STUDENT SESSION

AGILE MULTI-USER ANDROID APPLICATION DEVELOPMENT WITH FIREBASE: AUTHENTICATION, AUTHORIZATION, AND PROFILE MANAGEMENT

Katarina Milojković*,
[0000-0001-8658-8973]
Miodrag Živković,
[0000-0002-4351-068X]
Nebojša Bačanin Džakula
[0000-0002-2062-924X]

Singidunum University,
Belgrade, Serbia

Abstract:

A tight project deadline can affect the quality of the application. In today's dynamic environment, there is a growing need for agile projects where the development team focuses on prioritizing and implementing functionalities that deliver maximum value to users or clients. The first step towards achieving agility in development is by integrating Firebase, a mobile and web application development platform, that offers a set of tools and services to help streamline the development process. The goal of this research is to share insights and knowledge regarding the utilization of Firebase in the development of a mobile application. Employing a research design methodology, specifically case study design, the research showcases an Android Studio application that uses Firebase to synchronize data in real-time, implement secure user authentication, have a scalable and serverless backend infrastructure, and securely store user-generated content. Key components such as Firebase Authentication, Cloud Firestore, and Firebase Storage play pivotal roles in creating the application's functionalities: authentication, authorization, and profile management. The research contributes by presenting a solution that streamlines backend infrastructure complexities and eliminates the need for a dedicated server infrastructure. This results in developers having more time to develop functionalities and work on UI/UX design. Furthermore, the straightforward Firebase tools and SDKs facilitate integration and enhance development agility, making it easily replaceable with an original solution or another backend service.

Keywords:

Agile Application Development, Firebase Authentication, Firebase Cloud Firestore, Firebase Cloud Storage, Android Studio.

INTRODUCTION

Modern business conditions require development teams to adopt agile practices in project management and software development in order to survive, thrive, and remain competitive in a turbulent and dynamic business environment. As a result, they produced an agile product.

In the fast-paced digital age, developers frequently encounter challenges such as the pressure to swiftly deliver applications that meet specific project requirements within tight deadlines. Reasons for this can be either personal, societal, or work-related. For example, a situation when investors prioritize getting quick and tangible outcomes rather than waiting for a fully developed product. Or when employers are unwilling to wait for several months or years for a developer to create

Correspondence:
Katarina Milojković

e-mail:
katarina.milojkovic.21@singimail.rs

DOI: 10.15308/Sinteza-2024-405-412

their desired product. A solution would be adopting agile methodologies for promptly creating desired products with a future plan for enhancements and maintenance. Additionally, another common scenario would be losing a competitive advantage by not efficiently transforming ideas into reality. Development speed will keep the product relevant, original, and interesting.

Operating and developing products in an agile manner means, above all, quickly responding and adapting to changes, thorough planning, respecting set deadlines, listening to and incorporating feedback from end users, aiming for efficiency and effectiveness, focusing on delivering value incrementally, and monitor the competition to identify opportunities and threats in order to stay competitive, improve, innovate and be relevant in the market. The emphasis is on quality, speed, and bringing value to the end user.

For the specific task of creating an Android Studio application, a solution to streamline the building process and enhance agility lies within using Firebase. This NoSQL and cloud-based platform offers a wide range of services, tools, and features accessible through the Firebase SDK (Software Development Kit) and API (Application Programming Interface) allowing developers to build, deploy, and manage their applications efficiently. Firebase's backend services such as Authentication, Cloud Firestore, and Cloud Storage provide a scalable and serverless backend infrastructure that reduces the need to build from scratch, manage, scale, and maintain core project functionalities like user authentication, authorization, database management, and file storage. Mentioned services can be seamlessly integrated with one another. While Firebase handles the backend infrastructure and services, developers can concentrate more on improving UI (User Interface), UX (User Experience), and implementing features.

The research showcases a soft project called 'Zero Food Waste', whose contribution is not only a tangible product, an Android Studio application but also a contribution to the social-economic development of the local community by reducing food waste and environmental pollution. The application demonstrates the use of the Firebase platform for multi-user access management, primarily profile management, authentication, and authorization.

In the 'Literature review' section, relevant research was summarized and presented to provide a deeper insight into this research. This section serves as a foundation for understanding the usage, advantages, and disadvantages of Firebase. The methodology used for this research is thoroughly explained in the 'Methodology' section, while the implementation of Firebase services for fulfilling set project requirements and the results of that implementation are shown in the 'Implementation and results' section. The 'Discussion' section outlines the results of the SWOT (Strengths, Weaknesses, Opportunities, and Threats) analysis, and the 'Conclusion' section presents the conclusions drawn from the research. The goal of this research is to share insights and knowledge regarding the utilization of Firebase in the development of a mobile application.

2. LITERATURE REVIEW

Firebase is Google's Internet of Things (IoT), mobile, and web development platform, created to help in the development process of IoT, iOS, Android, and Web applications. Despite being an extremely useful tool for creating IoT applications, it provides basic server-side processing and limited query functions. [1] The platform also provides space for storing data offline and updates it automatically after having a network connection. [2]

Firebase being a Backend-as-a-Service (BaaA), authors Dhanush, Rathi, Harini, Teja, and Kumar [3] pointed out that developers do not need to manage servers or write APIs. The reason behind this is that the platform is created as a server, the API, and the database, written in a way that can be modified based on different needs.

Firebase services like A/B testing, statistics, authentication, cloud messaging, crashlytics, dynamic links, invites, performance monitoring, analytics, remote configuration, real-time databases, cloud storage, and Firestore accessed through Firebase APIs or Firebase SDKs offer a variety of unique features. Services available through Firebase can be utilized within a mobile application tailored to meet the requirements of clients. Both options are available for iOS, Android, and web. [1] [4]

When considering the benefits of Firebase services for the application development process, according to the research of Hossain, Ullah, and Haque, it is evident that all those services make Firebase a powerful and invaluable tool for researchers and building applications. [5]

2.1. FIREBASE AUTHENTICATION

Verification is efficiently managed in Firebase through an authentication service called Firebase Authentication. The service ensures secure login processes and by tracking those processes, developers can improve login and boarding user experience. It serves as an all-in-one identity solution, supporting various authentication methods such as email/password authentication, phone authentication, and social logins like Google, Facebook, GitHub, and Twitter. The authentication is through client-side code or an existing login server. [1] [6]

User management features enable developers to implement authentication with email and password, securely storing user data in Firebase. Mohd Harith, Ramli, and Mohd Muji [6] further explained in their research that user-submitted information is seamlessly sent to the Firebase backend service. Firebase Authentication displays a list of registered users, providing user IDs and account creation timestamps that ensure a comprehensive record of user information in Firebase.

Security rules govern user access and actions, ensuring the integrity and confidentiality of stored data. Read or write access to the site is automatically provided to new applications. Therefore, according to the research of Chougale, Yadav, and Gaikwad [1], developers should review and adjust rules to limit or grant access and maintain security against potential attacks. In the relevant literature, researchers Rahman, Sarkar, and Biswas [7] concluded that the Firebase Authentication system is great and secure because even the admin cannot know the password of the user registered using the software.

2.2. CLOUD FIRESTORE

Storage, real-time synchronization, automatic scaling, offline support, and querying for retrieving, sorting, and filtering data are features provided in Firebase through a cloud-based NoSQL serverless database with real-time notification capability called Firestore. The service simplifies common challenges in app development by allowing developers to focus on business logic and user experience without worrying about database configuration details. Firestore working together with Firebase client-side SDK libraries is operating offline, continuing when reconnecting with the Internet. Having a schemeless structure, Firestore organizes data with documents and collections. Each document is a JSON object containing fields and values, while collections group related documents.

The Firestore database is only limited by the cloud region's data center physical constraints. Notable users of Firestore are the New York Times, BeReal, social media applications, and mobile games with over a hundred million users. [4] [8]

2.3. CLOUD STORAGE FOR FIREBASE

Solution for cloud storing and retrieving user-generated data, such as images, audio, and videos is offered in Firebase through a powerful and flexible cloud storage service called Firebase Storage. The service can be accessed from anywhere with an internet connection and is automatically scalable, supporting files of any size. An advantage of Firebase Storage is having security features, including access control rules, enabling developers to restrict access to specific files or folders and giving developers the ability to focus on upgrading user experience without managing server infrastructure. Firebase Storage has a storage system where files are organized in buckets. Developers can put files in buckets based on criteria like user or file type. Buckets are accessible through unique URLs as well as uploaded files. In order to upload a file, you need to create and use a reference to the file's location within the bucket. The resumable uploads feature of Firebase Storage ensures continuity even if interruptions occur during the upload process or network problems. [4] [9]

2.4. SOFT PROJECTS AND AGILE

The difference between hard and soft projects is mentioned by Milojkovic [10] in her research. The main result of hard projects is a material asset or a tangible product, whereas soft projects produce a series of causal or interactive activities achieving something such as establishment, relocation, arrangement, and organization.

The case study research conducted by Milojkovic and Milojkovic [11] indicated the ability of Agile methodology to greatly benefit businesses striving to achieve customer satisfaction and exceptional quality. Agile guides the focus of teams on operational, technical, and design improvements of their products. Having an Agile workflow means following planned activities, time, and budget, as well as continuing with activities that improve and evaluate the performance of project outcomes. The advantages of using Agile are always trying to find efficient solutions, simplifying parallel tasks, eliminating procrastination, and saving time.

When considering Firebase contributing to agile process implementation, according to the research of Sharma and Dand [12], powerful application companies are using Firebase just because it is perfectly in line with agile development. By doing so, they improved user interaction and their real-time updates. Padme [13] stated in his research that the platform can elevate data synchronization, performance, responsiveness, and overall quality of Android applications. By combining Firestore and Android UI/UX, developers can develop responsive, interactive, and engaging UI.

3. METHODOLOGY

Through a detailed analysis of project documentation, the subject of this research is clearly demonstrated and explained through a practical soft project. The SWOT analysis was used to evaluate the strengths, weaknesses, opportunities, and threats of implementing Firebase in the Android Studio application.

In conducting this research, a case study research design based on the practical example of the mobile Android Studio application "Zero Food Waste" was employed. The used methodology points out problems in the development process and how can they be solved with Firebase Authentication, Cloud Firestore, and Firebase Storage. The case study of the project is composed of application code analysis, developer's observations and notes during the build phase, and project work that includes user requirements specification, software development methodology, system analysis, and presentation layer design.

4. IMPLEMENTATION AND RESULTS

The Android application “Zero Food Waste” is a tangible product of the corresponding student project. It was agilely developed during the intensive University course “Mobile Application Development” with Android Studio and Firebase. The vision of this project is to reduce food waste in gastronomy objects and increase awareness of the problem.

4.1. ANDROID STUDIO PROJECT SETTINGS

The Integrated Development Environment (IDE) used to develop the "Zero Food Waste" application is Android Studio version Hedgehog|2023.1.1. The project was created as an “Empty Vews Activity” with the programming language set to Java. The minimum SDK version for the application is set to Lollipop 5.0, and the build configuration language is recommended Kotlin DSL (build.gradle.kts). For visual representation and testing of the application, the Android Emulator virtual device was utilized. The hardware configuration of the virtual device is Pixel with Play Store option, 5.0" screen size, 1080x1920 resolution, and 420dpi density. The system image used is Nougat with API level 24, x86 ABI, and Android 7.0 (Google Play) as the target. This device is compatible with Google Play, enabling testing with Google Play services. Additionally, two user-permission tags were added to the AndroidManifest.xml file for internet access and external storage. In the build.gradle.kts (Module: app) file, a dependency has been added to enable the use of the Glide library needed for handling images.

4.2. FIREBASE INTEGRATION

The Android Studio and Firebase were connected to the same Google account. The Android Studio project was connected to Firebase through Firebase Assistant in the Android Studio tools menu option. After creating a project in Firebase, the Android Studio project was connected to the created Firebase Android application. The Firebase Assistant offers tools and infrastructures from Google for developing, growing, and earning money from applications: Analytics, Authentication, Realtime Database, Cloud Firestore, Cloud Storage for Firebase, Cloud Functions for Firebase, Firebase ML, Crashlytics, Performance Monitoring, Test Lab, App Distribution, Cloud Messaging, In-App Messaging, Remote Config, and Admob. Through the Assistant, in the Android Studio project was added Firebase Authentication SDK, Cloud Firestore SDK, and Cloud Storage SDK. This can be seen in build.gradle.kts (Module :app).

Sign-in method “Email/Password” was enabled in the Firebase console, allowing users to sign up using their email address and password. Because all Firebase backend services are designed to scale seamlessly as the application’s user base grows and provide serverless backend infrastructure. By using Firebase Authentication, Firebase Cloud Firestore, and Firebase Cloud Storage, the scalability of the application as the number of registered users increases has been established.

In the Firebase console, the Firestore Database and Storage tabs have set rules. Apart from needing to change the Firebase security rules to allow only the authenticated users to read, write, and update, there were no difficulties encountered during the implementation phase.

Firebase Firestore and Storage synchronize data in real-time. The reason why is Firestore used for storing user's account information instead of Real-time Database is because of its extra feature that gives offline usage to users. Users can get access to all the data stored in Firestore even if they are offline which is not available in real-time database. Also, any updates made offline will be implemented when the connection is restored.

4.3. SIGNUPACTIVITY JAVA

Before the registration process begins, the user needs to enter all required EditText user interface elements based on their chosen checkbox and indicated guidelines. With the createUserWithEmailAndPassword() method provided by the Firebase Authentication SDK, every user will have a unique identifier (UID).

When the user is successfully created in Firebase Authentication, the document for the created user's UID is created inside the "Users" collection using the collection() method and the document() method provided by Firebase Firestore SDK. If the "Users" collection does not exist while creating the document, it will automatically be created by Firestore. The user's UID was obtained with the getCurrentUser() method provided by the Firebase Authentication SDK. After creating the document inside the collection, all entered data for the created account will be set to that document.

The registration process is completed when all the entered data for the created account is set to the created document. The user is then redirected to the home page and all previous actions made are cleared. On the other hand, if the registration process fails, users are notified, and their previous actions are not cleared.

Because the Administrator checkbox option does not exist, the admin was created as a regular user and, in Firebase Firestore. the checkbox field was manually changed. This ensures that the admin account can only be created by a person who manages the Firebase platform for the Zero Food Waste application.

With the checkbox logic, the application can create all required types of users and condition users to fill out every necessary field based on the type of account they want to create. Also, Firebase Authentication provides a secure registration process where user passwords are encrypted and cannot be viewed in the Firebase Console.

```
public void createUser() {
    fAuth.createUserWithEmailAndPassword(editTextEmailAddress.getText().toString(),
editTextPassword.getText().toString()).addOnSuccessListener(new OnSuccessListener<AuthResult>()
{
            @Override
            public void onSuccess(AuthResult authResult) {
                FirebaseUser user = fAuth.getCurrentUser();
                DocumentReference df = fStore.collection("Users").document(user.getUid());
                Map<String, Object> userInfo = new HashMap<>();
                //... put in userInfo everything entered in required EditText fields
                //... checkbox logic
                df.set(userInfo
                redirectionLogic();
            }
    }).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {//...}
    });
}
```

Listing 4.3. CreateUser() function written in Java.

4.4. LOGINACTIVITY JAVA

The application redirects only the already logged-in users wanting to access the login page to the home page. This is done by checking if the returned current sign-in user is not null with the getCurrentUser() method provided by the Firebase Authentication SDK.

After clicking the login button, the application checks if the email and password fields are not empty. If a field is empty, the user will see a red dot with an exclamation point that displays the message "Field is empty". Only when both fields are valid, the application will try to sign in with the signInWithEmailAndPassword() method provided by the Firebase Authentication SDK. If the user with entered credentials does not exist in Firebase Authentication, a toast message will be shown to the user with an adequate message. On the other hand, if the user is successfully logged in, the application will redirect the user, regardless of their account type, to the home page. On the login page, the user is offered an additional button for redirecting to the sign-up page to create an account.

4.5. PROFILEACTIVITY JAVA AND EDITPROFILEACTIVITY JAVA

Upon opening the profile page from the home page menu, the user's profile photo from Firebase Storage and the user's data from Firestore is loaded and set in appropriate fields. This is possible with methods provided by the Firebase Storage SDK and Firebase Firestore SDK. Methods like collection() and document() are used for accessing the user's document inside the "Users" collection in Firestore. On the other hand, with the getInstance(), getReferance(), and child() method, the image saved in Firebase Storage is found by constructing the path to the image using the user's UID under the "user_images" directory. With the getDownloadUrl() method and the Glide library, the successfully retrieved download URL is loaded into the ImageView UI component. If the retrial fails, for either image or the data, the user will be informed via toast message.

```
private void uploadImageFromFirebaseStorage() {
    FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
    if (user != null) {
        StorageReference imageRef = FirebaseStorage.getInstance().getReference()
                .child("user_images")
                .child(user.getUid() + ".jpg");

        imageRef.getDownloadUrl()
                .addOnSuccessListener(new OnSuccessListener<Uri>() {
                    @Override
                    public void onSuccess(Uri uri) {
                        Glide.with(ProfileActivity.this)
                                .load(uri)
                                .into(imageViewProfile);
                    }
                })
                .addOnFailureListener(new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception e) {//...}
                });
    }
}
```

Listing 4.5. U uploadImageFromFirebaseStorage() function written in Java.

The profile page has two options for changing the profile photo by choosing the image from the internal storage or with the random photo generator implemented functionality. Accessing the user's internal storage, the application handles the dangerous permission accordingly by requesting the needed permission from the user. The set image will be uploaded to Firebase Storage by constructing the path to the image using the user's UID under the "user_images" directory. With the putFile() method, the set image Uniform Resource Identifier (URI) will be saved.

The logged-in user also has the option to delete the account or logout. The deleting process includes showing the delete confirmation dialog, deleting all the account data saved in Firebase Firestore, deleting the profile image from Firebase Storage, and lastly deleting the user from Firebase Authentication. This is all possible with the delete() method provided by Firebase Firestore SDK, Firebase Storage SDK, and Firebase Authentication SDK. After deleting, the user is signed out with the signOut() method provided by Firebase Authentication SDK.

Activating the edit profile button opens the corresponding page, stores the displayed user data in Bundle extras, and transfers them through the profile activity's explicit Intent object. Saving the made changes in the edit profile page is done by finding what data is changed and updating only those fields in Firestore. This can be done with the same logic used for registration when the user's data is saved in the created UID.

5. DISCUSSION

The results of SWOT analyses highlight the Android Studio project's strengths, weaknesses, opportunities, and threats with using Firebase.

- Strengths: easy and fast connecting Android Studio project to Firebase and its services via SDKs. Project functionalities are efficiently created with pre-build methods provided by Firebase service SDKs. Those few lines of code can easily be replaced with another original solution. Firebase enables developers free time to focus on improving user interface and user experience and work on other functionalities, features, and innovations. The Firebase Console provides a possibility to restrict CRUD (Create, Read, Update, and Delete) user access with Firebase security rules. Firestore and Storage provide real-time updates. Also, Firestore provides offline access to stored data. The application can scale seamlessly with the help of Firebase.
- Weaknesses: Firebase has a pay-as-you-go model where they charge based on the application's usage of resources, exciding the set limit the platform is not free.
- Opportunities: the possibility of better positioning on the market because developers have free time for improvements and marketing without thinking about the backend infrastructure. Developers can develop quickly which is a significant competitive advantage. With packages offered by Firebase, the application can support a large number of users and data.
- Threats: Not protecting the user access level can cause problems such as data loss or endangering the user's private data.

6. CONCLUSION

The application was efficiently created for the set one-month deadline. Because of using Firebase, the application was improved with custom buttons and toast messages, and by implementing functionality for choosing a profile photo with a random photo generator made using https://picsum.photos. Handling dangerous permissions was improved as well as showing a confirmation dialog before deleting an account. The login page saves the entered data with Shared Preferences whereas the signup page uses Internal Storage in order for users to continue where they left off after returning to the application. This was a huge relief for testing login and registration functionalities.

Firebase is a great choice for small to medium-sized projects, prototypes, or applications that value quick and easy development. It is also well-suited for projects that require real-time updates and collaboration.

The research contributes by presenting a solution that streamlines backend infrastructure complexities and eliminates the need for a dedicated server infrastructure. This results in developers having more time to develop functionalities and work on UI/UX design. Furthermore, the straightforward Firebase services accessed through Firebase SDKs facilitate integration and enhance development agility, making it easily replaceable with an original solution or another backend service. Firebase services save developers time, but not at the expense of product quality and security.

Future directions of this research would be developing features for users to verify, recover, and change their email addresses and passwords. In addition to authentication, authorization, and profile management functionality, other functionalities foreseen by the "Zero Food Waste" project can be implemented. This includes sales management, donations management, and recipient needs management, as well as ordering and payment functionalities.

7. REFERENCES

[1] P. Chougale, V. Yadav and A. T. Gaikwad, "FIREBASE -OVERVIEW AND USAGE," in *Journal of Engineering and Technology Management*, 2022.

[2] I. K. G. Sudiartha, I. N. E. Indrayana, I. W. Suasnawa, S. A. Asri and P. W. Sunu, "Data Structure Comparison Between MySql Relational Database and Firebase Database NoSql on Mobile Based Tourist Tracking Application," in *Journal of Physics: Conference Series*, 2020.

[3] G. A. Dhanush, D. Rathi, G. Harini, T. P. V. K. Teja and P. Kumar, "Develop a Scalable and Serverless Client-based Application using Agile Methodology," in *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 2021.

[4] K. Marimuthu, A. Panneerselvam, S. Selvaraj, L. P. Venkatesan and V. Sivaganesan, "Android Based College App Using Flutter Dart," in *Green Intelligent Systems and Applications*, 2023.

[5] I. Hossain, S. M. A. Ullah and A. K. M. M. Haque, "Managing the Activities of a University Department through Android Application," in International *Journal of Engineering and Information Systems (IJEAIS)*, 2022.

[6] M. I. D. Mohd Harith, N. I. Ramli and S. Z. Mohd Muji, "Development of Food Giver Mobile Applications by Using Android Studio," in *Evolution in Electrical and Electronic Engineering*, 2021.

[7] S. Rahman, S. Sarkar and M. Biswas, "Project C -Contact tracing with Firebase," in *SSRN Electronic Journal*, 2021.

[8] R. Kesavan, D. Gay, D. Thevessen, J. Shah and C. Mohan, "Firestore: The NoSQL Serverless Database for the Application Developer," in *2023 IEEE 39th International Conference on Data Engineering (ICDE),* 2023.

[9] C. Khawas and P. Shah, "Application of Firebase in Android App Development-A Study," in *International Journal of Computer Applications*, 2018.

[10] D. Milojkovic, "The Role of Soft Projects for Managing Local Sustainable Development," in *4th ISPEC INTERNATIONAL CONGRESS ON CONTEMPORARY SCIENTIFIC RESEARCH, Ganja*, 2023.

[11] D. Milojković and K. Milojković, "Improving the Business Resilience of an Organization by Applying Agile Project Managemet Approach," in *FINIZ 2022 - Business Resilience in a Changing World, Belgrade*, 2022.

[12] D. Sharma and H. Dand, "Firebase as BaaS for College Android Application," in *International Journal of Computer Applications*, 2019.

[13] V. Padme, "ENHANCING ANDROID UI/UX WITH FIRESTORE," in *International Research Journal of Modernization in Engineering Technology and Science*, 2024.