Firebase Storage

1. AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.AppX0eb">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name=
                        "android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

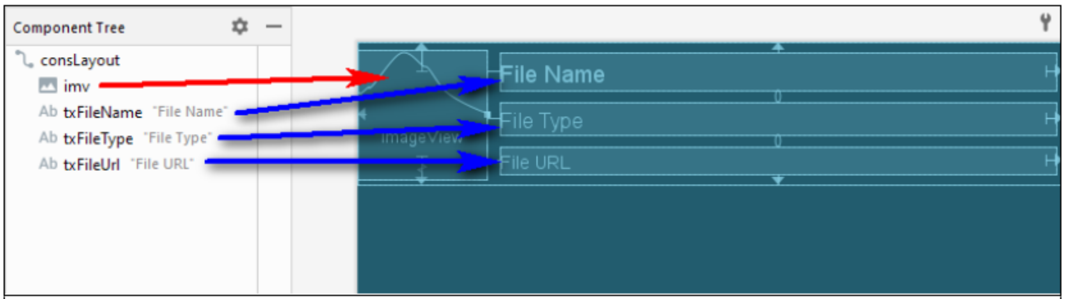
2. build.gradle (Project)

```
buildscript {
    dependencies {
        classpath 'com.google.gms:google-services:4.4.0'
    }
}// Top-level build file where you can add configuration options common to
all sub-projects/modules.
plugins {
    id 'com.android.application' version '8.1.1' apply false
    id 'org.jetbrains.kotlin.android' version '1.9.0' apply false
}
```

3. build.gradle (module)

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

android {
    namespace 'polinema.mi.appx0eb'
    compileSdk 34

    defaultConfig {
        applicationId "polinema.mi.appx0eb"
        minSdk 24
        targetSdk 34
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner
                    "androidx.test.runner.AndroidJUnitRunner"
    }
    buildFeatures {
        viewBinding true
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile(
                    'proguard-android-optimize.txt'),
                    'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
}

dependencies {

    implementation 'androidx.core:core-ktx:1.12.0'
    implementation 'androidx.appcompat:appcompat:1.6.1'
    implementation 'com.google.android.material:material:1.9.0'
    implementation
          'androidx.constraintlayout:constraintlayout:2.1.4'
    implementation
          'com.google.firebase:firebase-storage-ktx:20.2.1'
    implementation
          'com.google.firebase:firebase-firestore-ktx:24.8.1'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation
          'androidx.test.espresso:espresso-core:3.5.1'
    implementation 'com.squareup.picasso:picasso:2.8'
}
```

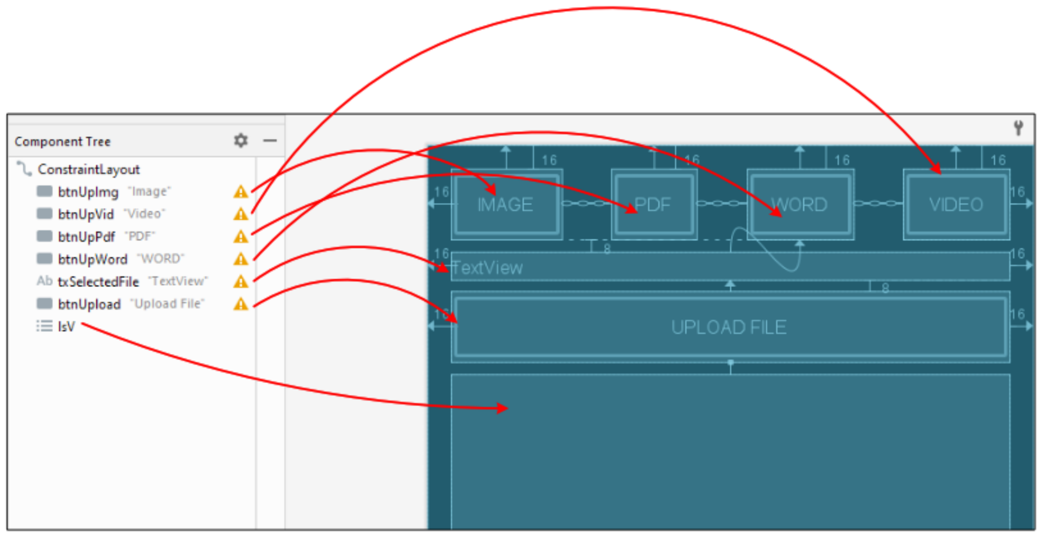
4. row_data.xml



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/consLayout"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical">

   <ImageView
      android:id="@+id/imv"
      android:layout_width="75dp"
      android:layout_height="75dp"
      android:layout_marginTop="4dp"
      android:layout_marginBottom="4dp"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent"
      app:srcCompat="@android:color/holo_purple" />

   <TextView
      android:id="@+id/txFileName"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_marginStart="8dp"
      android:layout_marginEnd="4dp"
      android:text="File Name"
      android:textAppearance=
                 "@style/Base.TextAppearance.AppCompat.Title"
      android:textSize="16sp"
      android:textStyle="bold"
      app:layout_constraintBottom_toTopOf="@+id/txFileType"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintHorizontal_bias="0.5"
      app:layout_constraintStart_toEndOf="@+id/imv"
      app:layout_constraintTop_toTopOf="parent" />

   <TextView
      android:id="@+id/txFileType"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
```

```
        android:layout_marginStart="8dp"
        android:layout_marginEnd="4dp"
        android:text="File Type"
        android:textStyle="bold"
        app:layout_constraintBottom_toTopOf="@+id/txFileUrl"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/imv"
        app:layout_constraintTop_toBottomOf="@+id/txFileName" />

    <TextView
        android:id="@+id/txFileUrl"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="4dp"
        android:text="File URL"
        android:textSize="12sp"
        android:textStyle="italic"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/imv"
        app:layout_constraintTop_toBottomOf="@+id/txFileType" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

5. activity_main.xml



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/btnUpImg"
        style="@style/Base.Widget.AppCompat.Button.Small"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:text="Image"
        app:layout_constraintEnd_toStartOf="@+id/btnUpPdf"
        app:layout_constraintHorizontal_chainStyle="spread_inside"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/btnUpVid"
        style="@style/Base.Widget.AppCompat.Button.Small"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="16dp"
        android:text="Video"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/btnUpWord"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/btnUpPdf"
        style="@style/Base.Widget.AppCompat.Button.Small"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="PDF"
        app:layout_constraintEnd_toStartOf="@+id/btnUpWord"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/btnUpImg"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/btnUpWord"
        style="@style/Base.Widget.AppCompat.Button.Small"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="WORD"
        app:layout_constraintEnd_toStartOf="@+id/btnUpVid"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/btnUpPdf"
        app:layout_constraintTop_toTopOf="parent" />
    <TextView
        android:id="@+id/txSelectedFile"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="16dp"
        android:text="TextView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnUpWord" />

    <Button
        android:id="@+id/btnUpload"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="16dp"
        android:text="Upload File"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf=
            "@+id/txSelectedFile" />
    <ListView
        android:id="@+id/lsV"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="16dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnUpload" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

6. CustomAdapter.kt (class)

```
package polinema.mi.appx0eb

import android.content.Context
import android.content.Intent
import android.net.Uri
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.BaseAdapter
import android.widget.ImageView
import android.widget.TextView
import com.squareup.picasso.Picasso

class CustomAdapter(val context : Context,
                arrayList : ArrayList<HashMap<String, Any>>) :
BaseAdapter() {
   val F_NAME = "file_name"
   val F_TYPE = "file_type"
   val F_URL = "file_url"
   val list = arrayList
   var uri = Uri.EMPTY

   inner class ViewHolder(){
      var txFileName : TextView? = null
      var txFileType : TextView? = null
      var txFileUrl : TextView? = null
      var imv : ImageView? = null
   }

   override fun getView(position: Int, convertView: View?, parent:
          ViewGroup?): View {
      var holder = ViewHolder()
      var view = convertView
      if(convertView == null){
         var inflater =
          context.getSystemService(Context.LAYOUT_INFLATER_SERVICE)
          as LayoutInflater
         view = inflater.inflate(R.layout.row_data,null,true)

         holder.txFileName =
                    view!!.findViewById(R.id.txFileName) as TextView
         holder.txFileType =
                    view!!.findViewById(R.id.txFileType) as TextView
         holder.txFileUrl = view!!.findViewById(R.id.txFileUrl) as TextView
         holder.imv = view!!.findViewById(R.id.imv) as ImageView
         view.tag = holder
      }else{
         holder = view!!.tag as ViewHolder
      }

      var fileType = list.get(position).get(F_TYPE).toString()
      uri = Uri.parse(list.get(position).get(F_URL).toString())

      holder.txFileName!!.setText(list.get(position).get(F_NAME).toString())
      holder.txFileType!!.setText(fileType)
      holder.txFileUrl!!.setText(uri.toString())
      holder.txFileUrl!!.setOnClickListener {
         val intent = Intent(Intent.ACTION_VIEW).setData(
            Uri.parse(holder.txFileUrl!!.text.toString())
         )
         context.startActivity(intent)
      }
```

```
        when(fileType){
            "PDF" ->  {holder.imv!!.setImageResource(
                                android.R.drawable.ic_dialog_dialer)}
            "Docx" -> {holder.imv!!.setImageResource(
                                android.R.drawable.ic_menu_edit)}
            "Video" ->  {holder.imv!!.setImageResource(
                                android.R.drawable.ic_media_play)}
            "Image" ->  {Picasso.get().load(uri).into(holder.imv)}
        }

        return view!!
    }

    override fun getItem(position: Int): Any {
        return list.get(position)    }

    override fun getItemId(position: Int): Long {
        return 0    }

    override fun getCount(): Int {
        return  list.size    }
}
```

7. MainActivity.kt (class)

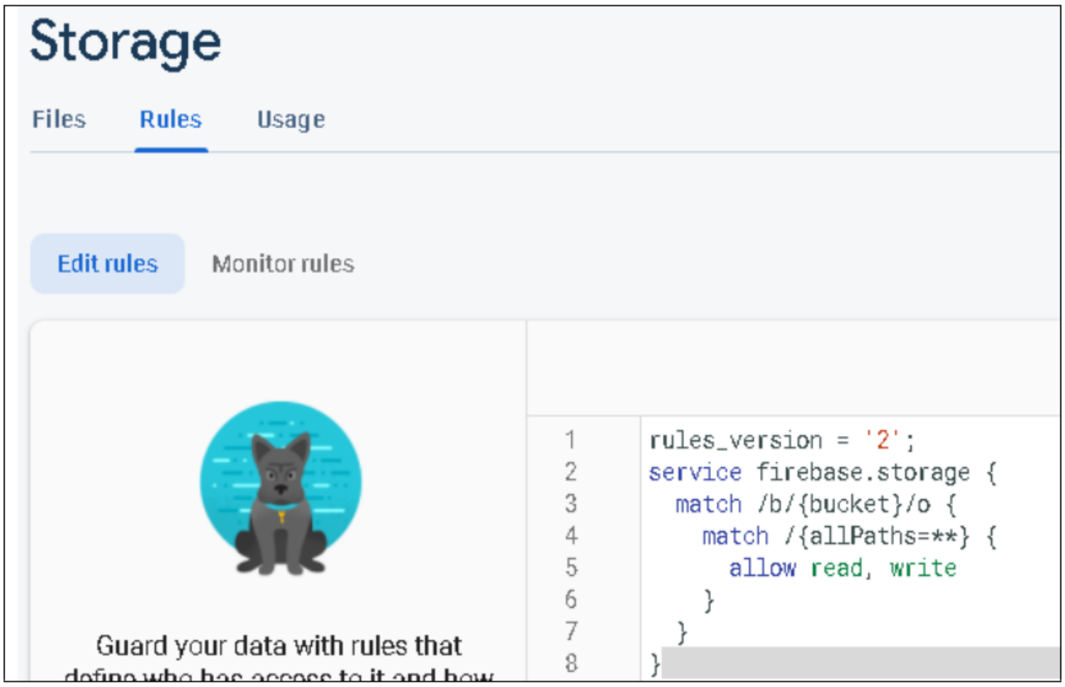
```
package polinema.mi.appx0eb

import android.app.Activity
import android.content.Intent
import android.net.Uri
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.provider.OpenableColumns
import android.util.Log
import android.view.View
import android.widget.Toast
import com.google.firebase.firestore.CollectionReference
import com.google.firebase.firestore.FirebaseFirestore
import com.google.firebase.ktx.Firebase
import com.google.firebase.storage.FirebaseStorage
import com.google.firebase.storage.ktx.storage
import polinema.mi.appx0eb.databinding.ActivityMainBinding
import kotlin.collections.ArrayList
import kotlin.collections.HashMap

class MainActivity : AppCompatActivity(), View.OnClickListener {
    lateinit var storage : FirebaseStorage
    lateinit var db : CollectionReference
    lateinit var alFile : ArrayList<HashMap<String,Any>>
    lateinit var adapter : CustomAdapter
    lateinit var uri : Uri
    lateinit var b: ActivityMainBinding
```

```
val F_NAME = "file_name"
val F_TYPE = "file_type"
val F_URL = "file_url"
val RC_OK = 100
var fileType =""
var fileName =""

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    b = ActivityMainBinding.inflate(layoutInflater)

    setContentView(b.root)

    b.btnUpPdf.setOnClickListener(this)
    b.btnUpImg.setOnClickListener(this)
    b.btnUpVid.setOnClickListener(this)
    b.btnUpWord.setOnClickListener(this)
    b.btnUpload.setOnClickListener(this)

    storage = Firebase.storage
    alFile = ArrayList()
    uri = Uri.EMPTY
}

override fun onStart() {
    super.onStart()
    db = FirebaseFirestore.getInstance().collection("files")
    db.addSnapshotListener {
                 querySnapshot, firebaseFirestoreException ->
       if(firebaseFirestoreException!=null){
           firebaseFirestoreException.message?.let {
                                        Log.e("firestore :", it) } //new update
       }
       showData()
    }
}

fun showData(){
    db.get().addOnSuccessListener {result ->
       alFile.clear()
       for(doc in result){
           val hm = HashMap<String,Any>()
           hm.put(F_NAME, doc.get(F_NAME).toString())
           hm.put(F_TYPE, doc.get(F_TYPE).toString())
           hm.put(F_URL, doc.get(F_URL).toString())
           alFile.add(hm)
       }
       adapter = CustomAdapter(this,alFile)
       b.lsV.adapter = adapter
    }.addOnFailureListener {
    }
}
```

```
override fun onActivityResult(
                   requestCode: Int, resultCode: Int, intent: Intent?) {
   super.onActivityResult(requestCode, resultCode, intent)
   if((resultCode == Activity.RESULT_OK) &&
                                         (requestCode == RC_OK)){
      if(intent != null){
         uri = intent.data!!
         val cursor = contentResolver.query(uri,null,
            null, null, null)
         val COL_FILE_NAME =
             cursor!!.getColumnIndex(OpenableColumns.DISPLAY_NAME)
         cursor!!.moveToFirst()
         fileName = cursor.getString(COL_FILE_NAME)
         b.txSelectedFile.setText(uri.toString())
      }
   }
}

override fun onClick(v: View?) {
   val intent = Intent()
   intent.action = Intent.ACTION_GET_CONTENT
   when(v?.id) {
      R.id.btnUpWord -> {
         fileType = "Docx"
         intent.setType( //MIME
            "application/vnd.openxmlformats-
                           officedocument.wordprocessingml.document"
         )
      }
      R.id.btnUpVid -> {
         fileType = "Video"
         intent.setType("video/*")
      }
      R.id.btnUpImg -> {
         fileType = "Image"
         intent.setType("image/*")
      }
      R.id.btnUpPdf -> {
         fileType = "PDF"
         intent.setType("application/pdf")
      }
      R.id.btnUpload-> {
         if(uri != null){
            val fileRef = storage.reference.child(fileName)
            val uploadTask = fileRef.putFile(uri)
            uploadTask.continueWithTask { task ->
              if(!task.isSuccessful){
                  task.exception?.let{
                     throw it
                  }
              }
```

```
                    fileRef.downloadUrl
                }.addOnCompleteListener {
                      val hm = HashMap<String, Any>()
                       hm.put(F_NAME, fileName)
                       hm.put(F_TYPE, fileType)
                       hm.put(F_URL, it.result.toString())
                       db.document(fileName).set(hm)
                                                 .addOnSuccessListener {
                         Toast.makeText(
                             this, "File successfully uploaded",
                             Toast.LENGTH_SHORT
                          ).show()
                       }
                }
              }
            }
        }
        if(v?.id != R.id.btnUpload)
            startActivityForResult(intent, RC_OK)
    }
}
```

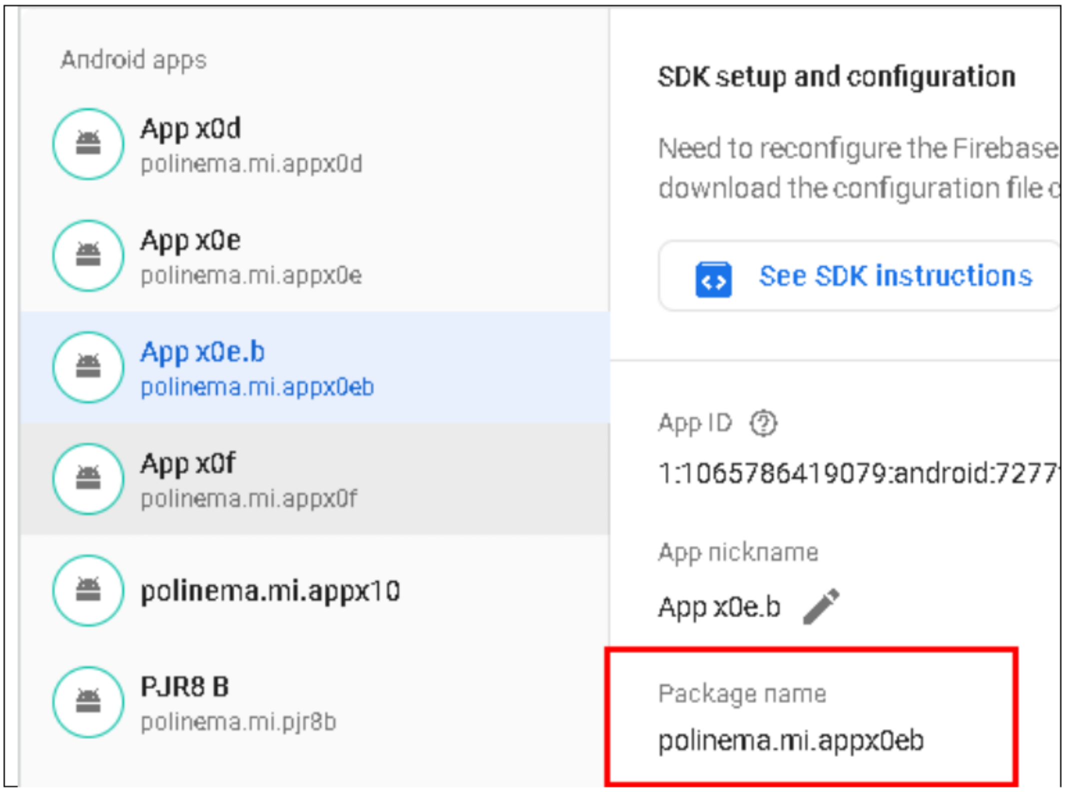
8. Firebase Storage Rules



```
rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write
    }
  }
}
```

Guard your data with rules that define who has access to it and how

9. Firebase Project Setting



Android apps

App x0d
polinema.mi.appx0d

App x0e
polinema.mi.appx0e

App x0e.b
polinema.mi.appx0eb

App x0f
polinema.mi.appx0f

polinema.mi.appx10

PJR8 B
polinema.mi.pjr8b

SDK setup and configuration

Need to reconfigure the Firebase
download the configuration file c

See SDK instructions

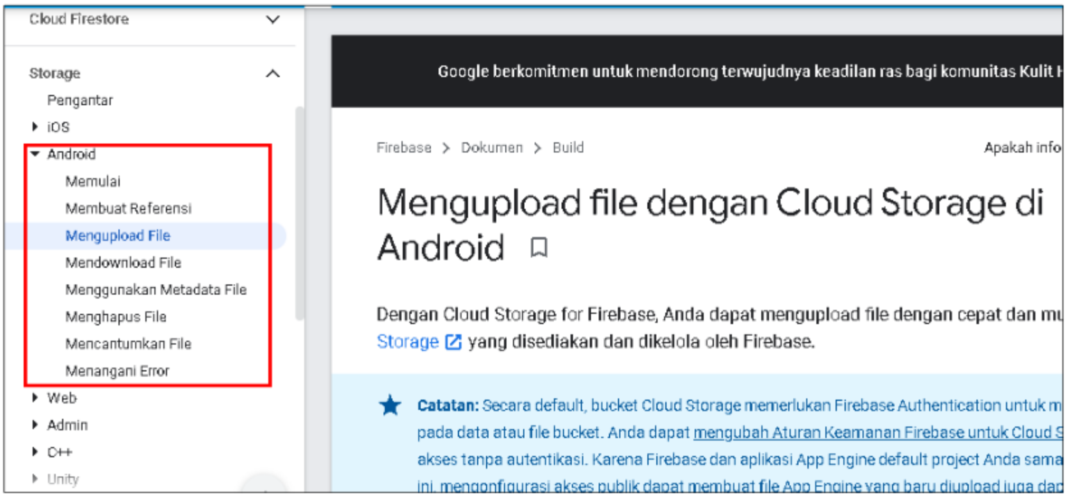
App ID
1:1065786419079:android:7277

App nickname
App x0e.b

Package name
polinema.mi.appx0eb

Referensi untuk tugas :

https://firebase.google.com/docs/storage/android/upload-files?hl=id



Google berkomitmen untuk mendorong terwujudnya keadilan ras bagi komunitas Kulit H

Firebase > Dokumen > Build

Apakah info

Mengupload file dengan Cloud Storage di Android

Dengan Cloud Storage for Firebase, Anda dapat mengupload file dengan cepat dan mu
Storage yang disediakan dan dikelola oleh Firebase.

Catatan: Secara default, bucket Cloud Storage memerlukan Firebase Authentication untuk m
pada data atau file bucket. Anda dapat mengubah Aturan Keamanan Firebase untuk Cloud S
akses tanpa autentikasi. Karena Firebase dan aplikasi App Engine default project Anda sama
ini, mengonfigurasi akses publik dapat membuat file App Engine yang baru diupload juga dap