Benefits of Firebase Authentication over Traditional Authentication

Ekta Yuvraj Waghulde[1] Ruchita Jaitapkar[2]
[1,2]ASM College, Thane, Maharashtra, India

Abstract— In today's world most apps must have some form of user identification. This allows them to set preferences, store data, and provide personalized experiences that are consistent across all the user's devices. In order to provide this, they must provide the facility to sign up new users, sign in existing users, manage account details, and keep all this data secure. It's a very difficult and time-consuming process. From user experience perspective, it is also very difficult to get right information. Users tend to hesitate to give out information that is confidential such as their user name, password, security questions, and anything else that should they don't to be leaked or cause any sort of damage to them. As a result, they generally prefer using credentials that they have already provided to a third party, and having that third party manage the sign-in for them ,so for example, if they have a Facebook account, they use Facebook to verify to your app that they are who they say they are, without needing to give you their information also.

Keywords: Traditional Authentication Vs Firebase Authentication, Email Link Authentication

I. INTRODUCTION

All applications need to know the identity of its user. After getting all the information about the user's identity allows the application to securely save user data in the cloud and provide the same personalized experience across all of the user's devices. It supports authentication using phone numbers, passwords, popularly federated identity providers like Google, Facebook and Twitter, and more.

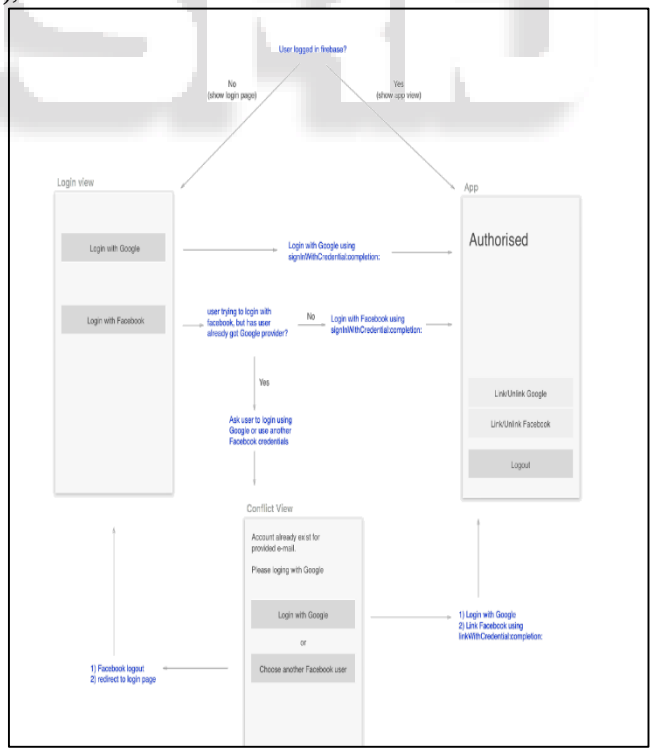
Authentication is the main module in any modern application because without authentication we would not able to identify the user, and if the system is incapable of that then the application will not sustain in the long run. Traditionally for authentication user must fill a form which consists of his information, and to become a user at multiple places or to get registered with multiple application the user must repeatedly fill the form. This leads to bad user experience and many a most of the time user leaves the application. Apart from the user side for authentication, we must deploy complex code with multiple API's calls on our servers which leads degradation of server performance. So, to overcome this lengthy procedure of authentication one must use Firebase Authentication, which provides authentication service along with UI. It also enables authentication using Facebook, Google, Twitter, GitHub credentials.

II. TRADITIONAL AUTHENTICATION VS FIREBASE AUTHENTICATION

In traditional authentication user normally sign in with user id and password which user can have multiple user id's by signing up number of times which can create 'n' number of users data to be stored and there will be no tracking of data or identification of user where in firebase authentication services provides you with authentication service with UI. It also enables authentication sign in Facebook, google, twitter credentials which makes provider easier to identify user and store data accordingly and user can easily backup the data. Whenever users try to login again with same credentials can restore the data and come up to same page where user last edited. When someone tries to sign in with google account and user is validated with validation link on our google account is an example of firebase authentication.

After adding firebase and authentication dependency through Facebook, the user can create login id by the following code:

```
FirebaseAuth auth=FirebaseAuth.getInstance(); auth. signInWithEmailAndPassword (email, password)
 . addOnCompleteListener (new OnCompleteListener ())
 {
    @Override
public void onComplete (Task task)
 {
      if(task.isSuccessful())
      {
                    FirebaseUser user=task. getResult(). getUser();
String email=user. getEmail();
  //… }
      }
   }
);
```



III. TYPES OF AUTHENTICATION

A. Firebase Authentication for Facebook

Examples most of apps use google or Facebook authentication to validate the user by sending validation link or a pop up of allowing user to access the application.

For example, Ludo king application, if a person logs in as a guest in the game and plays two levels and logs out, after he logs in again he won't be able to play from the third level he has to play from the first level again. But if the person logs in through Facebook and plays two levels and logs out, he can continue playing from the third level when he logs in again.

B. Email link Authentication

One can use Firebase Authentication to sign in a user by sending them an email having a link, which they can click to sign in. In this process, the user's email address is also verified.

There are numerous benefits to signing in by email:

- Low friction sign-up and sign-in.
- Lowers the risk of password reuse through different applications, which can undermine security of even well-selected passwords.
- The ability to authenticate a user while also verifying that the user is the legitimate owner of an email address.
- A user must only have an accessible email account to sign in. There is no requirement of phone number or social media account.
- The existing user can easily sign in into the account. For example, if a person forgets his password he will be still able to login into the account with resetting his password.

Apart from the user side for authentication, we must deploy complex code with multiple API's calls on our servers which leads degradation of server performance. So, to overcome this lengthy procedure of authentication one must use Firebase Authentication.

C. Firebase Authentication with the Help of Phone

For example, OTP, call answering, (providing accept or reject request) You can use Firebase Authentication to enable a user to sign in to your app by sending an SMS to user's device which contains a one-time-password. The user then enters this OTP in your app, if the OTP matches then sign in is successful and the user can then access your app.

D. Firebase Authentication for Rave

1) Challenges:

Rave is available on android, iOS and now is being developed for VR. It required a particular platform agnostic login system which could be able to handle authentication across many platforms (VR,iOS,Web,Android), which would be secure and easy to use. Rave didn't want to worry about the security implications related to securing user tokens or expiring of tokens.

2) Solution:

Firebase Authentication has proven to be transparent and very easy implement and use. It works across multiple platforms seamlessly, thus it allows easy integration across its VR platform; iOS app and Android approve found the process of both server-side and client-side implementation to be painless. One of the biggest benefits of using Firebase Authentication is security, database security, credential storage and transmission. Rave takes the security of its customers seriously and is very confident in the services provided by Firebase Authentication. It was possible for Rave to implement Firebase Authentication with one or two days on client and in one hour on the server. This allowed Rave to set up and run a multi-platform login system.

IV. CONCLUSION

This paper highlights on the study about the firebase authentication and its uses. This paper also provides information about how to make your data secure, easy integration of data. By using Firebase Cloud Functions make your application serverless. By studying this paper, we also come to know that we can secure our accounts information, passwords safely.

ACKNOWLEDGEMENT

I thank Mrs. Reeta Singh for her assistance by providing proper formats and all faculties for guiding. I would also like to show our gratitude to the librarian staff Mr. Milind Dubal for sharing references during this research.

REFERENCES

[1] https://firebase.google.com/
[2] https://firebase.google.com/docs/auth
[3] Firebase Products:https://firebase.google.com/ products/
[4] https://hackernoon.com/introduction-to-firebase-218a23186cd7
[5] Navdeep Singh, Study of Google Firebase API for Android, IJIRCCE (An ISO 3297: 2007 Certified Organization) Vol. 4, Issue 9, September 2016
[6] Firebase Release and Services: https://en. wikipedia.org/ wiki/Firebase

