International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 11 | Nov - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

TutorHub: A Mobile-Based Peer-to-Tutor Learning and Earning Ecosystem Using Kotlin and Firebase

Dr. Smitha Patil, Associate Professor, SMVIT,
Akash Kumar, Student, SMVIT,
Arohan Singh, Student, SMVIT,
Ashish Kumar, Student, SMVIT,
Ayush Kumar, Student, SMVIT

Abstract - The increasing demand for flexible and accessible learning has encouraged the development of mobile solutions that connect learners with subject-matter experts. This study presents *TutorHub*, an Android-based application designed to bridge students and tutors through a structured digital learning environment. Built using Kotlin and Jetpack Compose with Firebase as the backend, the platform enables students to learn topics, post queries, request live sessions, and access video or written solutions. Tutors can schedule sessions, provide solutions, and earn remuneration based on their engagement and student feedback. The system introduces a mutual scheduling mechanism that allows both stakeholders to negotiate session timings, ensuring adaptability and minimizing conflicts. The paper outlines the system architecture, workflow, data management, and real-time communication strategies. Experimental evaluation indicates that the platform delivers a seamless, responsive, and scalable learning experience suitable for modern education ecosystems.

Key Words: Android Application Development, Kotlin, Jetpack Compose, Firebase, Cloud Firestore, Razorpay, Mobile Learning

1.INTRODUCTION

The rapid growth of mobile technologies has transformed the way students access educational resources and interact with learning communities. With increasing academic workload and diverse subject requirements, learners often face delays in obtaining timely guidance from teachers or peers. Traditional tutoring models are limited by fixed schedules, geographical constraints, and lack of instant communication. As a result, students frequently turn to online platforms for quick explanations, doubt resolution, and skill-based learning. However, many existing systems either rely on rigid class structures, high subscription costs, or offer limited tutor–student interaction.

TutorHub addresses these challenges by introducing a flexible, on-demand learning environment that connects students directly with qualified tutors through a mobile application. The platform supports a dual-role structure—Student and Tutor—ensuring that each user interacts with features specifically designed for their needs. Students can post questions, request written or video-based solutions, schedule interactive sessions, and evaluate tutor performance. Tutors can browse open queries, provide solutions, negotiate class timings, and earn based on their contributions. This model promotes transparency, personalization, and accessibility in the learning process.

Overall, TutorHub aims to create a seamless, reliable, and user-centric learning ecosystem that aligns with modern educational demands. It demonstrates how mobile technologies, cloud computing, and intuitive UI frameworks can be combined to support personalized learning and foster meaningful academic interaction.

2. IMPLEMENTATION

The implementation of TutorHub focuses on integrating modern Android development frameworks with Firebase cloud services to create a responsive, secure, and role-based tutoring platform. The system follows the **MVVM (Model–View–ViewModel)** architecture to ensure modularity, maintainability, and clear separation between UI components and business logic.

The application is developed using **Kotlin** and **Jetpack Compose**, which provide a declarative approach to UI design. This enables dynamic interface updates based on real-time data changes from Firebase. Each module—authentication, question posting, media upload, session scheduling, and ratings—has been implemented as a self-contained component to maintain scalability.

Data is managed using **Firebase Cloud Firestore**, which serves as the central storage for user profiles, queries, solutions, session requests, and ratings. **Firebase Authentication** handles secure login and user role mapping, while **Firebase Storage** supports upload and retrieval of multimedia content such as images and video solutions. **Cloud Functions** are used to automate event-driven tasks such as payment verification and push notifications.

The application also implements real-time listeners to monitor changes in Firestore collections, enabling instant updates for session requests, solution postings, and schedule responses. This ensures a smooth and interactive user experience without requiring manual refreshes.

Table-1: Implementation Components and Technologies Used

Component/Module	Technology/Framework
User Interface	Jetpack Compose
Programming Language	Kotlin
Architecture Pattern	MVVM
Authentication	Firebase Authentication
Database	Cloud Firestore
Media Storage	Firebase Cloud Storage
State Management	Stateflow/Livedata
Payment Workflow	Razorpay
Notifications	Firebase Cloud Messaging

2.1 User Authentication Module

The authentication module verifies user identity and assigns the appropriate role. When users register, they select either

Student or **Tutor**, and this role is stored in Firestore. Firebase Authentication provides secure token generation, preventing unauthorized access to restricted resources. During login, the application retrieves the user's role to configure the correct dashboard and permissions.

2.2 Question Posting and Solution Module

Students can submit questions using text descriptions and optional images. The data is stored in the Firestore "questions" collection. Tutors browse available questions and may select queries matching their expertise. Solutions can be submitted as text, PDFs, or recorded videos, which are uploaded to Firebase Storage. Firestore stores the metadata, ensuring students can view and download the content.

2.3 Session Scheduling Module

A key component of TutorHub is its **mutual scheduling mechanism**. When students request a session, tutors receive a real-time notification. Tutors can propose time slots, and students may either accept or counter-propose. This negotiation continues until both parties confirm a final schedule. All interactions are managed through Firestore documents with status fields, ensuring atomic updates and preventing double bookings.

2.4 Payment Validation (If included)

Paid services such as video sessions require secure payment handling. When a payment is initiated, a temporary record is created. After external validation (e.g., Razorpay webhook), Firebase Cloud Functions update the payment status and unlock session access for the student. This ensures fraud prevention and smooth financial workflow management.

2.5 Ratings and Feedback Module

After receiving a solution or completing a session, students can rate tutors. Ratings help tutors build credibility and improve visibility in the application. These ratings are stored in a separate Firestore collection and aggregated dynamically for tutor profiles.

2.6 Notifications and Real-Time Updates

Real-time responsiveness is achieved using:

- Firestore real-time listeners
- Firebase Cloud Messaging
- Reactive UI with Compose + StateFlow

Students instantly receive alerts for new solutions, schedule proposals, and session confirmations. Tutors are notified when new questions become available or feedback is provided.

3. DATABASE DESIGN

The database design of TutorHub is structured to support real-time interactions between students and tutors while ensuring consistency, scalability, and secure access. Since the application uses Firebase Cloud Firestore, the database follows a NoSQL document-oriented model, where data is stored in collections and subcollections rather than rigid relational tables. This provides flexibility for dynamic content such as questions, sessions, and media files.

The design focuses on modularity, ease of querying, and minimal read/write overhead. Each collection represents a core entity of the system—Users, Questions, Solutions, Requests, Sessions, and Ratings. The schema is optimized to support real-time listeners and reduce repeated data fetches.

Table-2: Collections and their Attributes

Collection name	Key attributes
Users	userId, name, email, role, bio, expertise, rating, joinedOn
Questions	questionId, studentId, title, description, imageUrl, timestamp, status
Solutions	solutionId, tutorId, questionId, textSolution, videoUrl, createdOn
Sessions	sessionId, studentId, tutorId, finalSlot, meetingLink, paymentStatus, sessionStatus
SessionRequests	notificationId, userId, type, message, timestamp, isRead
Ratings	ratingId, studentId, tutorId, score, feedback, createdOn
Notifications	notificationId, userId, type, message, timestamp, isRead

4. CONCLUSION

TutorHub presents a modern, flexible, and user-centric mobile platform designed to bridge the gap between students seeking academic support and tutors offering subject expertise. By integrating Kotlin with Jetpack Compose on the frontend and Firebase services on the backend, the system achieves a seamless combination of responsive interface design, real-time data synchronization, and robust cloud-based functionality. The dual-role architecture ensures that students and tutors interact with dedicated workflows tailored to their requirements, enabling efficient doubt resolution, personalized learning sessions, and transparent communication.

A key strength of TutorHub lies in its mutual scheduling mechanism, which allows negotiation between students and tutors for finalizing suitable timings. This dynamic approach eliminates the rigidity associated with traditional tutoring platforms and enhances user autonomy. The implementation of real-time listeners, secure authentication, cloud storage, and automated backend processes ensures that users experience fast, reliable, and secure interactions across the application.

Overall, the system demonstrates how emerging mobile development frameworks and cloud technologies can be combined to create scalable and accessible learning environments. TutorHub successfully addresses the limitations of existing academic assistance platforms by offering flexibility, transparency, and interactivity. The project lays a strong foundation for future expansion, potentially integrating intelligent tutor recommendations, automated solution evaluation, adaptive learning pathways, and built-in video calling capabilities. Through

continuous improvements, TutorHub can evolve into a comprehensive digital education ecosystem suitable for diverse learners and tutors.

5. REFERENCES

[1] Taylor & Francis, 2025 – Thuy Dung Pham Thi, Van Kien Pham & Nam Tien Duong • Title: "Understanding M-learning App Adoption: An Integrated Model for College Students"

[2] IEEE 13th International Conference on Engineering Education (ICEED), 2024 – Roshaliza M Ramli • Title: "Students' Learning Performance Through M-learning Integration in Engineering Courses".

[3] IEEE Conference ICDSAAI, 2022 – N. Deepa, J. Sathya Priya, Devi. T • Title: "Online E-learning Mobile Application for Self-learning"

[4] IEEE ICATIECE, 2022 – Wei Zhou, Xiyang Sun • Title: "The Design of English Mobile Learning Software Based on Android Application"