

# Android

-

# Firestore



Peter Borovanský  
KAI, I-18

MS-Teams: [2sf3ph4](#), [List](#), [github](#)  
borovan 'at' ii.fmph.uniba.sk

## •Room

- @Database
- @Entity
- @Dao

## •Firebase

- Authentication
  - Email/Password, Google, FB, Twitter
- Realdatabase
- Storage
- Push notifications

# Kam (inam) uložiť naše dáta

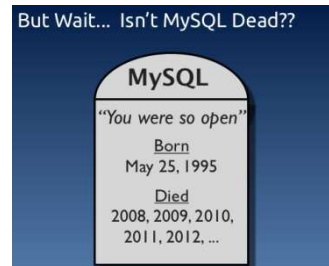
bolo minule:

- máme vlastný server – protokol najčastejšie http-https
  - najčastejšie (v bakalárkach) AMP – Apache-MySQL-PHP, prip. java-servlets
  - tak trochu doba minulá...

[LinuxDays 2017:](#)

[MySQL sežere Vaše data](#)

[David Karban](#)



W - Windows  
A - Apache  
M - MySQL  
P - PHP  
WAMP - Server

bude:

- lokálne
  - Sqlite <https://developer.android.com/training/data-storage/sqlite>
  - Room <https://developer.android.com/topic/libraries/architecture/room>
- cloudové úložiská a ich služby
  - Relačné - tabuľkovo orientované
    - [Parse.com](#) <http://parseplatform.org/> (kúpil Facebook)
  - nosql – json
    - [firebase.com](#) (Google)





# SQLite vs. Room

---

SQLite databáza často

- obsahovala veľa boiler-plate kódu
- operácie, ktoré blokovali main-thread
- sql dotazy, ktoré sa konštruujú a testujú v run-time

od 2017 Android má Room, ktorá umožňuje

- používať komponenty Room (@Entity, @Dao, @Database)
- udržiavať relácie medzi entitami (keys)
- púšťať DB operácie mimo main-thread, prirodzene pomocou korutín



# Room

- je náhrada za bývalú SQLite, ktorá existuje v Androide od API-1

```
build.gradle
dependencies {
    implementation "androidx.room:room-runtime:2.2.5"
    kapt "androidx.room:room-compiler:2.2.5"
    implementation "androidx.room:room-ktx:2.2.5"
}
```

```
build.gradle
plugins {
    . . .
    id 'kotlin-android-extensions'
    id 'kotlin-kapt'
}
```

- **@Database** – abstraktná trieda RoomDatabase
  - Room.databaseBuilder() persistentná inštancia, dáta existujú aj po skončení procesu,
  - Room.inMemoryDatabaseBuilder() – dáta zmiznú keď proces zanikne
- **@Entity** – tabuľky v SQL databáze
- **@DAO** – data access object – metódy na prístup k databáze



# RoomDB

- Room je vylepšená SQLite, ktorá existuje v Androide od API-1
- vytvoríme aplikáciu na registrovanie študentov s funkciami:
  - signup/login/logout/delete
- v návrhovom vzore MVVM
- s použitím corutín
- obohatíme build.gradle (app) o
- room

```
implementation "androidx.room:room-runtime:2.2.5"  
kapt "androidx.room:room-compiler:2.2.5"  
implementation "androidx.room:room-ktx:2.2.5"
```

- coroutines

```
implementation "org.jetbrains.kotlinx:kotlinx-coroutines-core:1.4.1"  
implementation "org.jetbrains.kotlinx:kotlinx-coroutines-android:1.4.1"
```

- plugins

```
plugins {  
    id 'com.android.application'  
    id 'kotlin-android'  
    id 'kotlin-android-extensions'  
    id 'kotlin-kapt'}
```



# @Entity

---

```
@Entity
data class Student (
    val isic          : String,
    val name          : String,
    val passwordHash  : Int,
    @ColumnInfo(passwordHash = "password_hash")
    val description   : String) {
    @PrimaryKey(autoGenerate = true)
    var id: Long = 0
}
```

# @Entity

foreignKeys/Embedded

```
@Entity
data class User(
    @PrimaryKey val userId: Long,
    val name: String,
    val age: Int
)
@Entity(foreignKeys = [
    ForeignKey(
        entity = User::class,
        parentColumns = ["userId"],
        childColumns = ["userOwnerId"],
        onDelete = CASCADE)]
)
```

```
data class Library(
    @PrimaryKey
    val libraryId: Long,
    val title: String,
    val userOwnerId: Long
)
data class UserAndLibrary(
    @Embedded val user: User,
    @Relation(
        parentColumn = "userId",
        entityColumn = "userOwnerId"
    )
    val library: Library
)
```



@Dao

```
interface StudentDAO {  
    @Insert(onConflict = OnConflictStrategy.REPLACE)  
    suspend fun insert(student: Student): Long  
  
    @Query("SELECT * FROM Student WHERE name = :name")  
    suspend fun getName(name: String): Student?  
  
    @Query("SELECT * FROM Student WHERE id = :id")  
    suspend fun getID(id: Long): Student?  
  
    @Query("SELECT * FROM Student WHERE isic = :isic")  
    suspend fun getISIC(isic: String): Student?  
  
    @Query("DELETE FROM Student WHERE id = :id")  
    suspend fun deleteID(id: Long)  
  
    @Insert  
    suspend fun insertAll(vararg students: Student)  
  
    @Delete  
    suspend fun delete(student: Student)  
}
```





# @Database

---

```
@Database(entities = arrayOf(Student::class), version = 1)
abstract class StudentDatabaseEasy: RoomDatabase() {
    abstract fun studentDAO(): StudentDAO

    fun getInstance(context: Context) = Room.databaseBuilder(
        context.applicationContext,
        StudentDatabase::class.java,
        "studentdatabase"
    ).build()
}
```

```
coroutineScope.Launch {
    if (db.getName(name) != null || db.getISIC(isic) != null) {
        withContext(Dispatchers.Main) {
            error.value = "Student already exists"
        }
    }
}
```



```
fun login(name: String, password: String) {  
    coroutineScope.launch {  
        val student = db.getName(name)  
        if (student == null)  
            withContext(Dispatchers.Main) {  
                error.value = "Student not found"  
            }  
        else {  
            if (student.passwordHash == password.hashCode()) {  
                Status.login(student)  
                withContext(Dispatchers.Main) {  
                    logged.value = true  
                }  
            } else {  
                withContext(Dispatchers.Main) {  
                    error.value = "Password is incorrect"  
                }  
            }  
        }  
    }  
}
```

# Rozsiahlejší príklad @Entity

```
@Entity(foreignKeys = [ForeignKey(
    entity = Company::class,
    parentColumns = ["id"],
    childColumns = ["company_id"],
    onDelete = ForeignKey.CASCADE)])
```

```
data class Employee (
    @ColumnInfo(name = "name")
    val name: String,

    @ColumnInfo(name = "company_id")
    val companyId : Int = 0) ←
{
    @PrimaryKey(autoGenerate = true)
    @ColumnInfo(name = "id")
    val employeeId : Int = 0
}
```

```
@Entity(tableName = "Department")
class Department(
    @ColumnInfo(name = "name")
    val name: String,

    @ColumnInfo(name = "company_id")
    val companyId : Int = 0)
{
    @PrimaryKey(autoGenerate = true)
    @ColumnInfo(name = "id")
    var id = 0
}
```

```
@Entity(tableName = "Company")
data class Company (
    @ColumnInfo(name = "name")
    val name: String,

    @ColumnInfo(name = "date_updated")
    @TypeConverters(DateConverter::class)
    val itemUpdatedDate: Date? = null,

    @Embedded
    private val location: Location? = null,

    @Embedded(prefix = "hq_")
    private val headLocation: Location? = null,

    @Ignore
    val picture: Bitmap? = null
) {
    @PrimaryKey
    @ColumnInfo(name = "id")
    val companyId = 0 ←
}
```

```
@Entity(primaryKeys = ["id", "code"])
class Office {
    val id : Int = 0
    var code: String
}
```

# Rozsiahlejší príklad

## @Dao

```
@Dao
interface EmployeeDao {
    @get:Query("SELECT * FROM Employee")
    val allEmployees: LiveData<List<Employee?>>?
    @RawQuery
    fun getAllEmployeesWithLimit(query: String?): List<Employee?>
    @Insert
    fun insertEmployee(employee: Employee?)
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    fun insertAll(employees: List<Employee?>)
    @Update
    fun updateEmployee(employee: Employee?)
    @Delete
    fun deleteEmployee(employee: Employee?)
}
```

```
@Dao
abstract class DepartmentDao {
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    abstract fun insertAll(departments: List<Department?>)
    @Insert
    abstract fun insert(product: Department?)
    @Delete
    abstract fun delete(product: Department?)
    @Transaction
    fun insertAndDeleteInTransaction(
        newDepartment: Department?,
        oldDepartment: Department?) {
        insert(newDepartment)
        delete(oldDepartment)
    }
}
```

```
@Dao
interface CompanyDao {
    @get:Query("SELECT * FROM Company")
    val allCompanies: LiveData<List<Company?>>?
    @get:Query("SELECT * FROM Company ORDER BY name")
    val allCompaniesOrdered: LiveData<List<Company?>>?
    @Insert
    fun insertCompany(company: Company?)
    @Query(
        "SELECT * FROM Company WHERE name LIKE :companyName")
    fun getCompanies(
        companyName: String?):
        LiveData<List<Company?>>?
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    fun insertAll(companies: List<Company?>)
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    fun insertAll(vararg companies: Company?)
    @Update
    fun updateCompany(company: Company?)
    @Update
    fun updateCompanies(vararg company: Company?)
    @Delete
    fun deleteCompany(company: Company?)
    @Delete
    fun deleteCompanies(vararg company: Company?)
}
```



# Cvičenie - B

Vytvorte aplikáciu, ktorá slúži na **evidenciu známok študentov** s nasledujúcimi entitami (verím, že tomu zápisu rozumiete). Máte urobiť Room model a minimálne základné GUI, aby bolo jasné, že viete pracovať s Room.

Základná verzia na hodnotenie:

- vie pridať položku do troch tabuliek (delete nemusíte riešiť), Znamky sú statický číselník,
- zobrazuje počet študentov, predmetov a hodnotení.

**Bonus: [1 bod]** nejaký listview zobrazuje všetky hodnotenia, v ľub. poradí, bez filtrov,

**Bonus: [1 bod]** viete zmazať študenta/predmet, ktorý už má hodnotenie, CASCADE...

```
drop table Student;
create table Student (
    id NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY,
    isic varchar(20) not null,
    meno varchar(20),
    priezvisko varchar(30),
    CONSTRAINT sid_pk PRIMARY KEY (id),
    CONSTRAINT isic_pk UNIQUE (isic)
);
insert into Student (isic, meno, priezvisko)
    values ('123456789', 'Sansa', 'Starkova');
commit;
drop table predmet;
create table Predmet (
    id NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY,
    kod varchar(20) not null,
    nazov varchar(80),
    CONSTRAINT pid_pk PRIMARY KEY (ID),
    CONSTRAINT kod_pk UNIQUE (kod)
);
insert into Predmet (kod, nazov)
    values ('1-AIN-472/12',
        'Vývoj mobilných aplikácií, zimný semester 2020/2021');
commit;
```

```
create table Znamka (
    ID NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY,
    text varchar(5),
    CONSTRAINT zid_pk PRIMARY KEY (ID),
    CONSTRAINT text_pk UNIQUE (text)
);
insert into Znamka (text) values ('A');
insert into Znamka (text) values ('B');
insert into Znamka (text) values ('C');
insert into Znamka (text) values ('D');
insert into Znamka (text) values ('E');
insert into Znamka (text) values ('Fx');
create table Hodnotenie (
    studentID NUMBER,
    predmetID NUMBER,
    znamkaID NUMBER,
    CONSTRAINT fk_student FOREIGN KEY (studentID)
        REFERENCES Student(ID),
    CONSTRAINT fk_predmet FOREIGN KEY (predmetID)
        REFERENCES Predmet(ID),
    CONSTRAINT fk_znamka FOREIGN KEY (znamkaID)
        REFERENCES Znamka(ID)
);
```



# Ďalšie čítanie - Room

## tutoriály

---

- **Room Persistence Library: Introduction:**  
<https://medium.com/@magdamiu/android-room-persistence-library-97ad0d25668e>
- **Room Persistence Library: Entity, Dao, Database:**  
<https://medium.com/@magdamiu/android-room-components-5a7458b99191>
- **Room Persistence Library: Relations:**  
<https://medium.com/@magdamiu/android-room-persistence-library-relations-75bbe02e8522>
- **Room Persistence Library: Queries and Migration Support:**  
<https://medium.com/@magdamiu/android-room-persistence-library-queries-and-migration-support-a9f21d2dc9d8>

# Cvičenie - B

Malá evidencia produktov pomocou bar code scannera

EAN 8584004040108

Horalka



[See on Ebay](#)

Brand	<a href="#">Sedita</a>
Manufacturer	<a href="#">I.D.C. Holding</a>
EAN	8584004040108
Country	Slovakia
Last Scan	Nov 25 2020 at 11:17 PM
GS1 Name	I.D.C. HOLDING, odš.závod Pečivárne
GS1 Address	Drieňová 3 Bratislava SK
Description	No description for 8584004040108
Barcode	



8 584004 040108



Parse + facebook

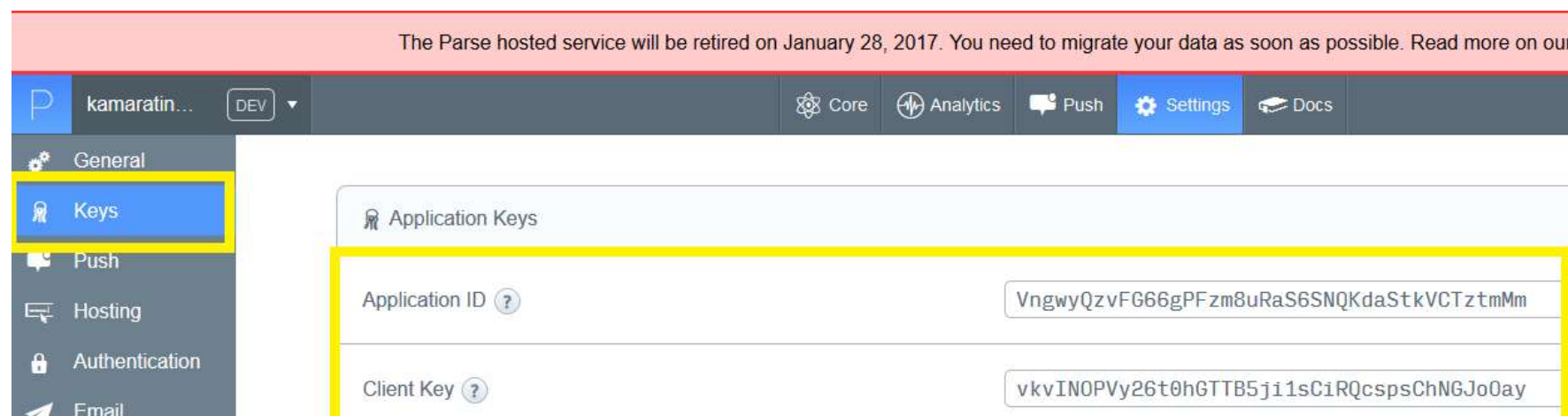
# Parse vs. Parse Server

- API pre komunikáciu mobilných/web aplikácií, ukladanie/zdieľanie dát
- podporoval PUSH notifikácie pomocou Google Cloud Messaging (GCM) vaša aplikácia dostane notifikáciu zo servera, ak iný užívateľ vyvolá event

Parse.com kúpený FB 2013 (free) končí v 2017 ☹

Ponúka migráciu na open-source Parse Server

- s veľmi podobným API 😊 
- na vlastnom serveri, s infraštruktúrou Node.js + MongoDB + Python





# Alternatívy

(k Parse Server)

Veci zadarmo sú (často) síce najlepšie, ale zase pomíjivé  
Treba pozrieť konkurenciu:

- Amazon AWS – iOS, Android, Web, ReactNative, ...

<https://aws.amazon.com/amplify/>



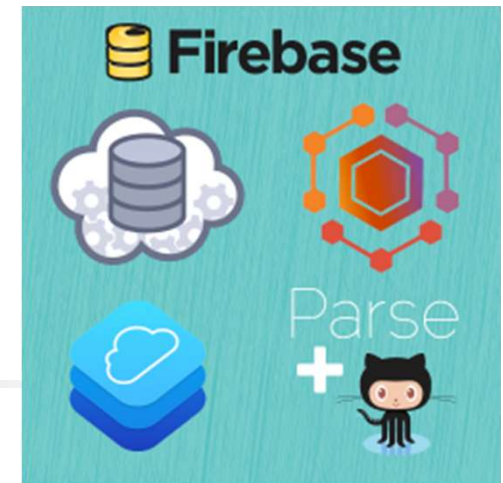
- MS Azure Mobile App SDK – iOS, Android, Xamarin, Cordova, ...



<https://docs.microsoft.com/en-us/azure/app-service-mobile/app-service-mobile-android-how-to-use-client-library>

- Firebase – iOS, Android, Flutter, Web, ...







- realtime JSON oriented DB,
- PUSH notifikácie, analytics, REST API



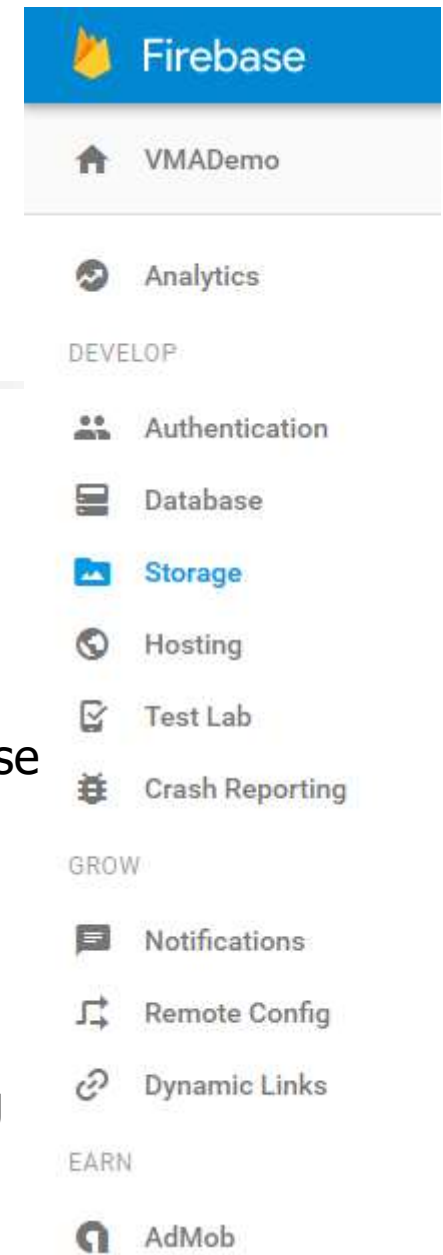


Startup Firebase(2011) kúpil Google(2014)

Ponúka služby/servisy:
























- Authentication – FB/Git/Google/Twitter/FirebaseEmail
- Realtime database – non sql store & synchronize database
- Cloud Firestore       
- File storage – secure upload & download files
- Test lab for Android – rôzne virtuálne zariadenia
- Cloud messaging – push notifikácie pomocou  
Google Cloud Messaging/Firebase Cloud Messaging
- Crash reporting
- Analytics
- ...

<https://firebase.google.com/docs?authuser=0>













# Firebase Products






























## Build your app

-  Cloud Firestore  
iOS  
-  Firebase ML  
iOS 
-  Cloud Functions  
iOS   C++ 
-  Authentication  
iOS   C++ 
-  Hosting  

-  Cloud Storage  
iOS   C++ 
-  Realtime Database  
iOS   C++ 

## Improve app quality

-  Crashlytics  
iOS  
-  App Distribution  
iOS 
-  Performance Monitoring  
iOS  
-  Test Lab  
iOS 

## Grow your business

-  Analytics  
iOS   C++ 
-  Extensions  
iOS   C++ 
-  Predictions  
iOS  C++ 
-  Firebase A/B Testing  
iOS  C++ 
-  Cloud Messaging  
iOS   C++ 
-  In-App Messaging  
iOS 
-  Remote Config  
iOS   C++ 
-  Dynamic Links  
iOS  C++ 
-  App Indexing  
iOS 

# Firebase Console

<https://console.firebase.google.com/project/kamaratinamape/database/data>

The screenshot shows the Firebase Authentication console for the project 'kamaratinamape'. The 'USERS' tab is selected. A modal dialog titled 'Add an Email/Password user' is open, showing input fields for 'Email' and 'Password'. The email 'vmaandroid@yahoo.com' and password 'androidVMA2016' are entered and highlighted with red boxes. The 'ADD USER' button is visible at the bottom right of the dialog.

Email	Providers	Created	Signed In	User UID ↑
-------	-----------	---------	-----------	------------

- ukážky sú robené pomocou tohoto účtu
- login a password vidíte v obrázku



Authentication

iOS C++

# Autentifikácia

Sign-in metódy:

- vlastná autentifikácia cez email/password
  - môžete definovať viacero email-účov
- cez FB/Google/Twitter/GitHub API

SIGN-IN METHOD		EMAIL TEMPLATES
Sign-in providers		
Provider	Status	
Email/Password	Enabled	
Google	Enabled	
Facebook	Enabled	
Twitter	Enabled	
GitHub	Disabled	
Anonymous	Enabled	

## Authentication

[Users](#) [Sign-in method](#) [Templates](#) [Usage](#)

pre začiatok odporúčam  
skúsiť aplikácie z balíka

GITHUB:  
[quickstart-android-master](https://github.com/firebase/quickstart-android)

Search by email address, phone number, or user UID					Add user
Identifier	Providers	Created	Signed In	User UID ↑	
(anonymous)		Dec 1, 2016	Dec 1, 2016	1s70KNetLsYbmZO7ZE5qsHvah7q2	
vmaandroid@yahoo.com		Nov 22, 2016	Dec 13, 2018	4eWsnHhdRPVgUAtV200q7V1gN...	
—		Dec 1, 2016	Dec 2, 2016	5UoBQBMAcXQ9rj2060MsG80IOJ...	
(anonymous)		Dec 2, 2018	Dec 2, 2018	7v4jPhXKQBVz5zq86oUuhIkAW1t1	
prostrediahm@gmail.com		Dec 7, 2019	Dec 7, 2019	861aiShu8OSzo8Z7v5Ms6cHU6RI2	

Project:auth.zip

<https://github.com/firebase/quickstart-android>



Authentication

iOS C++

# Sign-in methods

<https://console.firebase.google.com/u/1/project/kamaratinamape/authentication/providers>

## Authentication

Users Sign-in method Templates Usage

### Sign-in providers

Provider	Status
Email/Password	Enabled
Phone	Disabled
Google	Enabled
Play Games	Disabled
Game Center <span>Beta</span>	Disabled
Facebook	Enabled
Twitter	Enabled
GitHub	Disabled
Yahoo	Disabled
Microsoft	Disabled
Apple <span>Beta</span>	Disabled
Anonymous	Enabled



Authentication

iOS C++

# Project Setting

<https://console.firebase.google.com/u/0/project/kamaratinamape/settings/general/android:com.example.firebasedemo1>

Your apps

- meno projektu (support email)
- rovnaké API pre všetky Android apps (package)
- SHA1-certificatite
- google-services.json

Add app

google-services.json

such as keys and  
identifiers, for the services you just enabled.

- com.google.firebase.quickstart.an...
- com.google.firebase.quickstart.auth
- FBAuth  
com.google.firebase.quickstart.auth.j...
- quickstart  
com.google.firebase.quickstart.datab...
- FBDatabase  
com.google.firebase.quickstart.datab...
- com.google.firebase.quickstart.fcm
- FBMessaging  
com.google.firebase.quickstart.fcm.ja...

App ID

1:539843735083:android:87fcb66cc6ca320a

App nickname

Add a nickname

Package name

com.google.firebase.quickstart.auth

SHA certificate fingerprints

Type

7a:94:75:11:dd:3d:57:2a:36:ed:2a:f2:76:13:a0:b8:68:0f:67:f1

SHA-1

[Add fingerprint](#)

# google-services .json

```
{ "project_info": {  
  "project_number": "539843735083",  
  "firebase_url": "https://kamaratinamape.firebaseio.com",  
  "project_id": "kamaratinamape",  
  "storage_bucket": "kamaratinamape.appspot.com"  
},  
  "client": [  
    { "client_info": {  
      "mobilesdk_app_id": "1:539843735083:android:e4c17d2977753b25",  
      "android_client_info": { "package_name": "sk.uniba.fmph.dai.borovan.fbdemo"  
    }  
  },  
    { "oauth_client": [{  
      "client_id": "539843735083-e4n6dg61g1npk7uka8ebf2rhcmg4t7v1.apps.googleusercontent.com",  
      "client_type": 3  
    }  
  },  
    {  
      "api_key": [ { "current_key": "AIzaSyCbfmtNkbnhj1qanA051uSfQ11_PTjPa8" } ],  
      "services": {  
        "analytics_service": { "status": 1 },  
        "appinvite_service": { "status": 1, "other_platform_oauth_client": [] },  
        "ads_service": { "status": 2 }  
      }  
    }  
  ],  
  "configuration_version": "1"  
}
```

```
\quickstart-android-master\auth\app  
.  
..  
.gitignore  
app.iml  
auth-app.iml  
build  
build.gradle  
google-services.json  
proguard-rules.pro  
src  
58 bytes  
64 bytes free
```

Tento súbor potrebujete  
mať v projekte, stiahnite  
a do pod-adresára \app





Authentication

iOS C++

# Autentifikácia cez FB

<https://firebase.google.com/docs/auth/android/facebook-login>



Facebook

Enable ☒

App ID

1286286781442318

App secret

663845afe79e444fafa94932a5d2cf7f

To complete set up, add this OAuth redirect URI to your Facebook app configuration. [Learn more](#)

[https://kamaratinamape.firebaseio.com/\\_/auth/handler](https://kamaratinamape.firebaseio.com/_/auth/handler)

Autentifikácia cez FB/Twitter/GitHub API:

CANCEL

SAVE

- predpokladá, že registrujete aplikáciu na FB/Twitter/Git developerskej konzole napr. <https://developers.facebook.com/apps/1286286781442318/settings/basic/>
- kde dostanete nejakú analógiu APP ID/Secret key
- tie zapíšete do Firebase API vašej Firebase appky
- Firebase vám vygeneruje **google-services.json**, ktorý zakompilujete do .apk

Project:auth.zip

<https://github.com/firebase/quickstart-android>



Authentication

iOS

# Facebook for developers

(dev konzola od FB)

**facebook** for developers Docs Tools Support My Apps

**FirebaseAuthApp**

APP ID: 1286286781442318 OFF Status: In D

Dashboard

Settings

Basic

Advanced

Roles

Alerts

App Review

PRODUCTS

Facebook Login

App ID

1286286781442318

App Secret

663845afe79e444fafa94932a5d2cf7f Reset

Display Name

FirebaseAuthApp

Namespace

App Domains

Contact Email

borovansky@gmail.com

Privacy Policy URL

https://kamaratinamape.firebaseio.com/\_\_/auth/handler

Terms of Service URL

Terms of Service for Login dialog and App Details

<https://developers.facebook.com/apps/1286286781442318/settings/basic/>



Authentication

iOS C++

# Facebook Sign-in

- App ID a App Secret zapíšete do Firebase Console do
- Authentication/Sign-in methods/Facebook



Facebook

☒ Enable

App ID

1286286781442318

App secret

663845afe79e444fafa94932a5d2cf7f

To complete set up, add this OAuth redirect URI to your Facebook app configuration. [Learn more](#)

`https://kamaratinamape.firebaseio.com/_/auth/handler`

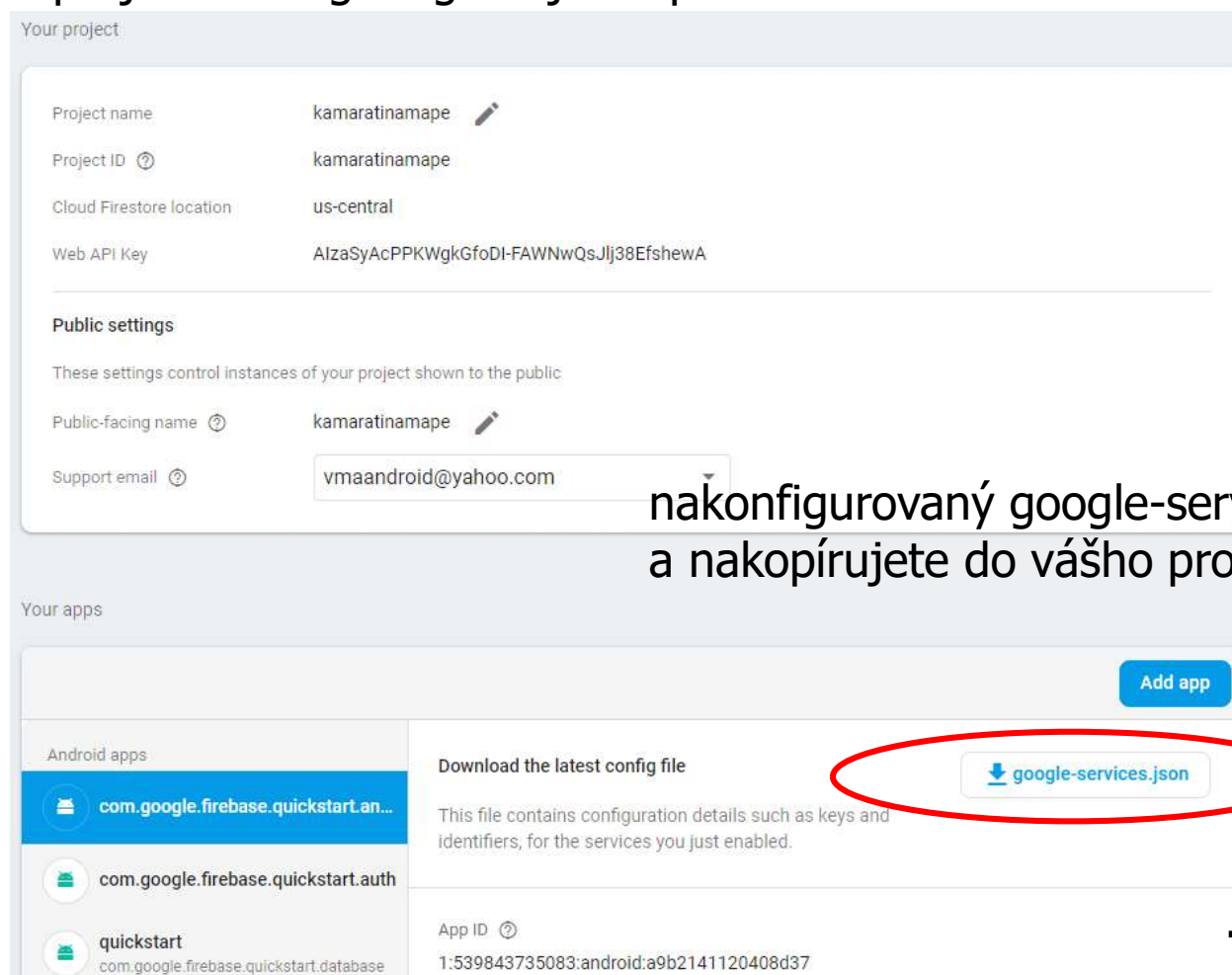
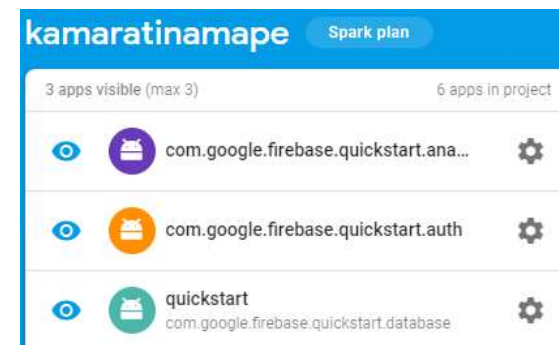
Cancel

Save



# Registrovanie appky

v project settings registrujete aplikáciu



nakonfigurovaný google-services.json stiahnete a nakopírujete do vášho projektu, do ...app/

... a skompilujete



Authentication

iOS C++

# Autentifikácia cez Google

<https://developers.google.com/android/guides/client-auth>

Musíte do Firebase projektu/aplikácie vložiť svoj SHA1 kľúč (viac bolo minule)  
Google sign-in is automatically configured on your connected iOS and web apps. To set up Google sign-in for your Android apps, you need to add the [SHA1 fingerprint](#) for each app [Project Settings](#).

Android apps

- com.google.firebase.quickstart.an...
- com.google.firebase.quickstart.auth
- FBAuth  
com.google.firebase.quickstart.auth.j...
- quickstart  
com.google.firebase.quickstart.datab...
- FBDatabase  
com.google.firebase.quickstart.datab...
- com.google.firebase.quickstart.fcm
- FBMessaging  
com.google.firebase.quickstart.fcm.ja...

Download the latest config file

[google-services.json](#)

This file contains configuration details such as keys and identifiers, for the services you just enabled.

App ID

1:539843735083:android:87fcb66cc6ca320a

App nickname

Add a nickname

Package name

com.google.firebase.quickstart.auth

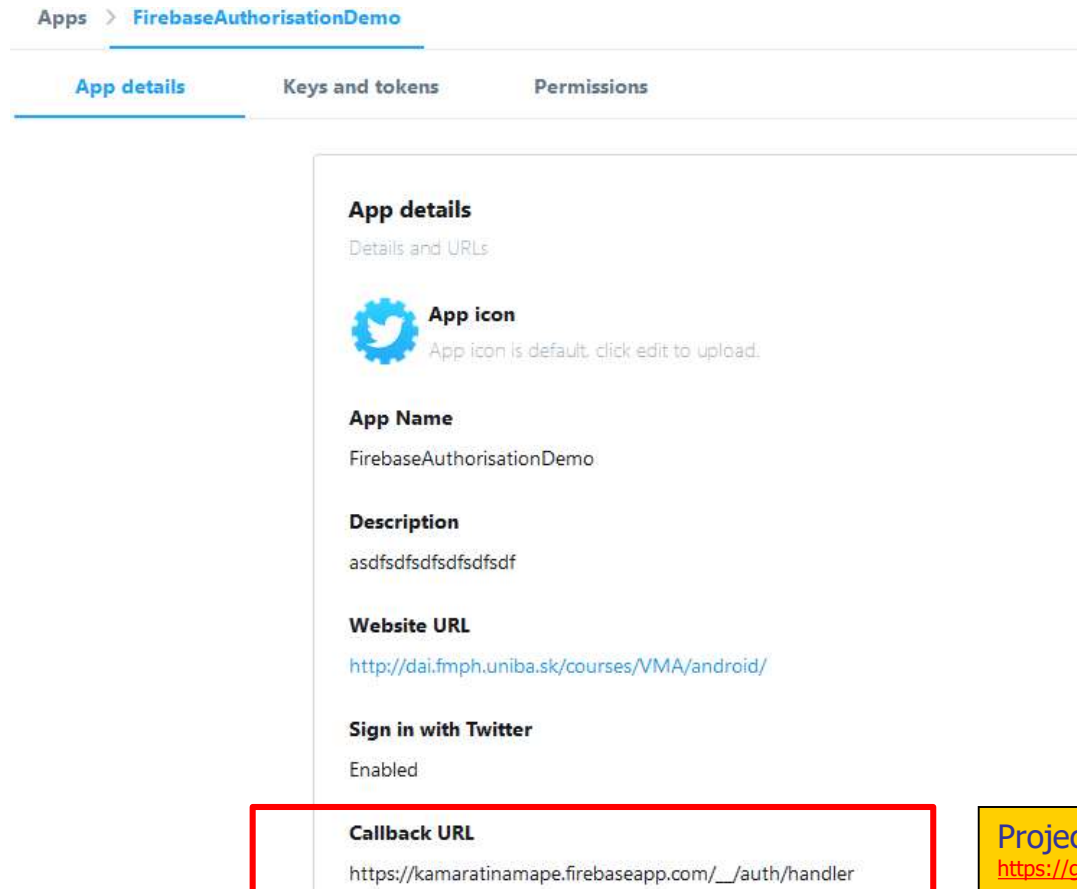
SHA certificate fingerprints	Type
7a:94:75:11:dd:3d:57:2a:36:ed:2a:f2:76:13:a0:b8:68:0f:67:f1	SHA-1

# Autentifikácia cez Twitter

<https://firebase.google.com/docs/auth/android/twitter-login>

<https://developer.twitter.com/en/apps/13160641>

Na Twitter developerskej konzole musíte registrovať aplikáciu, a získaťe Consumer API keys a Access token, ten prezradíte Firebase Console



The screenshot shows the 'App details' page in the Twitter Developer Console. The page has three tabs: 'App details' (selected), 'Keys and tokens', and 'Permissions'. The 'App details' section includes the following fields:

- App icon:** A blue Twitter bird icon. Below it, text says 'App icon is default, click edit to upload.'
- App Name:** 'FirebaseAuthorisationDemo'
- Description:** 'asdfsdfsdfsdfsdfsdf'
- Website URL:** 'http://dai.fmph.uniba.sk/courses/VMA/android/'
- Sign in with Twitter:** 'Enabled'
- Callback URL:** 'https://kamaratinamape.firebaseio.com/\_/auth/handler' (This field is highlighted with a red border in the original image).

Project:auth.zip

<https://github.com/firebase/quickstart-android>

# Autentifikácia cez Twitter

<https://developer.twitter.com/en/apps/13160641>

Apps > **FirebaseAuthorisationDemo**

App details

**Keys and tokens**

Permissions

## Important notice about your access token and access token secret

To make your API integration more secure, we will no longer show your access token and access token secret beyond the first 10 characters. You will be able to regenerate it at anytime here, which will invalidate your current access token and secret. Please save this information carefully, as you will not be able to retrieve it again. Only your API keys, which will still be shown here as they are below. To learn more, [visit the Forums](#).

## Keys and tokens

Keys, secret keys and access tokens management.

### Consumer API keys

k7YuSJH9qjJLeZn51N3TzCV0c (API key)

Xpz88UjjjTVQU61IkQWHxIOyiQIfkRJPu7qHAY0V8311NCfCtu (API secret key)

Regenerate

### Access token & access token secret

492951543-CZmXbG2ad1LICjIU8H8RJo2ZmRIWLpWK4JLtzQW (Access token)

Sp8zOMrLSAw7mCarrLnpNwmshYFj5Eonwg9JsQG8DtSn9 (Access token secret)

Read-only (Access level)

**Project: auth.zip**

<https://github.com/firebase/quickstart-android>



# Autentifikácia cez Twitter

<https://developer.twitter.com/en/apps/13160641>

API KEY a SECRET treba vložiť do Firebase Console pre Twitter Authentication



☒ Enable

API key

k7YuSJH9qjJLeZn51N3TzCV0c

API secret

Xpz88UjjJTVQU61IkQWHxIOyiQIfkRJPU7qHAY0V83I1NCfCtu

To complete set up, add this callback URL to your Twitter app configuration. [Learn more](#)

[https://kamaratinamape.firebaseio.com/\\_\\_/auth/handler](https://kamaratinamape.firebaseio.com/__/auth/handler)



Cancel

Save

Project:auth.zip

<https://github.com/firebase/quickstart-android>

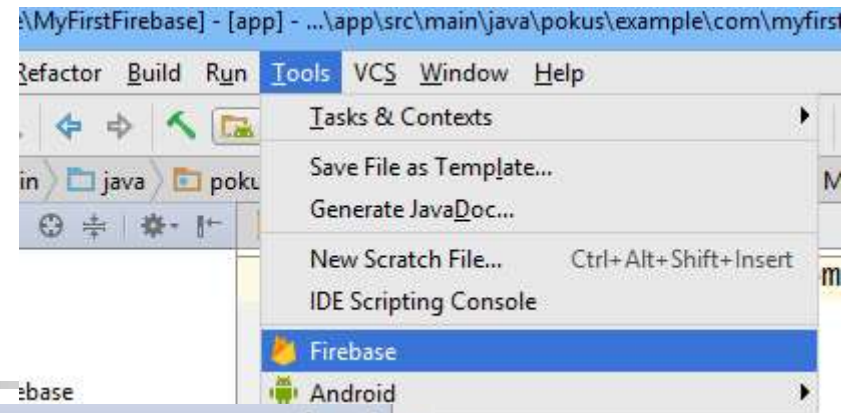





Authentication









iOS   C++ 

# Firestore v AS



 **Firebase**

Firebase gives you the tools and infrastructure from Google to help you develop, grow and earn money from your app. [Learn more](#)

- ▶  **Analytics**  
Measure user activity and engagement with free, easy, and unlimited analytics. [More info](#)
- ▶  **Cloud Messaging**  
Deliver and receive messages and notifications reliably across cloud and device. [More info](#)
- ▼  **Authentication**  
Sign in and manage users with ease, accepting emails, Google Sign-In, Facebook and other login providers. [More info](#)  
 [Email and password authentication](#)
- ▶  **Realtime Database**  
Store and sync data in realtime across all connected clients. [More info](#)
- ▶  **Storage**  
Store and retrieve large files like images, audio, and video without writing server-side code. [More info](#)
- ▶  **Remote Config**  
Customize and experiment with app behavior using cloud-based configuration parameters. [More info](#)
- ▶  **Test Lab**



# Autentifikačný kit v AS

Assistant

← Firebase > Authentication

## Email and password authentication

You can use Firebase Authentication to let your users sign in with their email addresses and passwords, and to manage your app's password-based accounts. This tutorial helps you set up an email and password system and then access information about the user.

[Launch in browser](#)

- 1 **Connect your app to Firebase**  
[Connect to Firebase](#)
- 2 **Add Firebase Authentication to your app**  
[Add Firebase Authentication to your app](#)  

To use an authentication provider, you need to enable it in the [Firebase console](#). Go to the Sign-in Method page in the Firebase Authentication section to enable Email/Password sign-in and any other identity providers you want for your app.
- 3 **Listen for auth state**  

Declare the `FirebaseAuth` and `AuthStateListener` objects.

```
private FirebaseAuth mAuth;
```

```
private FirebaseAuth.AuthStateListener mAuthListener;
```

In the `onCreate()` method, initialize the `FirebaseAuth` instance and the `AuthStateListener` method so you can track whenever the user signs in or out.

```
mAuth = FirebaseAuth.getInstance();
```

Ponúka vám to step-by-step návod na vytvorenie rôznych typov Firebase aplikácií

Add Firebase Authentication to your app

Performing this action will make the following changes to your project.

```
build.gradle (project-level)

Add Firebase Gradle buildsript dependency
classpath 'com.google.gms:google-services:4.0.1'
```

---

```
app/build.gradle

Add Firebase plugin for Gradle
apply plugin: 'com.google.gms.google-services'
```

build.gradle will include these new dependencies:

```
compile 'com.google.firebase:firebase-auth:16.0.1:15.0.0'
```

This will also enable the firebase-core library which includes Firebase Analytics. [Learn more](#)

[Accept Changes](#) [Cancel](#)

# 1) Connect your app to Firebase

Success!

You've signed in to Android Studio.

To continue, go back to Android Studio.



Explore Google services you can now use in your Android app:



Firebase



Google Cloud Platform

Connect to Firebase ✕

**Firebase**

☒ Create new Firebase project [What's this?](#) Signed in as vmaandroid@yahoo.com [Sign out](#)

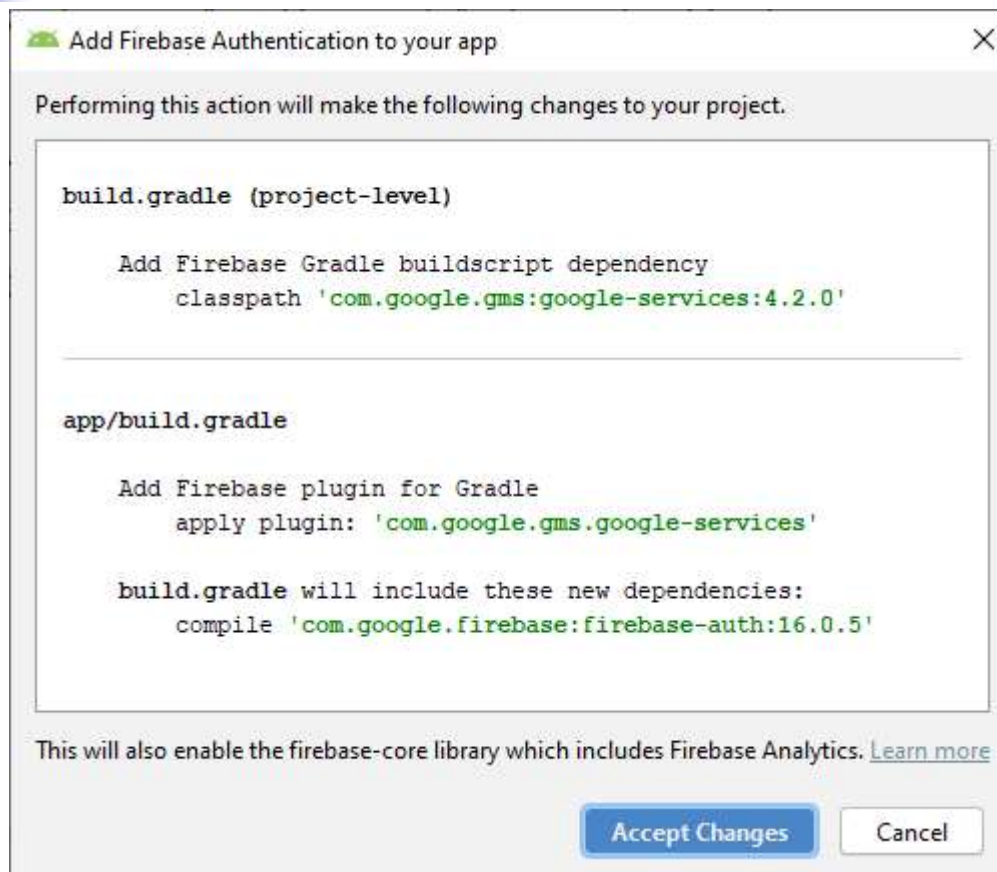
☐ Choose an existing Firebase or Google project

FBDemo	1 Android app(s) connected
kamaratinamape	11 Android app(s) connected

Country/region [What's this?](#)

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 anytime. [Learn more](#)

## 2) Add Firebase Auth to your app





# build.gradle (app)

```
dependencies {  
    // Firebase BoM ( https://firebase.google.com/docs/android/Learn-more#bom)  
    implementation platform('com.google.firebase:firebase-bom:26.1.1')  
  
    // Firebase Authentication (Java)  
    implementation 'com.google.firebase:firebase-auth'  
    // Firebase Authentication (Kotlin)  
    implementation 'com.google.firebase:firebase-auth-ktx'  
    // Google Sign In SDK (only required for Google Sign In)  
    implementation 'com.google.android.gms:play-services-auth:19.0.0'  
  
    // Firebase UI  
    // Used in FirebaseAuthUIActivity.  
    implementation 'com.firebaseui:firebase-ui-auth:7.1.1'  
  
    // Facebook Android SDK (only required for Facebook Login)  
    // Used in FacebookLoginActivity.  
    implementation 'com.facebook.android:facebook-login:4.42.0'  
    implementation 'androidx.browser:browser:1.3.0'  
}
```

```
plugins {  
    id 'com.android.application'  
    id 'kotlin-android'  
    id 'com.google.gms.google-services'  
}
```

Project:FirebaseDemo1



# build.gradle (Project)

```
// Top-level build file where you can add configuration options common to all  
sub-projects/modules.  
buildscript {  
    ext.kotlin_version = "1.3.72"  
    repositories {  
        google() ←  
        jcenter()  
    }  
    dependencies {  
        classpath "com.android.tools.build:gradle:4.1.0"  
        classpath 'com.google.gms:google-services:4.3.4' ←  
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"  
    }  
}  
allprojects {  
    repositories {  
        google() ←  
        jcenter()  
    }  
}
```



# Firebase Email Authentication

## sign-up new user

```
lateinit var mAuth: FirebaseAuth
. . .
mAuth.createUserWithEmailAndPassword(email, password)
    .addOnCompleteListener(this, object:
        OnCompleteListener<AuthResult> {
            override fun onComplete(task: Task<AuthResult>) {
                if (task.isSuccessful()) { // Sign in success, update
                    Log.d(TAG, "createUserWithEmail:success")
                    val user = mAuth.currentUser
                } else { // If sign in fails,
                    Log.w(TAG, "createUserWithEmail:failure",
                        task.getException())
                }
            }
        })
```



Authentication

iOS    

# Firebase Email Authentication

## sign-in an existing user

---

```
lateinit var mAuth: FirebaseAuth
...
mAuth.signInWithEmailAndPassword(email, password)
    .addOnCompleteListener(this) { task ->
        if (task.isSuccessful) { // Sign in success
            Log.d(TAG, "signInWithEmail:success")
            val user = mAuth.currentUser
        } else { // If sign in fails
            Log.w(TAG, "signInWithEmail:failure", task.exception)
        }
    }
}
```





Authentication

iOS C++

# Firebase Email Authentication

## get user info

```
val user = FirebaseAuth.getInstance().currentUser
if (user != null) { // Name, email address, and profile photo Url
    val name = user.displayName
    val email = user.email
    val photoUrl: Uri? = user.photoUrl
    val emailVerified = user.isEmailVerified
    val uid = user.uid
    infoTV.setText("$name, $email, $uid")
}
```

FirebaseDemo1

miki@sme.sk

qqqqqqq

SIGN IN

INFO

SIGNUP

null, miki@sme.sk, CWlvuMztroOnSvRxJZqxZmKGKNb2

Project:FirebaseDemo1



Authentication

iOS



C++



# Iná autentifikácia

Pozrite si [Facebook](#), resp. [Google](#) Login API

12:31

Firebase Authentication

**Java**

Run the Firebase Auth quickstart written in Java.

OPEN

12:31

Firebase Authentication

**Kotlin**

Run the Firebase

GoogleSignInActivity	Use a Google Sign In credential to authenticate with Firebase.
FacebookLoginActivity	Use a Facebook Login credential to authenticate with Firebase.
TwitterLoginActivity	Use a Twitter Login credential to authenticate with Firebase.
EmailPasswordActivity	Use an email and password to authenticate with Firebase.
PasswordlessActivity	Use only an email to authenticate with Firebase.
PhoneAuthActivity	Use a phone number to authenticate with Firebase.
AnonymousAuthActivity	Sign in anonymously and then later upgrade to a full Firebase Auth user.
FirebaseUIActivity	

Project:auth.zip

<https://github.com/firebase/quickstart-android>

# Firestore ako databáza

- noSQL databázy
- **Realtime Database** (efektívna pre mobilné app, synchronizácia)
  - ukladanie a synchronizácia v reálnom čase so všetkými pripojenými klientami
  - všetky dáta sú jeden veľký json dátový strom (JSON Tree)
  - existuje dávno, stabilná, **regionálne** má veľmi slušnú latenciu
- **Cloud Firestore** (novinka)
  - Realtime updates, powerful queries, automatic scaling
  - dáta sú v kolekciách, hierarchicky organizované, subkolekcie, ...
  - novinka, beta r.2018, skalabilita, prepojenie cez viaceré dátové centrá



Realtime Database

iOS



Cloud Firestore

iOS

Obe podporujú offline support pre mobilných klientov (Android, iOS, web)  
- zmeny počas off-line sa ukladajú do cache a synchronizujú, keď on-line

# Realtime Database

## nosql databáza - rules

- Default, no access, only FB console

```
// These rules don't allow anyone read or write access to your database
{
  "rules": {
    ".read": false,
    ".write": false
  }
}
```

- Public, anyone can...

```
// These rules give anyone, even people who are not users of your app,
// read and write access to your database
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

- FB/Google/Git Authenticated only

```
{
  "rules": {
    "users": {
      "$uid": {
        ".read": "$uid === auth.uid",
        ".write": "$uid === auth.uid"
      }
    }
  }
}
```

<https://kamaratinamape.firebaseio.com/>

★ Default security rules require users to be authenticated

kamaratinamape

- 4eWsnHhdRPVgUAtV200q7V1gNOI3: "treti :
  - pikatchus
    - address: "treti status"
    - lati: 48
    - longi: 17
    - name: "treti"
    - time: 1480539003061
  - statuses
    - druhy: "druhy status"
    - prvy: "prvy status"
    - treti: "treti status"

Realtime Database

DATA RULES USAGE BACKUPS

## Firebase Email Authentication

★ Default security rules require users to be authenticated

```
1 {
2   "rules": {
3     ".read": "auth != null",
4     ".write": "auth != null"
5   }
}
```



Realtime Database

iOS </> C++

# Realtime Database

zápis dát

```
databaseReference = FirebaseDatabase  
    .getInstance()  
    .reference
```

→ **object** `databaseReference.child("pikatchus")`  
    `.setValue(Pokemon(name, address, lati, longi))`

```
class Pokemon : Serializable
```

```
globalState == mutableMapOf<String,String>()  
globalState[name]=status // prvy -> prvy status
```

→ **kolekcia** `databaseReference.child("statuses").setValue(globalState)`

```
val user = firebaseAuth.currentUser  
if (user != null) { // ak je user nalogovaný  
→ databaseReference.child(user.uid).setValue(status)  
uid  
}
```

<https://kamaratinamape.firebaseio.com/>

★ Default security rules require users to be authenticated

```
kamaratinamape  
├── 4eWsnHhdRPVgUAtV200q7V1gNOI3 "treti status"  
└── pikatchus  
    ├── address: "treti status"  
    ├── lati: 48  
    ├── longi: 17  
    ├── name: "treti"  
    ├── time: 1480539003061  
    └── statuses  
        ├── druhy: "druhy status"  
        ├── prvy: "prvy status"  
        └── tretí: "tretí status"
```

Project:FirebaseDemo3.zip

# Realtime Database

čítanie dát - synchronizácia

Pri akejkol'vek zmene sa zavolá listener

→ po zápise

**databaseReference.addValueEventListener (**

```
object: ValueEventListener() {
    override fun onDataChange(dataSnapshot: DataSnapshot) {
        for (child in dataSnapshot.children) { % toto sú zmeny
            val o = child.value % (child.key, child.value)
            val str = "changed " + child.key + " is: "
            val o = child.value % zmenil sa (key,o=value)
            → if (o is Pokemon) % zmenil sa objekt
                statusMemo.append("Pokemon has ")
            Log.d(TAG, child.key)
            editTextMemo.append(
                "changed ${child.key} is: ${o.toString()}\n")
        }
    }
}
```

<https://kamaratinamape.firebaseio.com/>

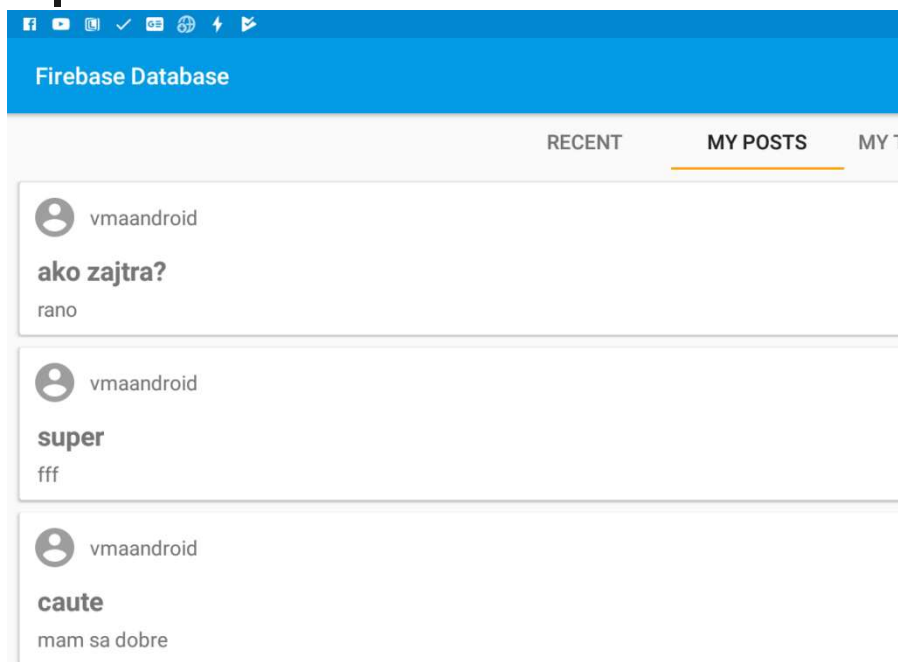
★ Default security rules require users to be authenticated

```
kamaratinamape
├── 4eWsnHhdRPVgUAtV200q7V1gNOI3: "treti status"
├── pikatchus
│   ├── address: "treti status"
│   ├── lati: 48
│   ├── longi: 17
│   ├── name: "treti"
│   └── time: 1480539003061
└── statuses
    ├── druhy: "druhy status"
    ├── prvy: "prvy status"
    └── tretí: "tretí status"
```



# Firestore DB

vyskúšajte si hotovú appku



pre začiatok odporúčam  
skúsiť aplikácie z balíka  
GITHUB:  
[quickstart-android-master](#)



Project:Database.zip



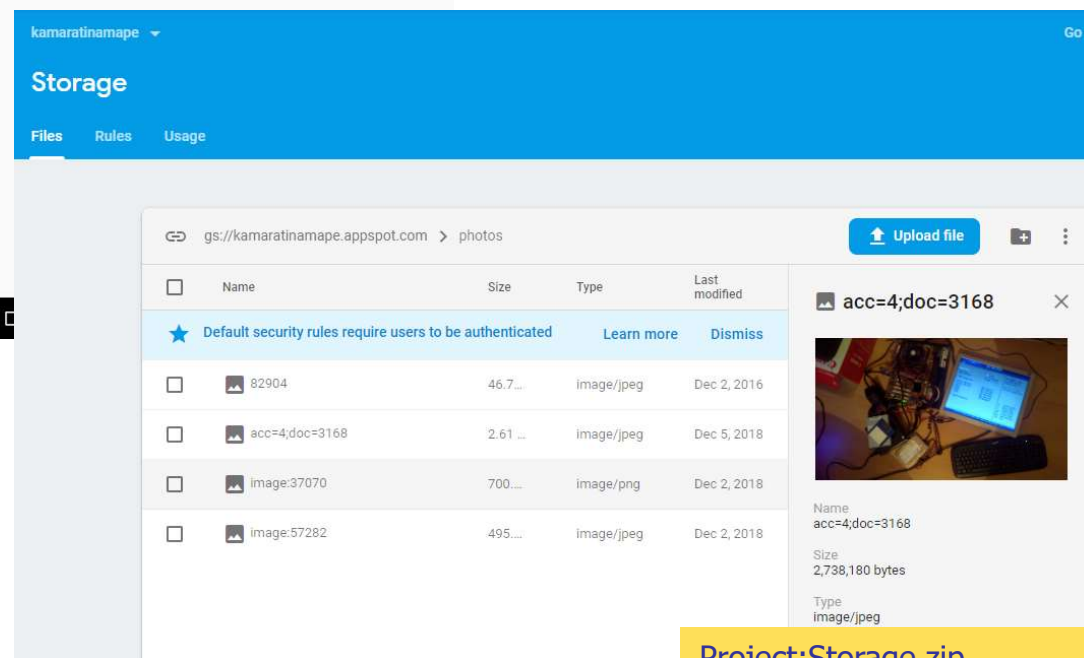
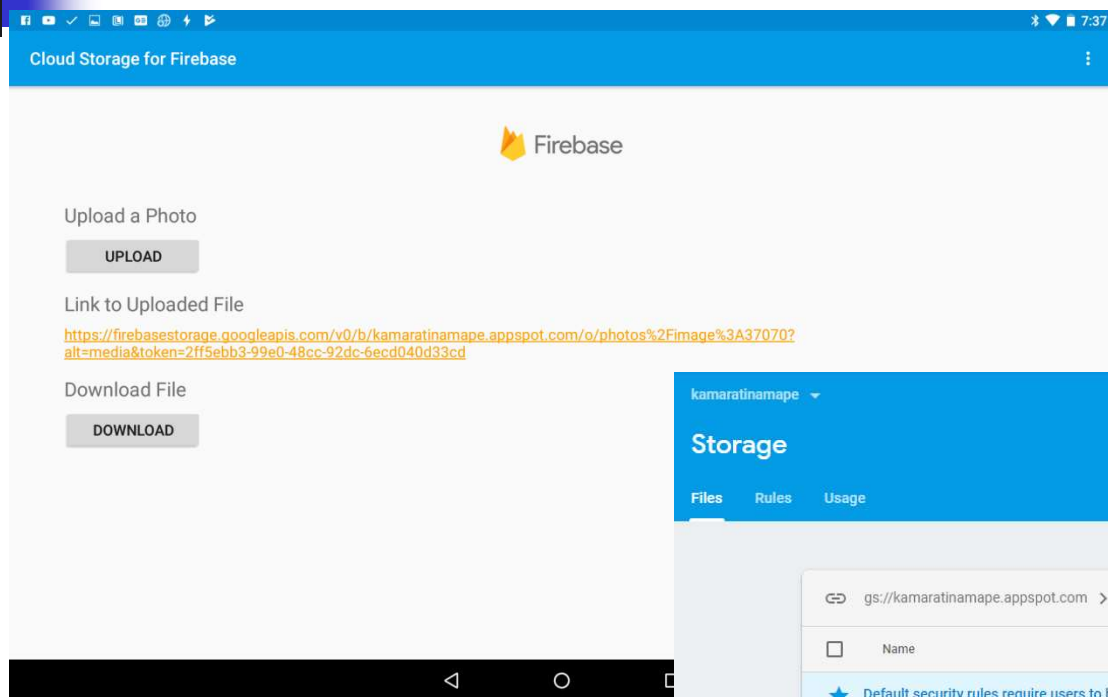


Cloud Firestore

iOS

# Firestore Storage

vyskúšajte si hotovú appku



pre začiatok odporúčam  
skúsiť aplikácie z balíka

GITHUB:

[quickstart-android-master](https://github.com/firebase/quickstart-android-master)

Project:Storage.zip





# Firebase Cloud Messaging

(vyskúšajte si)

- Firebase Messaging – push notifikácie
- cross-platform (iOS, Android, ...)
- payload of up to 4kB to client app.

pre začiatok odporúčam  
skúsiť aplikácie z balíka

GITHUB:

[quickstart-android-master](https://github.com/firebase/quickstart-android-master)

- Notification messages – when app is in the **background**

```
{"message":{  
  "token":"chLzRZ59Svk:APA91bGEy41ulMs3qQnThxYL6VWJAOu61pIHWkGTUEHQe4rWlyWL9yutLHxiwmgYdstis7T54I68yKhWZj95TnKXjUynd4rt2oLQ1gPAaIa249g2-h4MKSg7Xkgie8uCVPx8sbB_itLr",  
  "notification":{  
    "title":"Notification Test",  
    "body":"test"  
  }  
}}
```

- Data messages – key/value pairs received in a callback function.

```
{"message":{  
  "token":"chLzRZ59Svk:APA91bGEy41ulMs3qQnThxYL6VWJAOu61pIHWkGTUEHQe4rWlyWL9yutLHxiwmgYdstis7T54I68yKhWZj95TnKXjUynd4rt2oLQ1gPAaIa249g2-h4MKSg7Xkgie8uCVPx8sbB_itLr",  
  "data":{  
    "Nick" : "Peter",  
    "body" : "teacher",  
    "Room" : "I-18"  
  }  
}}
```

# Firestore Cloud Messaging

(vyskúšajte si)

Device Token

InstanceID Token:

chLzRZ59Svk:APA91bGEy41u1Ms3qQnThxYL6VWJAOu61pIHWkGTUEHQe4rWlyWL9yutLHxiwmgYdstis7T54I68yKh  
wZj95TnKXjUynd4rt2oLQ1gPAaIa249g2-h4MKSg7Xkgie8uCVPx8sbB\_itLr

Sending a test message from Firestore Console (app is in background!)

1 Notification

Notification title (optional) ⓘ

Enter optional title

Test on device

You can test this campaign by entering or selecting the [FCM registration tokens](#) ⓘ of your development device below.

Add an FCM registration token

Recently Used ⓘ

☒ chLzRZ59Svk:APA91bGEy41u1Ms3qQnThxYL6VWJAOu61pIHWkGTUEHQe4rWlyWL9yutLHxiwmgYdstis7T54I68yKh

Cancel Test

pre začiatok odporúčam  
skúsiť aplikácie z balíka  
GITHUB:  
[quickstart-android-master](#)

Project:messaging.zip



# Toto nepôjde na emulátore

---

- treba mu nainštalovať Google Play Services
- <https://stackoverflow.com/questions/46464356/firebase-message-not-received-on-emulator>
- ako:
- [https://medium.com/@dai\\_shi/installing-google-play-services-on-an-android-studio-emulator-fffceb2c28a1](https://medium.com/@dai_shi/installing-google-play-services-on-an-android-studio-emulator-fffceb2c28a1)