WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Application of Firebase in Windows and Android Applications

[1]Mohini Gawande, [2,*]Sarveshkumar Nasare, [3]Saloni Meshram, [4]Sahil Bhoyar, [5]Sanket Bachchaw

[1]Assistant Professor, [2,3,4,5]UG Students
[1,2,3,4,5]Department of Computer Science & Engineering
[1,2,3,4,5]Govindrao Wanjari College of Engineering and Technology Nagpur, India

ABSTRACT

The web applications as well as websites are more and more relied upon the large amount of database and unorganized data such as videos, images, audio, text, files and other arbitrary types. It is hard for Relational Database Management System (RDBMS) to handle the non-structured data. Firebase is a relatively newer technology for handling large amount of unstructured data in comparison with RDBMS. It is very fast than RDBMS, since it uses JavaScript parser. This paper focuses on the application of Firebase with Android and Windows platforms and aims to understand the related concepts, advantages and respective limitations. The paper features the work to demonstrate some of the features of Firebase by developing a work-system of Android app and Windows app connected using same Firebase instance.

Keywords: Android, Windows, Python, auto-py-to-exe, Firebase, firebase_admin Python SDK

1. INTRODUCTION

The server used for Android apps are Oracle SQL, Microsoft SQL Server and MySQL which are connected to the server using PHP files. The Firebase ecosystem stores data in JSON format. The other servers use a table format which stores data in the form of rows and columns.

The Firebase is NoSQL based. There are very few cloud based services available which are similar to firebase, such as:

AWS Mobile Hub- It is integrated console that helps to test, build, create and monitor the mobile apps that leverages AWS services.

CloudKit- It is an Apple framework which allows to save data and store assets but similar to IOS only.

Parse Server- It was released by Facebook(now named as **META**) to replicate functionality of **Parse** which is an open source server. This is no longer in existence as Facebook shut this project down.

1.1 Firebase

Firebase is Google's application development platform. It helps developers to build high-quality apps. It stores the data in the form of "JavaScript Object Notation" (**JSON**) which doesn't use query for inserting, updating, deleting or adding operation performed on it. It is the backend of a system that works as a database for storing data.

The services available are:

1.1.1 Firebase Analytics(Google Analytics)

It provides insight into app usage. It is an app measurement solution that also provides free, unlimited reporting on up to 500 distinct events user engagement. This feature enables the developer of the application to understand how users are using the application. The SDK has the feature of capturing events and properties on its own and also allows getting custom data.

1.1.2 Firebase Cloud Messaging (FCM)

Firebase Cloud Messaging (FCM) provides a reliable and battery-efficient connection between the server and devices that allows delivering and receiving messages and notifications on iOS, Android, and the web at no cost.

1.1.3 Firebase Auth

Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made libraries for user-interfaces to authenticate users to the app. It supports authentication using phone numbers, passwords, popular federated identity providers like Google, Facebook and Twitter, and more.

1.1.4 Real-time Database

Firebase provides services like backend and real-time database. An API is provided to the application developer which allows synchronization of application data across clients and stored on Firebase's cloud. The client libraries are provided by the company which enables integration with Android, IOS, and other types of applications.

1.1.5 Firebase Firestore

Likewise Firebase Real-time Database, it keeps the data in sync across client apps through realtime listeners and offers offline support for mobile and web so the responsive apps can be built that work regardless of network latency or Internet connectivity.

1.1.6 Firebase Storage

It facilitates easy and secure file transfer regardless of network quality for the Firebase apps. It is backed by Google Cloud Storage which is cost-effective object storage service. The developer can use it to store images, audio, video, or other user-generated content.

1.1.7 Firebase Test Lab for Android

It provides cloud-based infrastructure for testing Android apps. With one operation, developers can initiate testing of their apps along a wide variety of devices and their configurations. The various test results like videos, screenshots and logs are available in the Firebase console. Even if developers haven't written any test code for their app, the Test Lab can exercise the app automatically, looking for crashes.

1.1.8 Firebase Crash Reporting(Firebase Crashlytics)

Real-time crash reporting in Firebase Crashlytics allows quickly troubleshooting and triaging any bugs in the app by gathering and grouping crashes based on where they occurred in the application's code.

1.1.9 Firebase Notifications

It is a freely available service which enables targeted user notifications for mobile app developers.

1.2 Android App

Android is a mobile operating system[12],[13] which is modified version of the Linux kernel and other open source software, designed primarily for touch screen mobile devices such as Tablets and Smartphone. Applications ("apps"), which extend the functionality of devices, are written using the Android software development kit (SDK) and, often, XML Java/Kotlin programming language. The SDK includes a comprehensive set of development tools, including a debugger, software libraries, a handset emulator based on QEMU(Quick EMUlator), sample code, documentation and tutorials. Initially, Eclipse was Google's supported integrated development environment (IDE) using the Android Development Tools (ADT) plugin; later on in December 2014, Google released Android Studio, based on IntelliJ IDEA, as its primary IDE for Android application development. Other development tools are also available, including a native development kit (NDK) for the applications.

Using the Android studio and the above listed tools, an Android app is made. The app will also be called as "Third-party app".

1.3 Windows App Using Python

Windows apps are made in a variety of languages; each language used is selected if it suits the requirements for the end-product. For instance, if your app is time critical, like a game, you should be developed in C or C++, or if something that involves AI then the app should use Python with its hundreds of libraries that allows doing anything with it. Overall, what language to be used is really up to the requirements, as long as it has a compiler that supports having Windows targets.

1.3.1 Python Programming language

Python is a high-level, interpreted, general-purpose programming language [14]. Its design philosophy emphasizes code readability with the use of significant indentations wherever necessary. Python is dynamically-typed as well as garbage-collected. It supports different programming paradigms, including structured (particularly procedural), object-oriented and functional programming.

There are numerous modules made in Python language. In this work, we have used following modules:

tkinter: Tkinter is the standard GUI library for Python meaning that it comes built in as the Python is installed. Python when combined with Tkinter provides an easy and fast way to create GUI applications. The Tkinter library provides a powerful object-oriented interface to the Tk GUI toolkit. Creating a GUI application using Tkinter is an easy task.

auto-py-to-exe: A module used to convert a **.py** file into **.exe** file[15] using a simple graphical user interface and **PyInstaller** module is used in background.

firebase-admin: The *Firebase Admin Python* SDK enables server-side (backend) *Python* developers to integrate Firebase into their services and applications[16], the detailed working of this module is discussed in **section 5**.

2. LITERATURE REVIEW

Walter Kriha [1] in his article has mention the overview of NoSQL database and common concepts, techniques and patterns as well as several classes of NoSQL databases (key/value-stores, document databases, columnoriented databases) and individual products. Various advantages and limitations of using NoSQL database has been discussed. Supriya S. Pore, Swalaya B. Pawari [2] conducted a comparative study between SQL and NoSQL. The study revolves around databases like SQL and NoSQL, it also differentiates among both. The Axiomatics of SQL and NoSQL databases has been studied and described in this paper. The study concludes that due to data consistency, ACID property is not used in the NoSQL databases Vatika Sharma, Meenu Dave [3] have given an overview of NoSQL databases focusing on how it has lowered the dominance of SQL with its background and characteristics. Daniel Pan [4] in his article has shown about connecting firebase with an Android app and basics of designing the structure of database in Firebase. Landon Cox [5] highlights the comparative study between SQLite and firebase. It also focuses on organizing data as a JSON tree in order to store in Firebase. The documentation in

3. STEPS TO ADD FIREBASE TO ANDROID

Firebase can be added to any project with minimum of Android 2.3 (Gingerbread) and at least have Google Play services 9.6.1. The steps below are to be followed up next:

A project needs to be created in firebase console. Enter project name and the location. Project name under firebase console may be anything and also may different from the application name given.

Click on “Add Firebase to the Android app” and follow steps mentioned along there.

The user will be prompted for the package name and optional SHA-1 certificate whose data is available at android studio.

At the end “google-services.json” file will be downloaded after the successful completion of above steps. The file can be downloaded again at anytime from the Firebase console. The downloaded file has to be copied into the project’s app or module folder. Firebase is added to the project.

4. USING FIREBASE FEATURES IN ANDROID APPLICATION

Using the features of Firebase in the Android application is very easy and requires a few lines of code. The features like authentication and real-time database have been discussed in section. The furthermore details about the features are available on Google Firebase guide link listed in the reference section. The methodology to use some features are as:

4.1 Authentication:

After adding firebase and authentication dependency to the Android application, the login id of the user can be created by the following Java code [6]:

```
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();

         //...

      }

   }

});
```

User login can also be used from Google, Facebook, Twitter and GitHub.

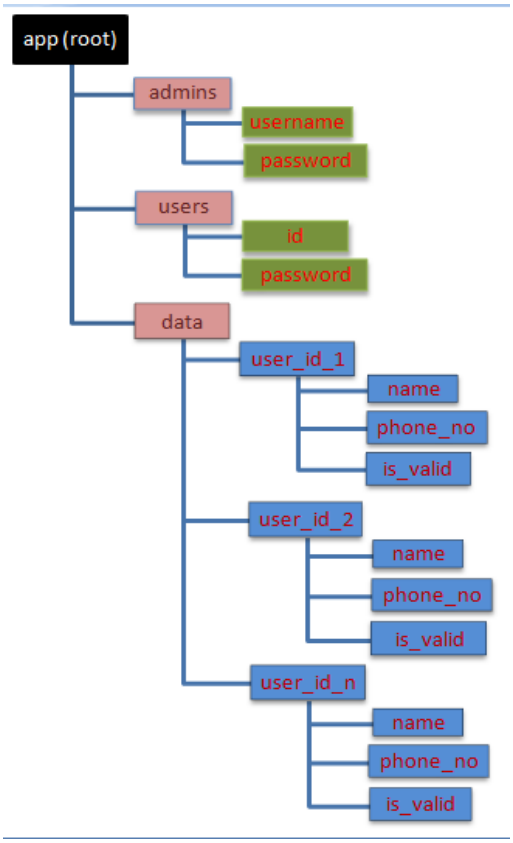


Figure 1: Firebase database structure

4.2 Database

Real-time database in Firebase is very easy to use. Once the respective dependencies of Firebase and database are added to the app, non-structured data can be added to database by the following Java code [7]:

```
//Write a message to the database

FirebaseDatabase database = FirebaseDatabase.getInstance(/*database URL */);

DatabaseReference myref = database.getReference("node");

myRef.setValue ("Hello, World");
```

In Figure 1, the structure of data stored in Firebase is shown where we have the **root** of the app and there are three children of the **root** that is **admins**, **users** & **data**. Again, the **admins**, **users** & **data** have their child nodes as shown in the figure. The **admins** and **users** consist of data about Admins (Windows app users) and Users (Android app users) of the work-system respectively. The **data** node consists of user id of users following the information (name and phone number) uploaded by the respective users through the android app, the Boolean **is_valid** specifies whether the data is checked and validated by the admins through Windows app or not.

4.3 Inserting New Data

An instance of object class or a map can be used to insert new data. Once the object is created, navigate the Firebase reference to the position where a child should be added. If a created list does not have a specific names for each child, the **push**() method can be used before the **setValue**() is called:

```
ref.push().setValue(/*object*/)

//or

 ref.setValue(/*object*/).
```

4.4 Updating Data

Navigate the Firebase reference to the parent of the item that one wants to update and then a map containing the update values can be created.

```
ref.updateChildren(/*map*/);
```

4.5 Removing Data

Navigate the Firebase reference to the item that one is to be removed

```
ref.removeValue(/*object*/);
```

5. STEPS TO ADD FIREBASE TO WINDOWS APPLICATION USING PYTHON

In the **section 3**, we set up the Firebase for the Android application. Now, we set up the same for Windows application. The windows app is made using **Python** programming language, the Firebase setup on Python is done using **firebase-admin** module which is installed by using the following command on the terminal:

```
pip install firebase-admin
```

To connect to Firebase, the following lines of **Python** code are also needed:

```
import firebase_admin

path = #path to private key which is in JSON format
databaseURL = #URL of database in string format
cred_obj = firebase_admin.credentials.Certificate(path)
default_app = firebase_admin.initialize_app(cred_obj, {
          'databaseURL':databaseURL
          })
```

The path to the private key must be provided to create the credentials object. To generate the key, go to ‘project settings’, click on ‘Generate new private key’, download the file, and place it in your directory structure. Furthermore, the **databaseURL** is also required, which is simply the URL that gives access to the created database. It is present on the Firebase Console page of realtime database. Firebase is now also set up with **Python**.

6. USING FIREBASE FEATURES IN WINDOWS APPLICATION

The work presented in this paper, deals with simple and basic **CRUD** (Create-Read-Update-Delete) operations done on the database and user authentication.

6.1 CRUD Operations

To perform CRUD operations, the following code [17] is used to set the reference variable to the **root** of the database:

```
from firebase_admin import db

ref = db.reference("/")
```

The following functions are used to perform operations on the database:

- **set() function :** This function is used to set the key-value pairs in the JSON format on the specific reference of the database. For this, **set()** variable should be called using reference variable and the key-value pair should be passed to it, as shown below:

```
import json

keyval = json.load("key:value")
#converted into JSON format

ref.set(keyval)#set() function
```

To delete the data from specific reference, the **set()** is called using reference variable and empty set is passed as parameter as shown below:

```
ref.set({})#deletes data from reference
variable
```

- **push() function :** This function is used to save data under a unique system generated key. The function also ensures that if multiple writes are being performed to the same key, they do not overwrite themselves. The function is called as:

```
ref.push().set(value)# value is passed
with a unique system generated key
```

- **get() function :** this function is used to retrieve data from the database. For this, the function is called on the reference variable as:

```
data = ref.get() #returns data from the reference in
the database
```

- **update() function :** This function is used to updating the database. To do this, **get()** function is preferred to use firstly following the update operation as shown below:

```
ref = db.reference("/app/admins/")
data_root = ref.get()

#refer figure-1 of the section '4.1.2'
for key, value in data_root.items():
   if(value["admins"] == "Sarveshkumar")&&
      value["phone_number"] = 123456789 :#demo
                                                #number
   ref.child(key).update({"phone_number":987654321})
```

6.2 Authentication

After setting firebase, the user can create login id using email & password by the following **Python** function [16]:

```
from firebase_admin import auth

def create_user(email, password):
      m_auth = auth.create_user(email, password)
      return m_auth
```

The following functions are also available in **firebase_admin.auth** module, which are generally used for basic operations on **Firebase authentication**:

- **delete_user(*uid*) :** Deletes the user identified by the specified **uid** as user ID.
- **delete_users(*uids*) :** Deletes the users specified by the given identifiers.
- **generate_email_verification_link(*email*, *action_code_settings=None*) :** Generates the out-of band email action link for email verification flows for the specified email address.
- **generate_password_reset_link(*email*, *action_code_settings=None*) :** Generates the out-of band email action link for password reset flows for the specified email address.
- **generate_sign_in_with_email_link(*email*, *action_code_settings*) :** Generates the out-of-band email action link for email link sign-in flows, using the action code settings provided.
- **get_user(*uid*) :** Gets the user data corresponding to the specified user ID.
- **get_user_by_email(*email*) :** Gets the user data corresponding to the specified user email.
- **get_user_by_phone_number(*phone_number*) :** Gets the user data corresponding to the specified phone number.
- **get_users(*identifiers*):** Gets the user data corresponding to the specified identifiers.
- **update_user(*uid*, ***kwargs*) :** Updates an existing user account with the specified properties.

These are generally used functions, yet there are a lot of other functions also defined in the module.

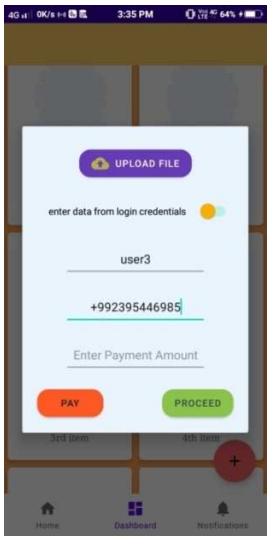
7. WORKFLOW OF THE SYSTEM

The system comprises of three main modules:

- **Android app:** The app is made for **users** and it requires **sign-up/sign-in** for users which demonstrates the Firebase authentication, about which **section-4.1** of the paper is covered with implementation. To demonstrate the **CRUD** operations on database, user will enter its **name** and **phone_no** which will be stored in **data** node of the **user** node (refer **figure-1**).
- **Firebase:** The Firebase works as a database which stores the data on the cloud storage.
  The structure of database is discussed in **section-4.2** and **figure-1** of the paper. The Firebase connects the **Android app** and **Windows app**.
- **Windows app:** The windows app is made for **admins** and it also requires **sign-in/sign-up** for admins, about which **section-6.2** is covered with implementation. The person signed in as **admin** manually checks and validates the **data** of the **user**, which demonstrates the accessibility of same information in the Firebase using both apps.

8. RESULT

The Android app presented in this work is tested on a mobile device (RAM:2GB/ROM:16GB) powered by **Android 8.1.0** and created in **Android Studio bumblebee (2021.1.1)** using **Java** & **XML**. The Windows app presented is tested on computer powered by **Windows-10 Pro (64-bit)** with processor of **3.1 GHz Intel core i3-2100 processor** with **4GB DDR3 RAM** and **500 GB HDD,** app is created in **VS-Code (1.67.2)** using **Python 3.10(64-bit)**.



8.1 Android App

The screenshot of Android app has been included in **figure-2**. The app contains several different fragments bound to the single activity whereas **figure-2** shows relevant **User Interface** responsible to deal with **Firebase** operations. Here, the **EditText** with text-hint **'Enter Your Name '** and **'Enter Phone Number'** accepts the specified input which is later on sent to be stored on Firebase by a click

on **PROCEED** button. The **View** instances shown in the **figure-2** other than these three (2-EditTexts and 1-Button) are not related to the work presented in this paper.

8.2 Firebase Database

The data received from android app is stored in database, along with the unique user-id of the user logged in on the android app. The windows app is also able to access the **Firebase**; hence it works as a bridge between **Android** app and **Windows** app.

8.3 Windows App

Figure 2: Android app

The screenshot of windows app has been included in **figure-3**. The app contains several different functionalities whereas **figure-3** shows relevant **User Interface** responsible to deal with **Firebase** operations. Here, the listbox under the **SCHOLARSHIP DOCUMENTS** section displays the data from **data** node of our Firebase database. After clicking on a list-item, admin signed in on Windows app can validate the data which was uploaded by the Android app user to the Firebase, which will in-turn toggle the value of respective **is_valid**(Boolean data) for independent data as shown in **figure-1** to denote that the data is validated by the admin. The **widgets** shown in the **figure-3** other than those which are in listbox, are not related to the work presented in this paper.

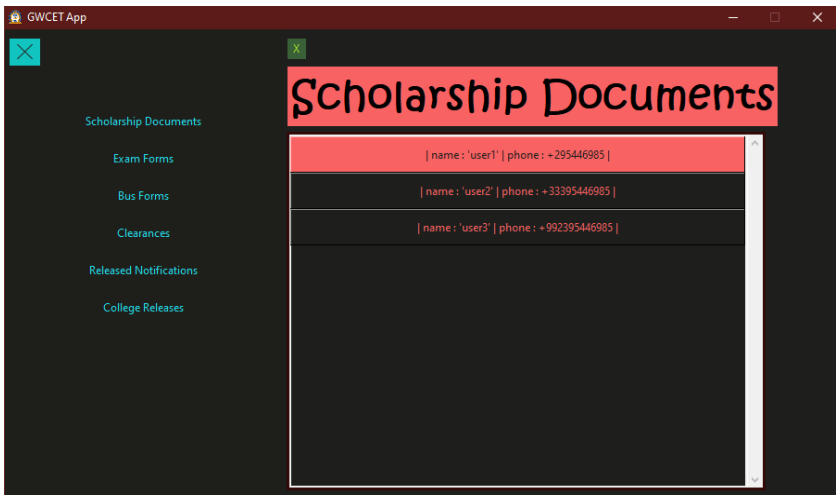


Figure 3: Windows App

9. CONCLUSION AND FUTURE SCOPE

This paper highlights on the study about Google provided Firebase API and features. This paper helps in studying how to use Firebase in the Android application as well as Windows application according to the developer requirement. This also helps in making both kinds of apps faster and efficient as no PHP is required as a third party language to communicate with the database. It provides a secure channel to communicate with the database directly from JAVA for Android application and Python for Windows application. The study material is based on the data provided online and referring to the examples given. Google has been updating Firebase on regular basis, It can not only be used in Android but also to connect cross platform. The work can be further extended by adding new features and exploring new possibilities in Android and Windows applications.

10. ACKNOWLEDGEMENT

It is our proud duty and privilege to acknowledge the kind of help and guidance received from several people in preparation of this report. It would not have been possible to prepare this project in this form without their valuable help, co-operation and guidance. First and foremost, we thank our project guide and co-ordinator **Mrs. Mohini Gawande** Asst. Professor Department of Computer Science and Engineering, Govindrao Wanjari College of Engineering and Technology for their valuable guidance and all the encouragement that lead towards completion of our project. We would like to thank **Mr. Vivekanand Thakare,** HOD, Department of Computer Science and Engineering, Govindrao Wanjari College of Engineering and Technology for his valuable suggestions and guidance throughout the period of this project. We also wish to record our sincere gratitude of **Dr. Salim Chavan**, Principal, Govindrao Wanjari College of Engineering and Technology for his constant support and encouragement in preparation of this report and for providing Library and laboratory facilities needed to prepare our project report.

Last but not least, we would like to thank our advisor, friends, parents, teaching and non-teaching staff of GWCET.

11. REFERENCES

[1] Kriha Walter, 2009 NoSQL Databases Hochschule der Medien. Stuttgart Media University. Stuttgart.

[2] Pore Supriya S, Pawar Swalaya B, 2015. Comparative Study of SQL & NoSQL Databases. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET). Volume 4 Issue 5, May 2015

[3] Sharma Vatika, Dave Meenu. 2012. SQL and NoSQL Databases. International Journal of Advanced Research in Computer Science and Software Engineering. Volume 2, Issue 8, August 2012.

[4] Daniel Pan. 2016. Firebase Tutorial. October, 2016.

[5] Cox Landon. 2017. SQLite in Android. March 2017.

[6] Kalsov, 2012. Developer Meet Firebase dated 18/3/17.

[7] "Firebase Realtime Database". Firebase, Inc dated 18/3/17.

[8] Bill Stonehem, Google Android Firebase: Learning the Basics Paperback, 2016 dated 18/3/17.

[9] Isuru Madusanka, Busy programmer‟s guide to Firebase with Android, 2013.

[10] G. Harisson, “10 things you should know about NoSQL databases” dated 18/3/17.

[11] https://www.techrepublic.com/article/10-things-you-should-know-about-nosql-databases/

[12] https://www.android.com

[13] https://developer.android.com/

[14] https://www.python.org/

[15] https://pypi.org/project/auto-py-to-exe/

[16] https://firebase.google.com/docs/reference/admin/python

[17] https://www.freecodecamp.org/news/how-to-get-started-with-firebase-using-python/