## Ejercicio de desarrollo API Heroku con Node.js



## **Endpoint Node.js**

```
//nttps://console.firebase.google.com/d/w/project/petagramm/var/
var registrarUsuarioDb="usuario_instagram";
app.post('/registrar-usuario', function(request, response) {
    //var id_dispositivo=request.body.id_dispositivo;
    // var id_usuario_instagram=request.body.id_usuario_instagram;
                 var db=firebase.database();
var userDevices=db.ref(registrarUsuarioDb).push();
                 });
tokenDevicesURI=registrarUsuarioDb
                 var path=userDevices.toString();
var pathSplit=path.split(registrarUsuarioDb + "/");
var idAutoGenerado=pathSplit[1];
                 var respuesta=generarRespuestaAUsuarioInstagram(db,idAutoGenerado);
response.setHeader("Content-Type", "application/json");
response.send(JSON.stringify(respuesta));
103
104
105
106
107
          function generarRespuestaAUsuarioInstagram(db,idAutoGenerado){
                 var respuesta={};
                 var usuario='
                 //nombre de la base de datos que generalmetne es el nombre del uri
var ref=db.ref(tokenDevicesURI);
ref.on("child_added",function(snapshot,prevChildKey){
                        usuario=snapshot.val();
                        respuesta={
id:idAutoGenerado,
                                id_dispositivo:usuario.id_dispositivo,
id_usuario_instagram:usuario.id_usuario_instagram
                       urn respuesta:
```

## Llamada desde Android

```
public void EnviarUsuarioApi(){

//envio notificacion a servidor heroku con usuario y token

RestApiAdapter restApiAdapter=new RestApiAdapter();
EndpointsApi endpointsApi=restApiAdapter.estRestApiAdapter();

EndpointsApi endpointsApi=restApiAdapter.estRestApiAdapter();

CallcUsuarioRespuesta> usuarioRespuestaCall=endpointsApi.registrarUsuarioInstagram(|id_dispositivo: "123456", id_usuario_instagram: "miUsuarioInstagram");

usuarioRespuestaCall.enqueue(new Callback<UsuarioRespuesta>() {

    @Override
    public void onResponse(Call<UsuarioRespuesta> call, Response<UsuarioRespuesta> response)

    | UsuarioRespuesta usuarioRespuesta=response.body();

    //guardo en el log la respuesta.
    | Log.d( tag: "ID_DISPOSITIVO", usuarioRespuesta.getId());

    | Log.d( tag: "ID_USUARIO", usuarioRespuesta.getId_usuario_instagram());

    | Colored en public void onFailure(Call<UsuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

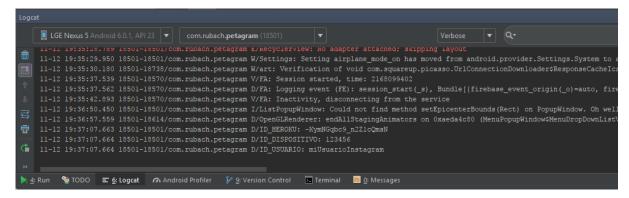
    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throwable t) {

    | Log.d( tag: "ID_HEROKU", usuarioRespuesta> call, Throw
```

Resultado del llamado al endpoint, como se ve se guarda en log las tres variables: ID, Id\_dipositivo e Id\_usuarioInstagram



La aplicación corre satisfactoriamente implementando retrofit2, heroku, node.js y firebase.

Por un tema de tiempo, no levanta el usuario real de Instagram, se envía uno creado en memoria.