

# Mobile Application Development

---

Produced  
by

David Drohan ([ddrohan@wit.ie](mailto:ddrohan@wit.ie))

Department of Computing & Mathematics  
Waterford Institute of Technology

<http://www.wit.ie>



Waterford Institute of Technology  
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE





# Android & Firebase Part 4

---

## Firebase Integration





# Agenda

---

- ❑ Firebase history
- ❑ The all new Firebase
- ❑ Real-time database
- ❑ Authentication
- ❑ Storage
- ❑ Remote config
- ❑ Hosting
- ❑ Crash reporting
- ❑ Test lab
- ❑ Firebase cloud messaging
- ❑ Dynamic links
- ❑ App indexing
- ❑ Analytics
- ❑ CoffeeMate Highlights & Demos along the way...



# Agenda

---

- Firebase history
- The all new Firebase
- Real-time database**
- Authentication
- Storage
- Remote config
- Hosting
- Crash reporting
- Test lab
- Firebase cloud messaging
- Dynamic links
- App indexing
- Analytics
- CoffeeMate Highlights & Demos along the way...



Database

Goodbye RDBMS...



# Real Time Database

---

- ❑ Unlike RDBMS, data is stored as JSON. It is no-SQL JSON database.
- ❑ What makes it real time it its ability to notify listeners of any change in data.
- ❑ Whenever any change is made in the JSON database structure, the firebase SDK notifies the app.
- ❑ So you can forget about REST API calls, connectivity checks, 3<sup>rd</sup> party libraries and polling.



# The Core Magic Of Firebase \*

---





# Saving Data In Firebase

---

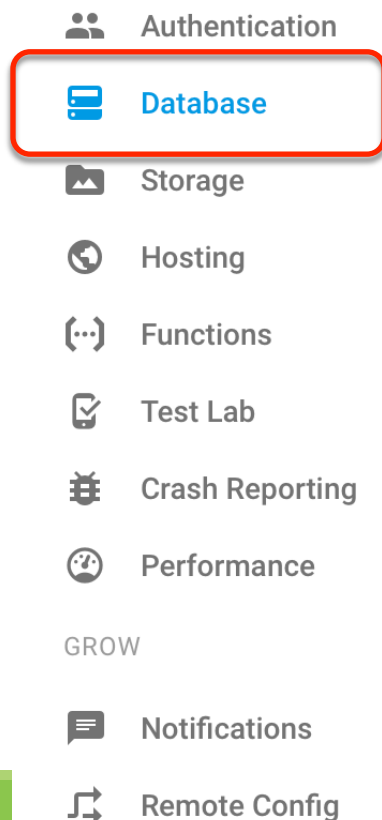
- ❑ Get database reference (your base node)
    - `DatabaseReference mDBRef = FirebaseDatabase.getInstance().getReference();`
  
  - ❑ Point it to the right JSON node
    - `mDBRef = mDBRef.child("mydb").child("table1");`
  
  - ❑ Set the value at the pointed node
    - `mDBRef.child('row1').setValue(myPOJO);`
  
  - ❑ Or push the value to create a unique key and set the value
    - `mDBRef.push().setValue(myPOJO);`
- Create a Key
- Firestore takes care of the Key





# View Your Saved Data

- ❑ Log on to <https://firebase.google.com>.
- ❑ Go to console and select your project.
- ❑ Hit database and select data tab





# Retrieving Data

---

- ❑ Data is retrieved via callbacks listeners.
- ❑ There are mainly 2 types of listeners
  - [ValueEventListener](#) – get entire child structure.
  - [ChildEventListener](#) – get only the child that got changed / added or deleted.
- ❑ Attach these listeners to [Database reference](#) we saw earlier.
- ❑ Both of these listeners are called once, so app can get the data and prepare UI



# Attaching Data Listeners

---

- ❑ ValueEventListeners can be added in 2 ways:
  - **addListenerForSingleValueEvent** (valEvListener); //next slide
    - ◆ *Get the entire data snapshot only once*
  - **addValueEventListener**(valEvListener);
    - ◆ *Get entire data snapshot whenever there is a change in any of the child nodes*
- ❑ ChildEventListener can be added in only one way:
  - **addChildEventListener**(childEvListener)
    - ◆ *Get updates as child nodes*



# Retrieving Data As POJO

---

```
public ChildEventListener childEvListener = new ChildEventListener() {  
  
    public void onChildAdded(DataSnapshot dataSnapshot, String s) {  
        MyPOJO m = dataSnapshot.getValue(MyPOJO.class);  
    }  
  
    public void onChildChanged(DataSnapshot dataSnapshot, String s) {}  
  
    public void onChildRemoved(DataSnapshot dataSnapshot) {}  
  
    public void onChildMoved(DataSnapshot dataSnapshot, String s) {}  
  
    public void onCancelled(DatabaseError databaseError) {}  
};
```



# Retrieving Data As POJO

---

```
public ValueEventListener valEvListener = new ValueEventListener() {  
    @Override  
    public void onDataChange(DataSnapshot dataSnapshot) {  
        MyPOJO m = dataSnapshot.getValue(MyPOJO.class);  
    }  
  
    @Override  
    public void onCancelled(DatabaseError databaseError) {}  
};
```



# Filtering Data

---

- ❑ Set [DatabaseReference](#) to the right parent node.
  - `Db = base.child('Institute').child('year');`
- ❑ To get all students with major = 'Computing'.
  - Set `orderByChild('major')`.
  - Set `equalTo('Computing')`;
  - Add listeners

```
Db.orderByChild('major').equalTo('Computing')  
.addChildEventListener(childEventListener);
```



# Pagination

---

- ❑ Create a [Query](#) object or keep using [DatabaseReference](#).  
Query is super class of DatabaseReference class.
- ❑ Set `orderBy` some child key (and filter if needed)
- ❑ Set limits (startAt, limit etc.)
- ❑ Attach data listeners
- ❑ And done. Appropriate callback will be called when operation is complete.



# Pagination Code

---

```
// class level
final int limit = 50;
int start = 0;

// event level
Query userListQuery = userDBRef.orderByChild("email")
.limitToFirst(limit).startAt(start);
userListQuery.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot d) {
        // Do something
        start += (limit+1);
    }
});
```





---

# Firebase Realtime Database



Quick DEMO...



---

# CoffeeMateFBI 1.0

## Setup & Code Highlights



# 1. Create a Firebase Project \*

The screenshot shows the Firebase console interface. At the top, there's a blue header with the Firebase logo and the text "Go to docs". Below the header, a large graphic of a smartphone is shown with a small person standing next to it. The main content area has a heading "Welcome back to Firebase" and a sub-heading "Continue building your apps with Firebase using some of the resources below." Below this, there are links for "Documentation", "Sample code", "API reference", and "Support". In the lower right section, there are two buttons: "CREATE NEW PROJECT" (highlighted with a red box) and "IMPORT GOOGLE PROJECT". On the left, under the heading "Your projects using Firebase", there is a card for a project named "PaceMaker" with the domain "pacemaker-a88f1.firebaseio.com".



# 1. Create a Firebase Project

The screenshot shows the Firebase console interface. A modal dialog titled "Create a project" is open in the center. The dialog contains the following elements:

- Project name:** A text input field containing "CoffeeMate FBI".
- Country/region:** A dropdown menu with "Ireland" selected.
- Disclaimer:** A paragraph of text stating: "By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is shared in your settings at any time. [Learn more.](#)"
- Buttons:** Two buttons at the bottom: "CANCEL" and "CREATE PROJECT".

The background of the console shows a "Welcome back to Firebase" message, a "Documentation" link, a "Sample code" link, and a section titled "Your projects using Firebase" which includes a card for "PaceMaker" with the URL "pacemaker-a88f1.firebaseio.com".



# 1. Create a Firebase Project \*

The screenshot shows the Firebase console interface. At the top, there's a blue header with the Firebase logo and the text "Go to docs". Below the header, a large illustration of a smartphone is shown with a small person standing next to it. The main content area is titled "Welcome back to Firebase" and includes a sub-header "Continue building your apps with Firebase using some of the resources below." followed by links for "Documentation", "Sample code", "API reference", and "Support".

Below this, there's a section titled "Your projects using Firebase" with two buttons: "CREATE NEW PROJECT" and "IMPORT GOOGLE PROJECT". Underneath, two project cards are visible:

- CoffeeMate FBI**: A card with a red border, showing the domain `coffeemate-fbi.firebaseio.com`.
- PaceMaker**: A card showing the domain `pacemaker-a88f1.firebaseio.com`.



# 1. Create a Firebase Project - Overview

The screenshot shows the Firebase console interface. The browser address bar displays `console.firebase.google.com/project/coffeemate-fbi/overview`. The page title is "CoffeeMate FBI" and the current view is "Overview".

The left sidebar contains a navigation menu with the following items:

- Overview (selected)
- Analytics
- DEVELOP
  - Authentication
  - Database
  - Storage
  - Hosting
  - Test Lab
  - Crash Reporting
- GROW
  - Notifications
  - Remote Config
- Spark
  - Free \$0/month
  - [UPGRADE](#)

The main content area displays a welcome message: "Welcome to Firebase! Get started here." Below this message are three prominent buttons for adding Firebase to different types of applications:

- iOS**: Add Firebase to your iOS app
- Android**: Add Firebase to your Android app
- Web**: Add Firebase to your web app

At the bottom of the page, there is a "Discover Firebase" section with two illustrative images: one showing a person working on a laptop with an upward-trending arrow, and another showing a person with a lanyard and ID badge.



# 1. Create a Firebase Project - Overview

The screenshot shows the Firebase console interface. At the top, the browser address bar displays 'console.firebase.google.com/project/coffeemate-fbi/overview'. The page header includes the Firebase logo, the project name 'CoffeeMate FBI', and the 'Overview' tab. A navigation sidebar on the left lists various services: Overview, Analytics, Authentication, Database, Storage, Hosting, Test Lab, Crash Reporting, Notifications, Remote Config, and Spark (Free \$0/month). The main content area features four service cards: Analytics, Authentication, Database, and Storage. Each card includes a brief description and 'Learn more' and 'GET STARTED' links.

**Analytics**  
Get detailed analytics to measure and analyse how users engage with your app  
[Learn more](#) [GET STARTED](#)

**Authentication**  
Authenticate and manage users from a variety of providers without server-side code  
[Learn more](#) [GET STARTED](#)

**Database**  
Store and sync data in real time across all connected clients  
[Learn more](#) [GET STARTED](#)

**Storage**  
Store and retrieve user-generated content like images, audio and video without server-side code  
[Learn more](#) [GET STARTED](#)



## 2. Add Firebase to your Android App \*

The screenshot shows the Firebase console interface for a project named 'CoffeeMate FBI'. The browser address bar shows the URL `console.firebase.google.com/project/coffeemate-fbi/overview`. The page features a blue header with the Firebase logo and the project name. A left sidebar contains navigation options: Overview (selected), Analytics, Authentication, Database, Storage, Hosting, Test Lab, Crash Reporting, Notifications, Remote Config, and Spark (Free \$0/month). The main content area displays a 'Welcome to Firebase! Get started here.' message with three large circular buttons: 'Add Firebase to your iOS app' (blue), 'Add Firebase to your Android app' (green, highlighted with a red rounded rectangle), and 'Add Firebase to your web app' (purple). Below these buttons is a 'Discover Firebase' section with two illustrative images.





## 2. Add Firebase to your Android App – Web Key \*

The screenshot shows the Firebase Console interface. The browser address bar displays `console.firebase.google.com/project/coffeemate-fbi/settings/general/`. The page title is "Settings" for the project "CoffeeMate FBI". The left sidebar contains navigation options: Overview, Analytics, DEVELOP (Authentication, Database, Storage, Hosting, Test Lab, Crash Reporting), GROW (Notifications, Remote Config, Dynamic Links), and EARN (AdMob). The main content area is divided into tabs: GENERAL, CLOUD MESSAGING, ANALYTICS, ACCOUNT LINKING, and SERVICE ACCOUNTS. Under the "Your project" section, the following fields are visible:

Project name	CoffeeMate FBI
Public-facing name	CoffeeMate FBI
Project ID	coffeemate-fbi
Web API Key	AlzaS[REDACTED]K7ITP8

Below the "Your apps" section, a message states: "There are currently no apps in the project. CoffeeMate FBI". Three buttons are provided to add Firebase to different types of apps:

- iOS**: Add Firebase to your iOS app
- Android**: Add Firebase to your Android app
- Web**: Add Firebase to your web app



## 2. Add Firebase to your Android App \*

### Add Firebase to your Android app

1 Register app   2 Download config file   3 Add Firebase SDK

**⚠ An OAuth2 client already exists for this package name and SHA-1 in another project. You can omit the SHA-1 for now and [read more about this situation and how to resolve it.](#)**

Android package name ⓘ

App nickname (optional) ⓘ

Debug signing certificate SHA-1 (optional) ⓘ

Required for Dynamic Links, Invites and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

CANCEL REGISTER APP  
in project CoffeeMateFBI

### Add Firebase to your Android app

1 Register app   2 Download config file   3 Add Firebase SDK

Android package name ⓘ

App nickname (optional) ⓘ

Debug signing certificate SHA-1 (optional) ⓘ

Required for Dynamic Links, Invites and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

CANCEL REGISTER APP  
in project CoffeeMateFBI

Remember to update/add to your google maps key if you change your package name (or else your map won't work!)



## 2. Add Firebase to your Android App \*

### Add Firebase to your Android app

1 Register app    2 Download config file    3 Add Firebase SDK

**Android Studio instructions**    Alternatives: [Unity](#) [C++](#)

[Download google-services.json](#)

2. Switch to the **Project** view in Android Studio to see your project root directory.

3. Move the **google-services.json** file you have just downloaded into your Android app module root directory.

*Already added the dependencies?*  
[Skip to the console](#)

**CONTINUE**



## 2. Add Firebase to your Android App \*

The screenshot shows the Firebase console interface. A modal dialog titled "Add Firebase to your Android app" is open, showing a progress indicator with three steps: 1. Enter app details, 2. Copy config file, and 3. Add to build.gradle. The dialog contains the following text and code:

The Google services plugin for [Gradle](#) loads the `google-services.json` file that you just downloaded. Modify your `build.gradle` files to use the plugin.

- Project-level build.gradle** (`<project>/build.gradle`):

```
buildscript {
  dependencies {
    // Add this line
    classpath 'com.google.gms:google-services:3.0.0'
  }
}
```
- App-level build.gradle** (`<project>/<app-module>/build.gradle`):

```
...
// Add to the bottom of the file
apply plugin: 'com.google.gms.google-services'
...
includes Firebase Analytics by default. @
```
- Finally, press "Sync now" in the bar that appears in the IDE:

A "FINISH" button is located at the bottom right of the dialog.



# CoffeeMateFBI.1.0 Dependencies (July/2017) \*

```
dependencies {
    compile fileTree(include: ['*.jar'], dir: 'libs')
    compile project(':volley')
    compile 'com.android.support:appcompat-v7:25.4.0'
    compile 'com.android.support:support-v4:25.4.0'
    compile 'com.android.support:design:25.4.0'
    compile 'com.makeramen:roundedimageview:2.2.1'
    compile 'com.android.support.constraint:constraint-layout:1.0.2'
    compile 'com.google.code.gson:gson:2.7'
    compile 'com.google.android.gms:play-services-auth:11.0.2'
    compile 'com.google.android.gms:play-services-maps:11.0.2'
    compile 'com.google.android.gms:play-services-location:11.0.2'

    compile 'com.google.firebase:firebase-core:11.0.2'
    compile 'com.google.firebase:firebase-auth:11.0.2'
    compile 'com.google.firebase:firebase-database:11.0.2'
    compile 'com.firebaseui:firebase-ui-database:2.1.0'
    testCompile 'junit:junit:4.12'
}
```

# FBDBManager – our Firebase Database utility class \*



```
public class FBDBManager {
```

```
private static final String TAG = "coffeemate";
```

```
public DatabaseReference mFirebaseDatabase;
```

```
public static String mFBUserId;
```

```
public FBDBListener mFBDBListener;
```

□ Our Firebase db reference.

```
public void open() {
```

```
    //Set up local caching
```

```
    FirebaseDatabase.getInstance().setPersistenceEnabled(true);
```

```
    //Bind to remote Firebase Database
```

```
    mFirebaseDatabase = FirebaseDatabase.getInstance().getReference();
```

```
    Log.v(TAG, "Database Connected :" + mFirebaseDatabase);
```

□ Binding to our db instance.



# FBDBManager – usage \*

```
public class CoffeeMateApp extends Application
```

```
{
```

```
    private RequestQueue mRequestQueue;  
    private static CoffeeMateApp mInstance;  
    //...
```

```
    public FBDBManager mFBDBManager;
```

```
    //...
```

```
    @Override
```

```
    public void onCreate() {
```

```
        super.onCreate();
```

```
        Log.v("coffeemate", "CoffeeMate App Started");
```

```
        mInstance = this;
```

```
        mRequestQueue = Volley.newRequestQueue(getApplicationContext());
```

```
        Log.v("coffeemate", "Adding Firebase offline persistence...");
```

```
        mFBDBManager = new FBDBManager();
```

```
        mFBDBManager.open();
```

```
    }
```

```
}
```

```
}
```

❑ Our utility class reference (inside our Application object class).

❑ Creating & ‘Opening’ a connection to our Firebase db.

# FBDBManager – our Firebase Database utility class \*



- ❑ ‘Query’ing the ‘*mFBUserId*’ node of the ‘*user-coffees*’ node.

```
public Query getAllCoffees()
```

```
{  
    Query query = mFirebaseDatabase.child("user-coffees").child(mFBUserId)  
        .orderByChild("rating");  
  
    return query;  
}
```

- ❑ As above, but only where the ‘*favourite*’ field is ‘*true*’.

```
public Query getFavouriteCoffees()
```

```
{  
    Query query = mFirebaseDatabase.child("user-coffees").child(mFBUserId)  
        .orderByChild("favourite").equalTo(true);  
  
    return query;  
}
```





# FBDBManager – usage (in CoffeeFragment) \*

```
@Override
```

```
public void onResume() {  
    super.onResume();
```

```
    query = app.mFBDBManager.getAllCoffees();
```

```
    if(favourites)
```

```
        query = app.mFBDBManager.getFavouriteCoffees();
```

```
    updateUI(query);
```

```
}
```

- ❑ Returning a ‘query’ of all coffees, (or favourite coffees) which we pass to our custom firebaseUI adapter via *updateUI()*

# FBDBManager – our Firebase Database utility class \*



- Adding an ' ValueEventListener ' and triggering our callback

```
public void getACoffee(final String coffeeKey)
{
    mFirebaseDatabase.child("user-coffees").child(mFBUserId).child(coffeeKey)
        .addValueEventListener(
            new ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot) {
                    Log.v(TAG, "The read Succeeded: " + dataSnapshot.toString());
                    mFBDBListener.onSuccess(dataSnapshot);
                }
                @Override
                public void onCancelled(DatabaseError firebaseError) {
                    Log.v(TAG, "The read failed: " + firebaseError.getMessage());
                    mFBDBListener.onFailure();
                }
            }
        );
}
```



# FBDBManager – usage (in EditFragment) \*

```
@Override
public void onResume() {
    super.onResume();

    app.mFBDBManager.attachListener(this);

    if(getArguments() != null) {
        coffeeKey = getArguments().getString("coffeeKey");
        app.mFBDBManager.getACoffee(coffeeKey);
    }

    titleBar = (TextView) getActivity()
        .findViewById(R.id.recentAddedBarTextView);
    titleBar.setText("Update a Coffee");
}
```

- ❑ Retrieving the ‘*coffeeKey*’ from the bundle and calling ‘*getACoffee()*’ - which in turn triggers the callback on the edit screen



# Firestore Console (actual data)

The image displays the Firebase Realtime Database console on the left and an Android emulator on the right. The console shows a tree view of the database with the following data:

```
user-coffees
├── 1SQVbMgN5bcLg9JXgG0ts3DNOAV2
│   ├── -KpRYrfoaJAqJZGt9Mx6
│   │   ├── address: "191 Hennessy's Road Waterford X91 PXA4"
│   │   ├── favourite: false
│   │   ├── googlephoto: "https://lh5.googleusercontent.com/-AXr-7Z4gX7k/..."
│   │   ├── latitude: 52.25
│   │   ├── longitude: -7.126
│   │   ├── name: "Firebase Coffee"
│   │   ├── price: 1.99
│   │   ├── rating: 2
│   │   ├── shop: "Fire Station"
│   │   ├── uid: "1SQVbMgN5bcLg9JXgG0ts3DNOAV2"
│   │   └── usertoken: "113437677814759908125"
│   └── -KpRZ845g8sLB4TDN0oW
│       ├── address: "5 Lismore Park Waterford Ireland"
│       ├── favourite: false
│       ├── googlephoto: "https://lh5.googleusercontent.com/-AXr-7Z4gX7k/..."
│       ├── latitude: 52.25
│       ├── longitude: -7.1399983
│       ├── name: "Coffee Latte"
│       ├── price: 2.49
│       ├── rating: 2
│       ├── shop: "Aldi"
│       ├── uid: "1SQVbMgN5bcLg9JXgG0ts3DNOAV2"
│       └── usertoken: "113437677814759908125"
├── C3PpngqTI80xuHvNtG4wZsVMhZv2
├── wotuVu5IZjhh0HRcCHRkJgsWh9y1
└── users
    ├── 1SQVbMgN5bcLg9JXgG0ts3DNOAV2
    │   └── userEmail: "dave@droidan@gmail.com"
```

The Android emulator shows the app interface with the following data:

Recently Added Coffee's		
★	Firestore Coffee Fire Station	€1.99 2.0★
★	Coffee Latte Aldi	€2.49 2.0★



# Firestore Console (actual data)

The image displays two side-by-side screenshots. The left screenshot shows the Firebase Realtime Database console for the 'coffee-mate-fbi' project. The 'user-coffees' node is expanded, showing two coffee entries. The first entry, 'Firebase Coffee', has its 'favourite' field set to 'true', which is highlighted with a red box. The second entry, 'Coffee Latte', has 'favourite' set to 'false'. The right screenshot shows an Android emulator running the 'CoffeeMateFBI' app. The app's 'Recently Added Coffee's' list displays two items: 'Firebase Coffee' (marked as a favorite with a yellow star) and 'Coffee Latte'. The 'Firebase Coffee' entry is also highlighted with a red box, matching the data in the console. The app footer shows the URL 'ddrohan.github.io'.

```
Realtime Database
├── user-coffees
│   ├── 1SQVbMgN5bcLg9JXgG0ts3DNOAV2
│   │   ├── -KpRYrfoaJAqJZGt9Mx6
│   │   │   ├── address: "191 Hennessy's Road Waterford X91 PXA4"
│   │   │   ├── favourite: true
│   │   │   ├── googlephoto: "https://lh5.googleusercontent.com/-AXr-7Z4gX7k/..."
│   │   │   ├── latitude: 52.25
│   │   │   ├── longitude: -7.126
│   │   │   ├── name: "Firebase Coffee"
│   │   │   ├── price: 1.99
│   │   │   ├── rating: 2
│   │   │   ├── shop: "Fire Station"
│   │   │   ├── uid: "1SQVbMgN5bcLg9JXgG0ts3DNOAV2"
│   │   │   └── usertoken: "113437677814759908125"
│   │   └── -KpRZ845g8sLB4TDN0oW
│   │       ├── address: "5 Lismore Park Waterford Ireland"
│   │       ├── favourite: false
│   │       ├── googlephoto: "https://lh5.googleusercontent.com/-AXr-7Z4gX7k/..."
│   │       ├── latitude: 52.25
│   │       ├── longitude: -7.1399983
│   │       ├── name: "Coffee Latte"
│   │       ├── price: 2.49
│   │       ├── rating: 2
│   │       ├── shop: "Aldi"
│   │       ├── uid: "1SQVbMgN5bcLg9JXgG0ts3DNOAV2"
│   │       └── usertoken: "113437677814759908125"
│   └── C3PpngqTI80xuHvNtG4wZsVMhZv2
│   └── wotuVu5IZjhh0HRcCHRkJgsWh9y1
└── users
    ├── 1SQVbMgN5bcLg9JXgG0ts3DNOAV2
    │   └── userEmail: "daveydrohan@gmail.com"
```

# Pricing

<https://firebase.google.com/pricing/>



Products	Spark Plan Generous limits for hobbyists Free	Flame Plan Fixed pricing for growing apps \$25/month	Blaze Plan <a href="#">Calculate pricing for apps at scale</a> Pay as you go
<b>Free Products</b> Authentication (except Phone Auth), Analytics, App Indexing, Dynamic Links, Invites, Remote Config, Cloud Messaging (FCM), Performance Monitoring, and Crash Reporting.	✓ Included	✓ Included Free	✓ Included Free
<b>Realtime Database</b> Simultaneous connections <span>?</span> GB stored GB downloaded Automated backups	100 1 GB 10 GB/month ✗	100K/instance 2.5 GB 20 GB/month ✗	100K/instance \$5/GB \$1/GB ✓



---

Thanks!

A simple line drawing of a smiling person with their arms raised in a celebratory gesture. The person has a round face with a wide smile, two dots for eyes, and a few strands of hair. Their arms are raised, and they appear to be jumping or dancing. A small copyright symbol is visible at the bottom right of the drawing.



# Some important points though...

---

- ❑ Do not think RDBMS, think JSON. How data should be structured is very important.
- ❑ Firebase has a recycler view, that integrates with real time database smoothly without any listeners. (FirestoreUI)
- ❑ Test lab which is available in paid plan (Blaze), is an amazing feature for testing your app on different real and virtual devices (next section)
- ❑ Set developer mode to true when testing Remote Config (next section).





# References & Links

---

- ❑ [Presentation by Kaushal Dhruw & Shakti Moyal 2016](#)
- ❑ <https://firebase.google.com>
- ❑ Demo app available at <https://goo.gl/WBP5fR>



---

# Questions?