

# Aplicación Móvil Android devLivery

**Facilitador:** Ing. Andrés Vasquez.  
[andres.vasquez.agramont@gmail.com](mailto:andres.vasquez.agramont@gmail.com)  
<https://github.com/andres-vasquez>

**Objetivo:** Explicar de manera práctica mediante un caso de uso la utilización de Firebase en un proyecto de base de datos en tiempo real.

## Descripción de la App

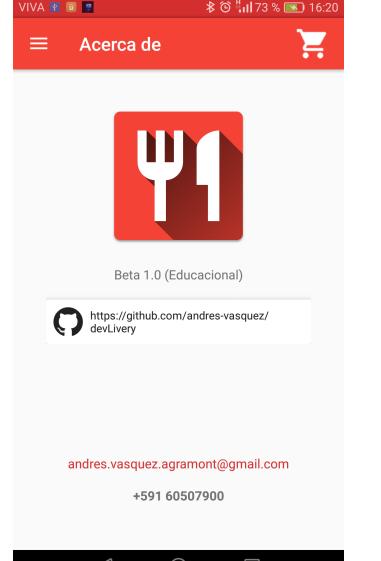
**Login**

Se utilizó la librería FirebaseUI que integra la funcionalidad de Login por:

- Email
- Gmail

app/build.gradle

```
compile "com.firebaseioui:firebase-ui-auth:0.4.1"
compile "com.firebaseioui:firebase-ui-database: 0.4.1"
```

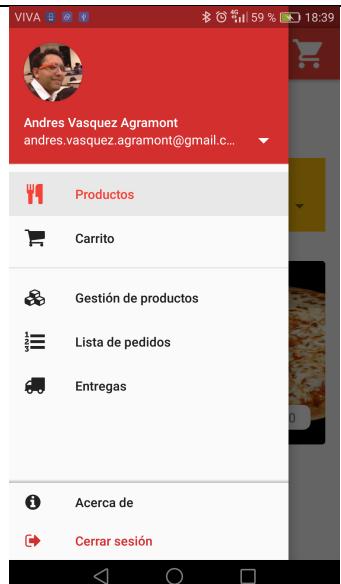


The screenshot shows the home screen of the devLivery app. At the top is a red header bar with the text "Acerca de" and a shopping cart icon. Below the header is a large red square containing a white fork and knife icon. To the right of the icon, the text "Beta 1.0 (Educacional)" is visible. At the bottom of the screen, there is contact information: an email address "andres.vasquez.agramont@gmail.com" and a phone number "+591 60507900". The bottom of the screen features a black navigation bar with three icons: a triangle pointing left, a circle, and a square.

## Menú de la App

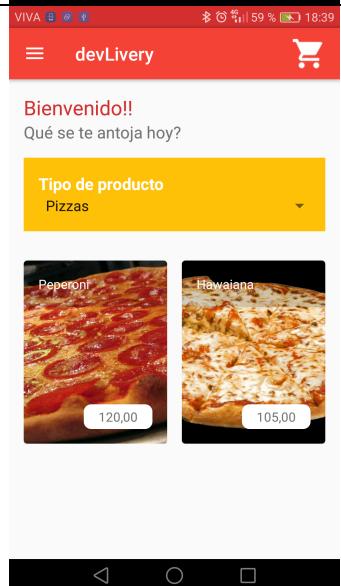
Se utilizaron las librerías: MaterialDrawer y Iconics para crear el menú lateral

```
app/build.gradle
compile('com.mikepenz:materialdrawer:5.3.6@aar') {
    transitive = true
}
compile 'com.mikepenz:iconics-core:2.7.1@aar'
compile 'com.mikepenz:fontawesome-typeface:4.6.0.2@aar'
```



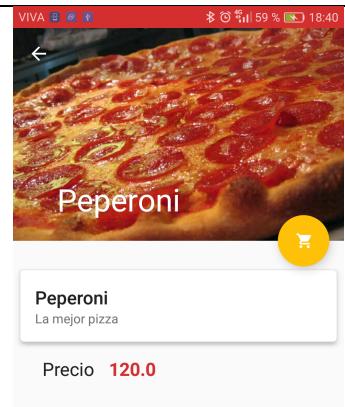
## Usuarios - Visualización de productos

Se utilizaron componentes propios de Android como spinner, CardViews y RecyClerView en modo GridLayout para la visualizar los productos disponibles.



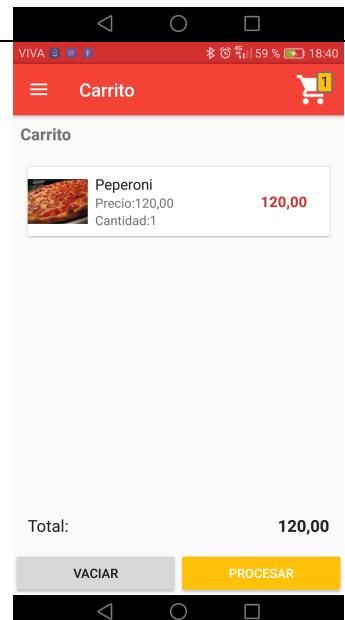
### Usuarios - Detalle de producto

Se utilizaron componentes propios de Android como CoordinatorLayout, FloatingActionButton y CollapsinToolbarLayout para obtener la vista deseada.



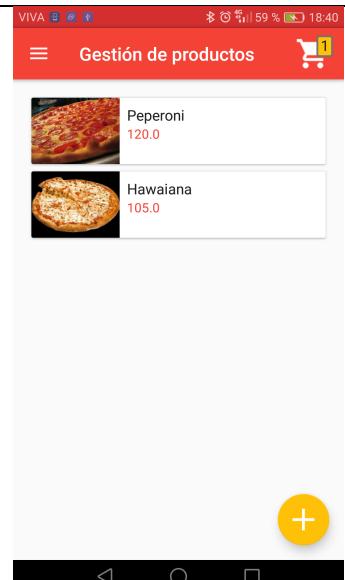
### Usuarios - Carrito de compra

Se utilizaron componentes propios de Android como CardViews y RecyClerView en modo LinearLayout para la visualizar el carrito de compra.



## Admin – Gestión de productos

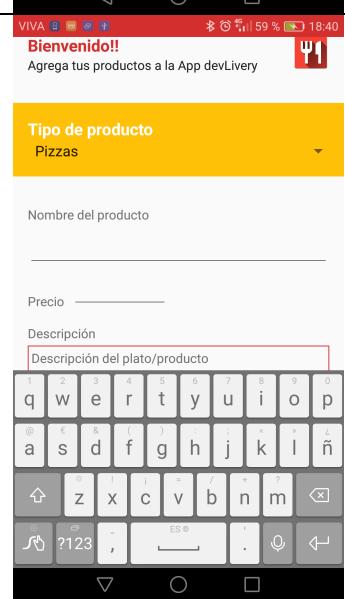
Se utilizaron componentes propios de Android como CardViews y RecyClerView en modo LinearLayout para la visualizar los productos disponibles.



## Admin – Gestión de productos

Se utilizaron componentes nativos de Android para desarrollar un formulario de registro de productos con los siguientes campos:

- Categoría
- Nombre del producto.
- Precio del producto.
- Descripción



## Admin - Gestión de productos

Activity donde se adjuntará una fotografía del producto deseado.

A partir de Android 6.0 se requiere la solicitud de permisos en tiempo de ejecución para lo cual se utilizará la librería easy-permissions.

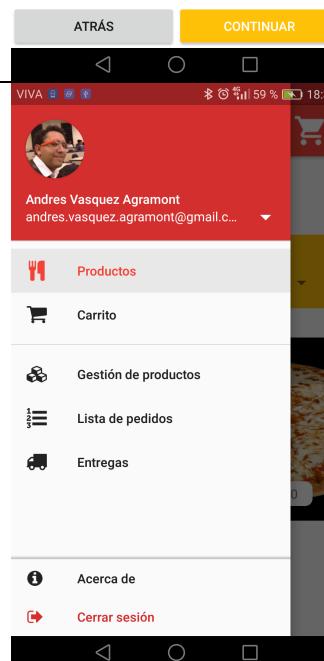
app/build.gradle  
compile '[pub.devrel:easypermissions:0.1.7](#)'



## Admin - Gestión de pedidos

Se utilizaron componentes propios de Android como CardViews y RecyClerView en modo LinearLayout para la visualizar los pedidos con estado PREPARACION.

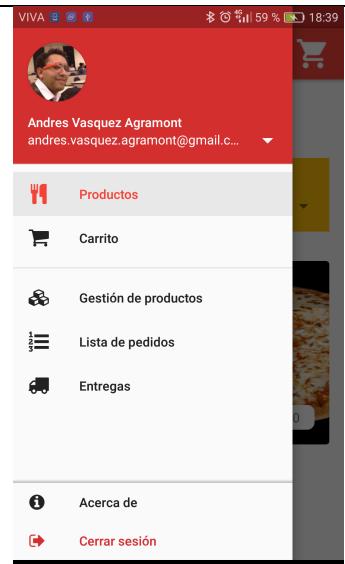
Cuando se hace click despliega los datos del pedido.



## Admin – Pedidos para delivery

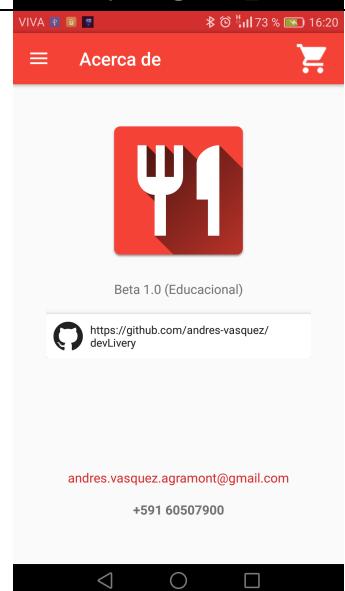
Se utilizaron componentes propios de Android como CardViews y RecyClerView en modo LinearLayout para la visualizar los pedidos con estado DELIVERY.

Cuando se hace click despliega abre la aplicación Google Maps con la información geográfica del pedido.



## About

Muestra detalles del creador de la aplicación.



## Contenido del taller:

1. Integración Android - Firebase
2. Crear producto.
  - a. Adjuntar imagen a Firebase
  - b. Enviar datos
3. Listar productos
4. Enviar pedidos
5. Listar pedidos por filtros:
  - a. En preparación
  - b. En delivery

## **1. Integración Android – Firebase.**

Ingrese a la consola de Firebase:  
<https://console.firebaseio.google.com/>

## Crear proyecto.

## Crear proyecto

Nombre del proyecto

País/Región ②

▼

De forma predeterminada, los datos de Firebase Analytics mejoran otras funciones de Firebase y productos de Google. En la configuración puedes controlar cómo se comparten dichos datos en cualquier momento. [Más información](#)

CANCELAR **CREAR PROYECTO**

## Añade Firebase a tu aplicación de Android.

Las aplicaciones que se conectan con servicios de Google requieren un certificado para su integración. Para obtenerlo sigue el siguiente link:

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

To get the debug certificate fingerprint:

MAC/LINUX	WINDOWS
-----------	---------

keytool -exportcert -list -v \ -alias androiddebugkey -keystore ~/.android/debug.keystore
--

La ejecución de dicho comando generará un resultado similar a este:

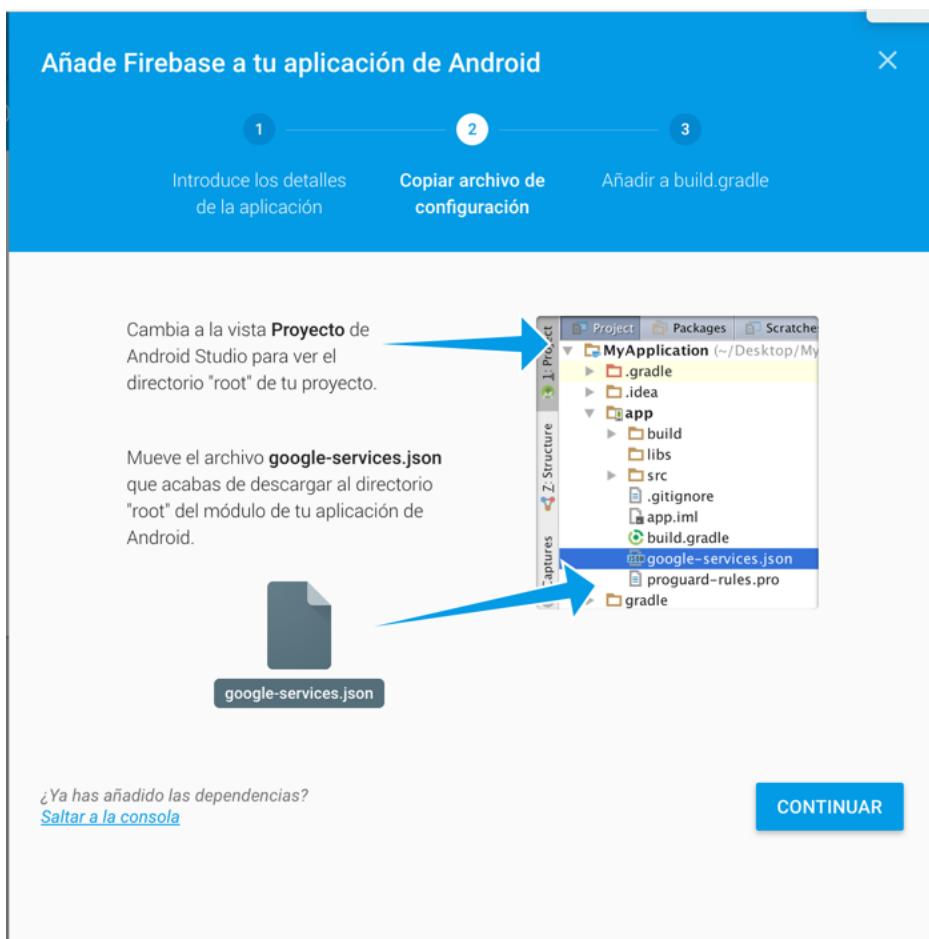
```
Alias name: androiddebugkey
Creation date: 04-01-2015
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=Android Debug, O=Android, C=US
Issuer: CN=Android Debug, O=Android, C=US
Serial number: 38214512
Valid from: Sun Jan 04 00:07:50 BOT 2015 until: Tue Dec 27 00:07:50 BOT 2044
Certificate fingerprints:
    MD5: 50:60:C0:12:BE:18:0E:DE:F8:0E:5B:F2:23:30:13:F8
    SHA1: FE:5A:AF:E7:2B:33:28:DE:93:8E:7A:B3:83:4E:A9:3D:CA:C2:5D:5B:91:CC:15:9A:BF:0E:E4:D5:05:4B:5D:FB
    Signature algorithm name: SHA256withRSA
    Version: 3

Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 63 BC FA C7 37 6B B9 97 5F E1 C2 19 71 8E 1B 3C c...7K...=...q..<
0010: 9F 43 6C E1 .Cl.
]
]

Andress-MacBook-Pro:~ andresvasquez$
```

Se debe copiar lo que está en el campo SHA1 en la consola de Firebase.

Una vez agragado el anterior se descargará un archivo con las credenciales de Firebase el cual debe ser copiado dentro de app desde la perspectiva project.



Por último se deben adicionar las librerías y classpath para configuración del entorno. Android – Firebase.

Añade Firebase a tu aplicación de Android X

1 Introduce los detalles de la aplicación    2 Copiar archivo de configuración    3 Añadir a build.gradle

El complemento de los servicios de Google para [Gradle](#) carga el archivo `google-services.json` que acabas de descargar. Para poder usar el complemento, debes modificar los archivos `build.gradle`.

1. **build.gradle de proyecto** (`<project>/build.gradle`):

```
buildscript {  
    dependencies {  
        // Add this line  
        classpath 'com.google.gms:google-services:3.0.0'  
    }  
}
```

2. **build.gradle de aplicación** (`<project>/<app-module>/build.gradle`):

```
...  
// Add to the bottom of the file  
apply plugin: 'com.google.gms.google-services'  
incluye Firebase Analytics de forma predeterminada ?
```

3. Por último, pulsa **Sincronizar ahora** en la barra que aparezca en el entorno IDE (entorno de desarrollo integrado):

Gradle files have changed since [Sync now](#)

Una vez creado el proyecto iniciar la perspectiva de Autenticación y habilitar las opciones Email y Gmail.

Authentication

USUARIOS    MÉTODO DE INICIO DE SESIÓN    PLANTILLAS DE CORREO ELECTRÓNICO

CONFIGURACIÓN WEB ?

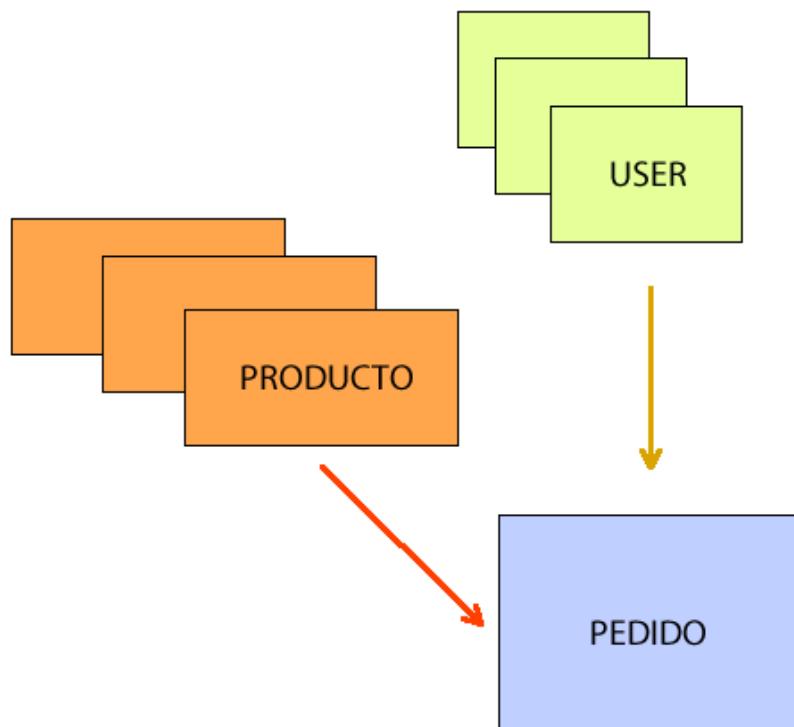
Proveedores de inicio de sesión

Proveedor	Estado
Correo electrónico/contraseña	Habilitado
Google	Habilitado
Facebook	Inhabilitado
Twitter	Inhabilitado
Github	Inhabilitado
Anónimo	Inhabilitado

## 2. Crear producto.

Antes de entrar en detalles de programación con Firebase exploraremos la estructura de datos y las clases en las que trabajaremos.

### Estructura de datos:



## Pedido.class

```
public class Pedido {

  private String key;
  private int estado;
  private List<Carrito> carrito;
  private String usuario;
  private String usuarioId;
  private String direccion;
  private String telefono;
  private String observaciones;
  private double monto;
  private double lat;
  private double lon;

  //Campos de AUD
  private String usuarioAtendido;
  private String usuarioDelivery;
  private String fechaPedido;
  private String fechaDespacho;
  private String fechaEntrega;
}
```

Las funciones desarrolladas en el taller se realizarán sobre las siguientes clases:

### Relacionadas a productos.

/controller/server/**FirebaseProductosController.class**

```
public class FirebaseProductosController {

  private Activity activity;
  private ResultadoGestion callBackGestion;
  private ResultadoLista callBackLista;

  public FirebaseProductosController(Activity activity, ResultadoGestion callBackGestion) {
    this.activity = activity;
    this.callBackGestion = callBackGestion;
  }

  public FirebaseProductosController(Activity activity, ResultadoLista callBackLista) {
    this.activity = activity;
    this.callBackLista = callBackLista;
  }
}
```

```

public void AdjuntarFoto(Uri uriFoto){
    //TODO adjuntar foto

    /*callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,
        taskSnapshot.getMetadata().getDownloadUrl().toString());*/
}

public void CrearProducto(Producto producto){
    //TODO Agregar pedido

    /*if(databaseError!=null){
        callBackGestion.onResponse(AppConstants.RESULTADO_INCORRECTO,
            databaseError.getMessage());
    } else {

        callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,activity.getResources().getString(R.s
            tring.registro_correcto));
    }*/
}

public void ListasProductos(String categoria){
    DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();
    Query ref;
    if(!categoria.isEmpty()){
        ref=databaseReference.child(AppConstants.TAG_PRODUCTO)
            .orderByChild(AppConstants.TAG_CATEGORIA).equalTo(categoria);
    } else {
        ref=databaseReference.child(AppConstants.TAG_PRODUCTO);
    }

    //TODO Listar categoria

    /*List<Producto> lstProductos=new ArrayList<Producto>();
    for(DataSnapshot dataProducto : dataSnapshot.getChildren()){
        Producto producto=dataProducto.getValue(Producto.class);
        producto.setKey(dataProducto.getKey());
        lstProductos.add(producto);
    }

    Log.e("Categoria",new Gson().toJson(lstProductos));

    callBackLista.onResponse(lstProductos);*/
}
}

public interface ResultadoGestion{
    public void onResponse(int codigoResultado, String mensaje);
}

public interface ResultadoLista{
    public void onResponse(List<Producto> lstProductos);
}
}

```

## Relacionadas a pedidos.

/controller/server/**FirebasePedidosController.class**

```

public class FirebasePedidosController {
    private Activity activity;
    private ResultadoGestion callBackGestion;
    private ResultadoLista callBackLista;

    public FirebasePedidosController(Activity activity, ResultadoGestion callBackGestion) {
        this.activity = activity;
        this.callBackGestion = callBackGestion;
    }

    public FirebasePedidosController(Activity activity, ResultadoLista callBackLista) {
        this.activity = activity;
        this.callBackLista = callBackLista;
    }

    public void CrearPedido(Pedido pedido){
        //TODO Enviar pedido
        /*if(databaseError!=null){
            callBackGestion.onResponse(AppConstants.RESULTADO_INCORRECTO,
                databaseError.getMessage());
        } else {
            callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,
                activity.getResources().getString(R.string.registro_correcto));
        }*/
    }

    public void ListasPedidos(int estado){
        //TODO Listar pedidos

        /*List<Pedido> lstPedidos=new ArrayList<Pedido>();
        for(DataSnapshot dataPedido : dataSnapshot.getChildren()){
            Pedido pedido=dataPedido.getValue(Pedido.class);
            pedido.setKey(dataPedido.getKey());
            lstPedidos.add(pedido);
        }
        callBackLista.onResponse(lstPedidos);*/
    }

    public interface ResultadoGestion{
        public void onResponse(int codigoResultado, String mensaje);
    }

    public interface ResultadoLista{
        public void onResponse(List<Pedido> lstPedidos);
    }
}

```

### a. Adjuntar imagen a Firebase

La función a implementarse será:

```
public void AdjuntarFoto(Uri uriFoto){

    FirebaseStorage storage = FirebaseStorage.getInstance();
    StorageReference storageReference = storage.getReferenceFromUrl(AppConstants.TAG_STORAGE);
    final StorageReference photoReference =
    storageReference.child(AppConstants.TAG_IMAGENES_PRODUCTOS).child(uriFoto.getLastPathSegment
());
    photoReference.putFile(uriFoto).addOnFailureListener(activity, new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {

callBackGestion.onResponse(AppConstants.RESULTADO_INCORRECTO,activity.getResources().getString
(R.string.error_adjuntar));
        }
    }).addOnSuccessListener(activity, new OnSuccessListener<UploadTask.TaskSnapshot>() {
        @Override
        public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
            callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,
                taskSnapshot.getMetadata().getDownloadUrl().toString());
        }
    });
}
```

### b. Enviar producto

Una vez adjuntada la imagen, la función a implementarse será:

```
public void CrearProducto(Producto producto){
    DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();
    databaseReference.child(AppConstants.TAG_PRODUCTO).push().setValue(producto, new
DatabaseReference.CompletionListener() {
    @Override
    public void onComplete(DatabaseError databaseError, DatabaseReference databaseReference) {
        if(databaseError!=null){
            callBackGestion.onResponse(AppConstants.RESULTADO_INCORRECTO,
databaseError.getMessage());
        } else {

callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,activity.getResources().getString(R
.string.registro_correcto));
        }
    }
});
```

### 3. Listar productos

La función a implementarse será:

```
public void ListasProductos(String categoria){
    DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();

    Query ref;
    if(!categoria.isEmpty()){
        ref=databaseReference.child(AppConstants.TAG_PRODUCTO)
            .orderByChild(AppConstants.TAG_CATEGORIA).equalTo(categoria);
    } else {
        ref=databaseReference.child(AppConstants.TAG_PRODUCTO);
    }

    ref.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {

            List<Producto> lstProductos=new ArrayList<Producto>();
            for(DataSnapshot dataProducto : dataSnapshot.getChildren()){
                Producto producto=dataProducto.getValue(Producto.class);
                producto.setKey(dataProducto.getKey());
                lstProductos.add(producto);
            }

            Log.e("Categoria",new Gson().toJson(lstProductos));

            callBackLista.onResponse(lstProductos);
        }

        @Override
        public void onCancelled(DatabaseError databaseError) {
        }
    });
}
```

### 4. Enviar pedidos

La función a implementarse será:

```
public void CrearPedido(Pedido pedido){
    //Firebase upload object
    DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();
    databaseReference.child(AppConstants.TAG_PEDIDOS).push().setValue(pedido, new
    DatabaseReference.CompletionListener() {
        @Override
        public void onComplete(DatabaseError databaseError, DatabaseReference databaseReference) {
            if(databaseError!=null){
                callBackGestion.onResponse(AppConstants.RESULTADO_INCORRECTO,
                    databaseError.getMessage());
            }
        }
    });
}
```

```
        } else {
            callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,
                activity.getResources().getString(R.string.registro_correcto));
        }
    });
}
```

## **5. Listar pedidos por filtros:**

La función a implementarse será:

```
public void ListasPedidos(int estado){  
    DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();  
    databaseReference.child(AppConstants.TAG_PEDIDOS).orderByChild(AppConstants.TAG_ESTADO)  
        .equalTo(estado).addValueEventListener(new ValueEventListener() {  
        @Override  
        public void onDataChange(DataSnapshot dataSnapshot) {  
            List<Pedido> lstPedidos=new ArrayList<Pedido>();  
            for(DataSnapshot dataPedido : dataSnapshot.getChildren()){  
                Pedido pedido=dataPedido.getValue(Pedido.class);  
                pedido.setKey(dataPedido.getKey());  
                lstPedidos.add(pedido);  
            }  
            callBackLista.onResponse(lstPedidos);  
        }  
        @Override  
        public void onCancelled(DatabaseError databaseError) {  
        }  
    });  
}
```

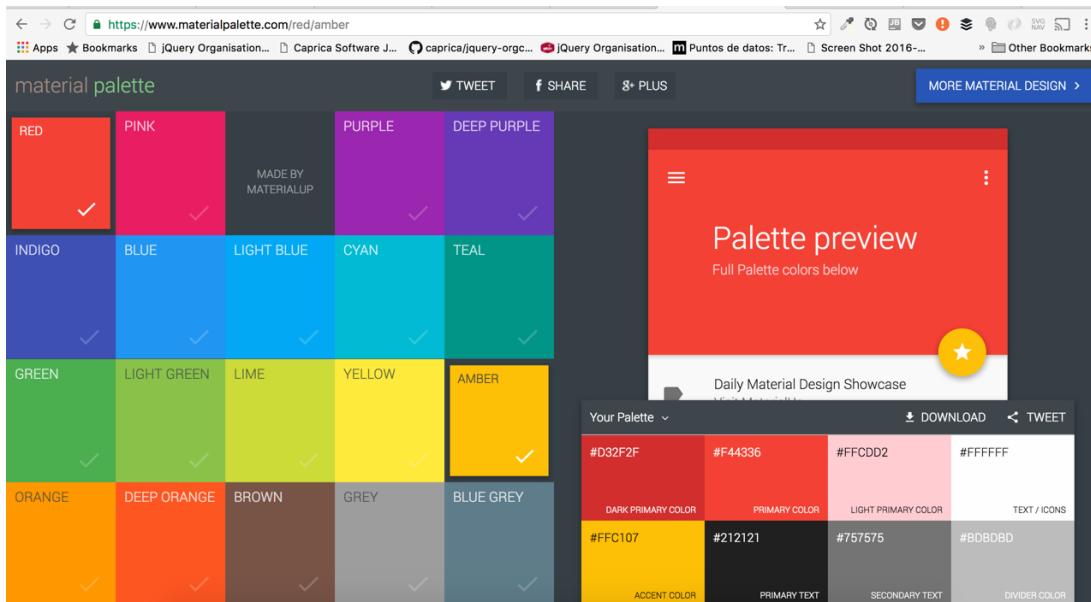
El código fuente del proyecto se encontrará disponible en el repositorio público en Github.

<https://github.com/andres-vasquez/devLivery>

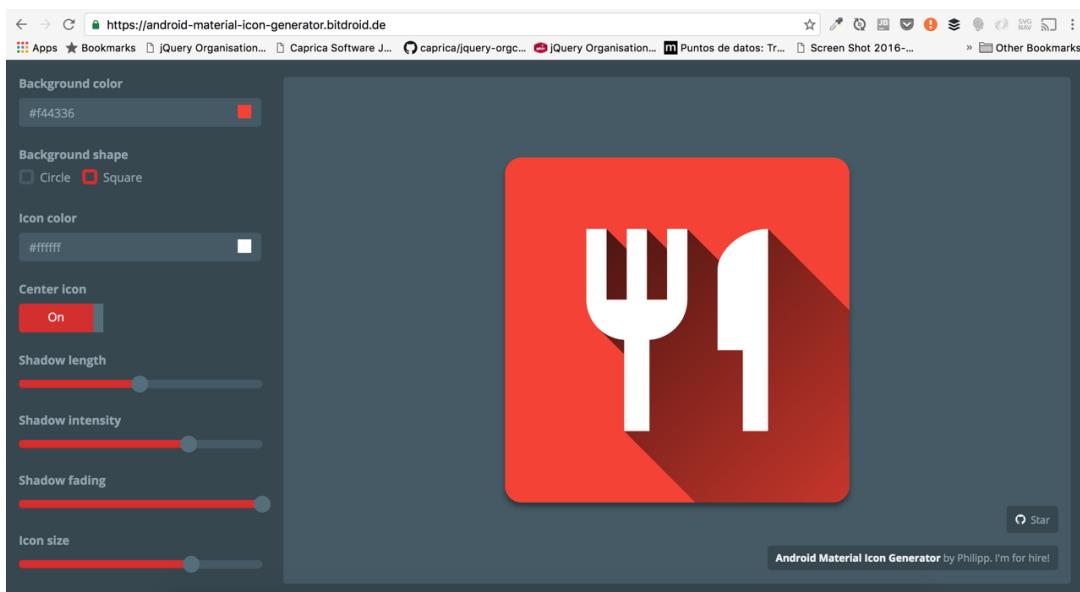
## Otros recursos

Creación del tema en Material Design.

Selección de colores: <https://www.materialpalette.com/>



Icono: <https://android-material-icon-generator.bitdroid.de/>



Otros íconos utilizados: <http://fontawesome.io/icons/>

## Archivos finales:

### **FirebasePedidosController**

```

public class FirebasePedidosController {
    private Activity activity;
    private ResultadoGestion callBackGestion;
    private ResultadoLista callBackLista;

    public FirebasePedidosController(Activity activity, ResultadoGestion callBackGestion) {
        this.activity = activity;
        this.callBackGestion = callBackGestion;
    }

    public FirebasePedidosController(Activity activity, ResultadoLista callBackLista) {
        this.activity = activity;
        this.callBackLista = callBackLista;
    }

    public void CrearPedido(Pedido pedido){
        //Firebase upload object
        DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();
        databaseReference.child(AppConstants.TAG_PEDIDOS).push().setValue(pedido, new
DatabaseReference.CompletionListener() {
            @Override
            public void onComplete(DatabaseError databaseError, DatabaseReference databaseReference) {
                if(databaseError!=null){
                    callBackGestion.onResponse(AppConstants.RESULTADO_INCORRECTO,
                        databaseError.getMessage());
                } else {
                    callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,
                        activity.getResources().getString(R.string.registro_correcto));
                }
            }
        });
    }

    public void ListasPedidos(int estado){
        DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();
        databaseReference.child(AppConstants.TAG_PEDIDOS).orderByChild(AppConstants.TAG_ESTADO)
            .equalTo(estado).addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                List<Pedido> lstPedidos=new ArrayList<Pedido>();
                for(DataSnapshot dataPedido : dataSnapshot.getChildren()){
                    Pedido pedido=dataPedido.getValue(Pedido.class);
                    pedido.setKey(dataPedido.getKey());
                    lstPedidos.add(pedido);
                }
                callBackLista.onResponse(lstPedidos);
            }
            @Override
            public void onCancelled(DatabaseError databaseError) {
            }
        });
    }
}

```

```
        });
    }

public interface ResultadoGestion{
    public void onResponse(int codigoResultado, String mensaje);
}

public interface ResultadoLista{
    public void onResponse(List<Pedido> lstPedidos);
}
```

## FirebaseProductosController

```
public class FirebaseProductosController {  
  
    private Activity activity;  
    private ResultadoGestion callBackGestion;  
    private ResultadoLista callBackLista;  
  
    public FirebaseProductosController(Activity activity, ResultadoGestion callBackGestion) {  
        this.activity = activity;  
        this.callBackGestion = callBackGestion;  
    }  
  
    public FirebaseProductosController(Activity activity, ResultadoLista callBackLista) {  
        this.activity = activity;  
        this.callBackLista = callBackLista;  
    }  
  
    public void AdjuntarFoto(Uri uriFoto){  
  
        FirebaseStorage storage = FirebaseStorage.getInstance();  
        StorageReference storageReference = storage.getReferenceFromUrl(AppConstants.TAG_STORAGE);  
        final StorageReference photoReference =  
storageReference.child(AppConstants.TAG_IMAGENES_PRODUCTOS).child(uriFoto.getLastPathSegment()  
());  
        photoReference.putFile(uriFoto).addOnFailureListener(activity, new OnFailureListener() {  
            @Override  
            public void onFailure(@NonNull Exception e) {  
  
                callBackGestion.onResponse(AppConstants.RESULTADO_INCORRECTO,activity.getResources().getString(R.string.error_adjuntar));  
            }  
        }).addOnSuccessListener(activity, new OnSuccessListener<UploadTask.TaskSnapshot>() {  
            @Override  
            public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
                callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,  
                    taskSnapshot.getMetadata().getDownloadUrl().toString());  
            }  
        });  
    }  
  
    public void CrearProducto(Producto producto){
```

```

DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();
databaseReference.child(AppConstants.TAG_PRODUCTO).push().setValue(producto, new
DatabaseReference.CompletionListener() {
    @Override
    public void onComplete(DatabaseError databaseError, DatabaseReference databaseReference) {
        if(databaseError!=null){
            callBackGestion.onResponse(AppConstants.RESULTADO_INCORRECTO,
databaseError.getMessage());
        } else {

            callBackGestion.onResponse(AppConstants.RESULTADO_CORRECTO,activity.getResources().getString(R
.string.registro_correcto));
        }
    });
}

public void ListasProductos(String categoria){
    DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();

    Query ref;
    if(!categoria.isEmpty()){
        ref=databaseReference.child(AppConstants.TAG_PRODUCTO)
            .orderByChild(AppConstants.TAG_CATEGORIA).equalTo(categoria);
    } else {
        ref=databaseReference.child(AppConstants.TAG_PRODUCTO);
    }

    ref.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {

            List<Producto> lstProductos=new ArrayList<Producto>();
            for(DataSnapshot dataProducto : dataSnapshot.getChildren()){
                Producto producto=dataProducto.getValue(Producto.class);
                producto.setKey(dataProducto.getKey());
                lstProductos.add(producto);
            }

            Log.e("Categoria",new Gson().toJson(lstProductos));

            callBackLista.onResponse(lstProductos);
        }

        @Override
        public void onCancelled(DatabaseError databaseError) {
        }
    });
}

public interface ResultadoGestion{
    public void onResponse(int codigoResultado, String mensaje);
}

```

```
public interface ResultadoLista{  
    public void onResponse(List<Producto> lstProductos);  
}  
}
```