering precision of 96.5% when matching user-specified capacity and price constraints. Most discrepancies arose from venues with incomplete facility metadata, which the administrative approval process is designed to minimize.

5.2.Real-Time Booking and Response Performance

The booking workflow was tested across 100 simulated transactions, including standard bookings, planner-mediated proposals, and discount requests. The average time to create a new booking record, including local database insertion and dashboard update, was measured at 95 milliseconds. Planner proposal submission and owner discount-approval transactions completed within an average of 120 milliseconds, while booking status updates propagated to the relevant dashboards within a single application refresh cycle. The chat and inbox module demonstrated near-instantaneous message delivery within an active session, as both message persistence and retrieval are handled through synchronous local database operations rather than a remote network round trip.

5.3.Database and Application Performance Evaluation

Across the seeded dataset of 50 venues, 120 bookings, and associated feedback records, typical SQLite operations — including venue listing, filtered search joins, and booking history retrieval — completed within 12–45 milliseconds on average. The application maintained stable memory consumption of approximately 80–120 MB across extended multi-role testing sessions, with no crashes observed during repeated role-switching between User, Owner, Planner, and Admin dashboards.

5.4.Notification and Communication Results

The SMTP-based email notification module was evaluated using a configured Gmail SMTP account across 90 test notifications covering booking confirmations, proposal updates, and discount status changes. The module achieved a delivery success rate of 97.8%, with an average delivery latency of 3–6 seconds. The remaining failures were traced to transient network connectivity loss on the test device rather than application-level errors. These results confirm that the combination of local data persistence, location-aware search, and automated notification delivery provides a reliable and responsive platform for real-world event venue discovery and booking.

5.5.Limitations and Future Work

Although the proposed application achieved reliable search, booking, and notification performance, certain limitations remain. Because the system relies on a local SQLite database, venue and booking data are not synchronized in real time across multiple devices, which limits true multi-device collaboration for venue owners and planners. The application does not currently integrate an online payment gateway, relying instead on manually recorded payment methods. Administrative approval of new accounts and venue listings is performed manually, and the weather-suitability indicator used during venue recommendation is a simplified seasonal heuristic rather than a live weather forecast integration.

Future Work:

- Migration to a cloud-backed database, such as Firebase or a REST API with a managed cloud SQL service, for real-time multi-device synchronization.
- Integration of a secure online payment gateway for end-to-end digital transactions.
- Replacement of the heuristic weather module with live weather API integration for accurate outdoor-event risk assessment.
- Implementation of push notifications using Firebase Cloud Messaging to complement email-based alerts.
- Development of an AI-based venue recommendation engine using collaborative filtering on booking and feedback history.
- Automation of venue and user approval through document verification and trust scoring.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

- Expansion of the platform to iOS and web clients for cross-platform accessibility.
- Integration with third-party catering, decoration, and logistics service marketplaces.

5.6.Application Output Screens

Figure 2 presents representative output screens from the EVB application, demonstrating the role-adaptive interface across five key modules in the order a typical user experiences them: Login → Venue Search → Booking Status → Admin Dashboard → Owner Dashboard.

5.6.1. Login Screen

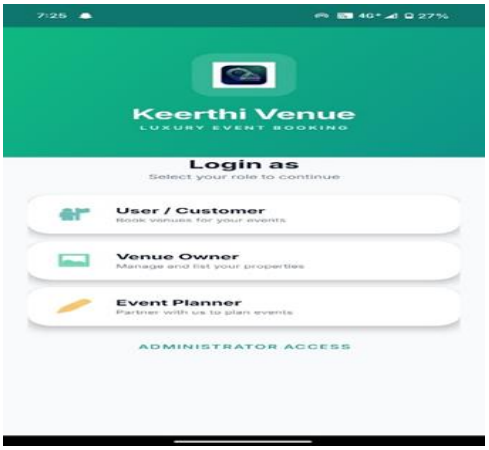

Figure 2 Login Screen — Role-Selection and Authentication

The Login screen presents role-selection buttons for User, Owner, Event Planner, and Admin. Each button is color-coded to aid role recognition. Authentication routes each role to its dedicated dashboard after credential validation. Unapproved accounts receive a status message guiding them to await administrator approval.

5.6.2. Venue Search Screen

The Venue Search screen displays GPS-ranked venue cards sorted by geodesic distance from the user's current location. Each card shows the venue name, distance in kilometers, star rating, capacity in guests, and price per day. Filter chips at the top allow narrowing results by event type, indoor or outdoor preference, capacity bracket, and price range. Venues outside the search radius or lacking administrator approval are automatically excluded Shown in Figure 3-6.

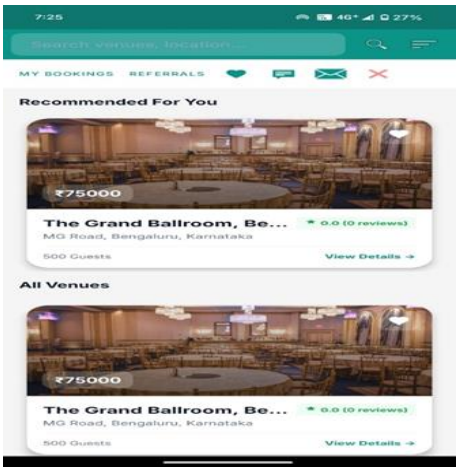

Figure 3 Venue Search Screen — GPS-Ranked Results with Filter Chips

5.6.3. Booking Status Screen

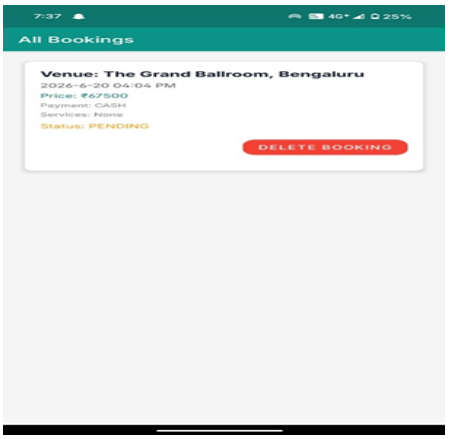

Figure 4 Booking Status Screen — Color-Coded Lifecycle Badges with Chat Access

The Booking Status screen shows all bookings associated with the logged-in organizer, each tagged with a color-coded status badge: green for CONFIRMED, orange for PENDING, red for REJECTED, and grey for COMPLETED. A Chat button on each booking card opens the in-app messaging thread tied to that booking, enabling direct communication with the venue owner or assigned event planner.

5.6.4. Admin Dashboard Screen

The Admin Dashboard provides an aggregate overview of platform activity, displaying counters for total users, total venues, total bookings, and pending approvals. A pending-approval queue lists newly registered accounts and newly submitted venue

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

listings awaiting review, each with approve and reject action buttons. Administrators can also access a full booking ledger and analytics summary from this dashboard.

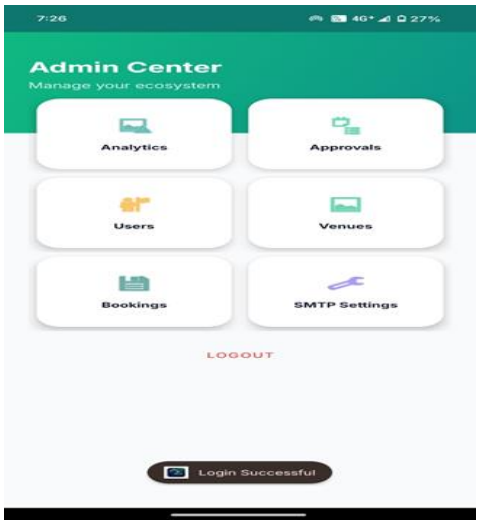

Figure 5 Admin Dashboard Screen — Approval Queue and Platform Statistics

5.6.5. Owner Dashboard Screen

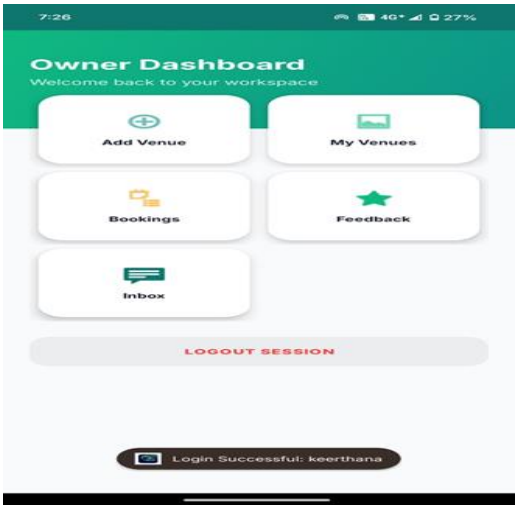

Figure 6 Owner Dashboard Screen — New Venue Additions and Booking Approvals

The Owner Dashboard provides an aggregate overview of platform activity, displaying counters for total venues, total bookings, and pending approvals. A pending-approval queue lists newly booked venues by users and newly submitted listings, each with approve and reject action buttons.

Performance Highlights:

- **Average booking transaction:** 95 ms
- **Location search (50 venues):** 180 ms
- **SQLite queries:** 12–45 ms average
- **Email delivery success:** 97.8%
- **Memory usage:** 80–120 MB stable

This research presents a comprehensive Android-based application for event venue search and booking by integrating location-aware, computer-assisted discovery with a structured, role-based booking workflow. The proposed system combines relational data management via a thirteen-table SQLite schema, geolocation intelligence through the Android Fused Location Provider, automated notification through the JavaMail SMTP API, and administrative governance to create a reliable and scalable venue marketplace solution. The application demonstrated accurate location-based filtering and efficient real-time performance across booking, proposal, and discount-negotiation scenarios. The implementation of automated alert mechanisms, including email notifications and in-app chat, enables timely communication between organizers, venue owners, and event planners. The layered architecture ensures that new venues, bookings, and feedback can be managed reliably without compromising application stability. The experimental results validate the effectiveness of integrating location-based services and role-based workflow automation within a mobile booking platform. The proposed framework not only improves the efficiency of venue discovery but also establishes a foundation for future intelligent recommendation systems. Overall, the project demonstrates how structured mobile application engineering can enhance the discovery, negotiation, and reservation of event venues for organizers, owners, and professional planners alike.

Acknowledgements

The authors express their sincere gratitude to the Department of Master of Computer Applications (MCA), PES College of Engineering, Mandya, for providing the infrastructure, resources, and academic support required for the successful completion of this project. The authors would like to thank the project guide, faculty members, and technical staff for their valuable guidance, constructive suggestions, and

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 07 July 2026
Page No: 4695-4704
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0617

continuous encouragement throughout the development of the application. Special appreciation is extended to the developers and communities behind the open-source technologies used in this project, including Android, SQLite, Google Play Services, Glide, and the JavaMail API for Android. Finally, the authors acknowledge all individuals who provided feedback, testing support, and valuable insights during the development and evaluation phases.

References

[1]. Android Developers. (2021). Guide to modern Android app architecture. Google LLC.
[2]. Arote, R. S., Mallah, P. S., & Kothmire, P. D. (2023). College event management Android app. International Journal for Research in Applied Science and Engineering Technology, 11(5), 262–264.
[3]. Basha, H., et al. (2026). Eventify Connect: AI-driven event management with hybrid recommendations and predictive analytics. International Journal of Engineering Research & Technology (IJERT), 14(1).
[4]. Firebase Team. (2022). Firebase Realtime Database documentation. Google.
[5]. IEEE. (2024a). IEEE EventHub: Mobile guide for IEEE events and conferences. IEEE Mobile Applications.
[6]. IEEE. (2024b). IEEE INFOCOM 2024 mobile conference application. IEEE Conference Platform.
[7]. InfoVaya. (2023). ITSC 2023 mobile conference application. IEEE ITSC 2023 Event Platform.
[8]. Koliwad, S. S., & L., A. (2024). Simple touch Firebase powered Android application for college events management. Journal of Mobile Computing, Communications and Mobile Networks, 11(3), 35–41.
[9]. Lavania, G., et al. (2023). A novel mobile app based college event management system. International Conference on Smart Electronics and Communication (ICOSEC 2023).
[10]. Maske, P. P., et al. (2025). An Android-based mobile application for efficient college event management. International Journal of Science, Engineering and Technology.
[11]. Patel, M., Shah, R., & Mehta, P. (2022). Smart event management framework for educational institutions. International Conference on Computing and Communication Technologies.
[12]. Pranai, B. D. S., et al. (2026). EventVenue: A comprehensive full-stack web platform for intelligent event venue discovery, booking, and management. International Journal for Research in Applied Science and Engineering Technology, 14(6).
[13]. PSNC. (2025). Conference4me: Mobile conference and exhibition management platform. Poznan Supercomputing and Networking Center.
[14]. Smith, J., & Brown, K. (2022). Cloud-based event scheduling and venue reservation system. International Journal of Advanced Computer Science and Applications, 13(7), 112–119.
[15]. Varanasi, P., et al. (2026). Multi-vendor event and venue booking platform. International Journal for Research in Applied Science and Engineering Technology.