UDC 004

CROSS-PLATFORM ONLINE REGISTRATION SYSTEM BASED ON FIREBASE AND KOTLIN FOR THE SERVICES INDUSTRY

Kotvytska A., Hrama M.
National University of Food Technologies, Kyiv, Ukraine
E-mail: kotvyckaaa@nuft.edu.ua

Effective management of customer flow is a critical task for modern service sector enterprises, including service stations, beauty salons, and educational centers. The report considers the development of a cross-platform online booking system using Kotlin Multiplatform and Firebase cloud services. The specific features of real-time data synchronization, user authentication, and transaction handling to prevent booking conflicts are analyzed. The proposed architecture ensures high development efficiency and cross-platform native performance.

The modern service market requires fast and reliable tools to automate customer interaction. An online booking system that minimizes the human factor and optimizes staff work schedules. Developing separate native applications for Android and iOS is often economically impractical for small and medium-sized businesses. Therefore, a relevant solution is the use of cross-platform technologies, in particular Kotlin Multiplatform (KMP), in combination with flexible cloud backend platforms such as Firebase [1]. One of the advantages of Kotlin Multiplatform is the ability to share business logic (data validation, network work, state preservation) between different operating systems without losing native performance and flexibility of the UI [2]. Unlike hybrid frameworks, KMP does not use intermediate layers (WebViews or JavaScript bridges), which ensures a high UI response speed, which is critical for a comfortable user experience when choosing services or free time.

For the implementation of the server part of the online booking system, the optimal choice is the Firebase platform, which provides a full range of Backend-as-a-Service (BaaS) services. With BaaS, development is significantly accelerated, as it eliminates the need to build and manage traditional server infrastructure from scratch. Firebase offers out-of-the-box solutions for secure user authentication, cloud hosting, serverless computing via Cloud Functions. These serverless features are particularly valuable for booking, as they can seamlessly handle background tasks (sending confirmation emails, validating reservations, executing automatic status updates).

The role of the data storage in such an architecture is performed by Cloud Firestore, a flexible NoSQL database designed for global scalability and high performance. The main feature of Firestore is support for real-time synchronization using the subscription model (data streams) [3]. This allows you to instantly update the lists of available time slots on all client devices: as soon as one user takes a certain time, this slot is automatically blocked for others without the need to manually refresh the application page. Furthermore, Firestore includes robust security rules to protect sensitive user data and built-in offline support, which ensures a seamless experience even in environments with an unstable internet connection. However, when developing online booking systems, a serious problem of competitive

access (Race Conditions) arises when 2 or more users try to book the same time at the same time. To prevent conflicts in the system, Firestore transactions are used [4]. The transaction performs an atomic check: it reads the current status of the time slot and, if it remains free, changes its status to "booked". If, during the transaction, the data on the server was changed by another client, the system automatically cancels the operation and offers the user to choose another time, guaranteeing the DB integrity.

To identify users and manage access control securely, the Firebase Authentication module is integrated. This service provides a comprehensive identity solution that supports secure login via traditional methods like phone numbers (SMS verification) and email, or through federated identity providers such as popular services (Google, Apple ID) [5]. This simplifies the client registration process to a few clicks, reducing onboarding friction and improving the overall user experience.

Furthermore, by leveraging industry standards like OAuth 2.0 and OpenID Connect, Firebase Authentication ensures that sensitive credentials are processed securely. Once a user successfully logs in, the system generates a secure token to manage their session. This mechanism integrates seamlessly with Cloud Firestore, allowing the implementation of strict database security rules so that clients can only view and manage their personal booking records. Offloading these responsibilities to Firebase also eliminates the need to manually develop complex backend logic for password hashing, account recovery flows, protection against unauthorized access.

An additional advantage of the selected technology stack is the built-in offline support in the Firebase SDK. If the user temporarily loses connectivity (for example, while traveling or in an underground parking lot), the application continues to work with a local cache. All created requests are recorded in a local queue and automatically synchronized with the cloud immediately after a stable Internet connection is restored. To retain customers and remind them of scheduled visits, the system uses the Firebase Cloud Messaging (FCM) service, which implements sending targeted push notifications. Thus, the synergy of Kotlin Multiplatform and Firebase services allows you to create a reliable, scalable and cost-effective cross-platform online booking system. This reduces development time (Time to Market) by almost half compared to native development for each platform separately, while ensuring uninterrupted data synchronization in real time and a high level of security.

References

1. JetBrains (2024) *Kotlin Multiplatform Development Fundamentals* [online]. URL: https://kotlinlang.org/docs/multiplatform.html.
2. Sadavin A. (2023) Architecture patterns for Kotlin Multiplatform projects, *Journal of Software Engineering*, vol. 14, no. 2, pp. 45–52.
3. Google Firebase (2025) *Cloud Firestore: Update data with transactions* [online]. URL : https://firebase.google.com/docs/firestore/manage-data/transactions.
4. Moroney L. (2021) *Definitive Guide to Firebase: Build Apps Quickly with Firebase*. Berkeley: Apress, 320 p.
5. Google Firebase (2024) *Firebase Authentication Overview* [online]. URL : https://firebase.google.com/docs/auth.