

Desarrollo WEB Full Stack

Maikoll Fabián Ballesteros Pinilla

Breider Yesid López Valero

Docente

JUAN PABLO GARZON RUIZ

Universidad de Cundinamarca

Seccional Ubaté

Ingeniería de Sistemas

2025

Contenido

CÓDIGO	3
Resultados	9
Links	11

CÓDIGO

Figura 1.

Función JavaScript para el envío de notificaciones entre usuarios

```
// functions/sendNotification.js
const admin = require('firebase-admin');
const serviceAccount = require('./serviceAccountKey.json');

// Inicializar Firebase Admin solo una vez
if (!admin.apps.length) {
  admin.initializeApp({
    credential: admin.credential.cert(serviceAccount)
  });
}

exports.handler = async function(event, context) {
  // Permitir CORS para la app Android
  const headers = {
    'Access-Control-Allow-Origin': '*',
    'Access-Control-Allow-Headers': 'Content-Type',
    'Access-Control-Allow-Methods': 'POST, OPTIONS'
  };

  // Manejar solicitudes OPTIONS para CORS pre-flight
  if (event.httpMethod === 'OPTIONS') {
    return {
      statusCode: 200,
      headers,
      body: JSON.stringify({ message: 'Preflight request successful' })
    };
  }

  // Procesar sólo solicitudes POST
  if (event.httpMethod !== 'POST') {
    return {
      statusCode: 405,
      headers,
      body: JSON.stringify({ message: 'Method not allowed' })
    };
  }
}
```

```

try {
  // Parsear el cuerpo de la solicitud
  const body = JSON.parse(event.body);
  console.log('Recibida solicitud de notificación:', body);

  // Extraer datos
  const { token, title, body: messageBody, data } = body;

  if (!token) {
    return {
      statusCode: 400,
      headers,
      body: JSON.stringify({ message: 'Token no proporcionado' })
    };
  }
}

```

```

// Crear mensaje para FCM
const message = {
  token,
  notification: {
    title,
    body: messageBody
  },
  data: data || {},
  android: {
    priority: 'high',
    notification: {
      sound: 'default',
      priority: 'high',
      channelId: 'high_importance_channel'
    }
  }
};

```

```
// Enviar la notificación
console.log('Enviando mensaje FCM:', message);
const response = await admin.messaging().send(message);
console.log('Notificación enviada correctamente:', response);

return {
  statusCode: 200,
  headers,
  body: JSON.stringify({
    success: true,
    messageId: response
  })
};

} catch (error) {
  console.error('Error al enviar la notificación:', error);

  return {
    statusCode: 500,
    headers,
    body: JSON.stringify({
      success: false,
      error: error.message
    })
  };
};
```

Figura 2.

HTML para la visualización Web

```

</head>
<body>
<div class="container">
  <div class="avatar"></div>
  <h2>Nombre del Trabajador</h2>

  <form id="serviceForm">
    <label for="direccion">Dirección</label>
    <input type="text" id="direccion" required>

    <label for="tipo">Tipo de Servicio</label>
    <select id="tipo" required>
      <option value="Jardinería">Jardinería</option>
      <option value="Carpintería">Carpintería</option>
      <option value="Electricidad">Electricidad</option>
      <option value="Plomería">Plomería</option>
    </select>

    <label for="descripcion">Descripción del servicio</label>
    <textarea id="descripcion" rows="4" required></textarea>

    <button type="submit">SIGUIENTE</button>
  </form>
</div>

```

Figura 3.

Implementación en Android Studio enlazado con Firebase

```
try {  
  
    String url = "https://adsv.netlify.app/.netlify/functions/sendNotification";  
  
    // Para depuración  
    Log.d( tag: "NotificationProvider", msg: "Enviando notificación a: " + url);  
    Log.d( tag: "NotificationProvider", msg: "Token: " + token);  
    Log.d( tag: "NotificationProvider", msg: "Datos: " + data.toString());  
  
    JSONObject jsonBody = new JSONObject();  
    try {  
        jsonBody.put( name: "token", token);  
        jsonBody.put( name: "title", title);  
        jsonBody.put( name: "body", body);  
  
        JSONObject dataJson = new JSONObject();  
        for (Map.Entry<String, Object> entry : data.entrySet()) {  
            dataJson.put(entry.getKey(), entry.getValue());  
        }  
        jsonBody.put( name: "data", dataJson);  
  
        Log.d( tag: "NotificationProvider", msg: "JSON a enviar: " + jsonBody.toString());  
    } catch (JSONException e) {  
        Log.e( tag: "NotificationProvider", msg: "Error creando JSON: ", e);  
        taskCompletionSource.setException(e);  
    }  
}
```

1 usage zXpect


```
private void submitRequest(String address, String description, String serviceType) {
    String clientId = FirebaseAuth.getInstance().getCurrentUser().getUid();
    Map<String, Object> requestData = new HashMap<>();

    requestData.put("client_id", clientId);
    requestData.put("address", address);
    requestData.put("description", description);
    requestData.put("service_type", serviceType);
    requestData.put("status", "pending");
    requestData.put("timestamp", System.currentTimeMillis());

    // Mostrar indicador de progreso
    showProgressDialog("Enviando solicitud...");

    mRequestProvider.createRequest(requestData)
        .addOnSuccessListener(aVoid -> {
            hideProgressDialog();
            showToast("Solicitud creada con éxito");
            navigateToVerifyRequest(address, serviceType);
        });
}
```

Realtime Database

 Preguntarle a Gemini sobre Realtime Database

[Datos](#)

[Reglas](#)

[Copias de seguridad](#)

[Uso](#)


[Extensions](#)

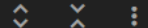


Protege tus recursos de Realtime Database contra los abusos, como fraudes de facturación o phishing.

[Configurar la Verificación de aplicaciones](#)



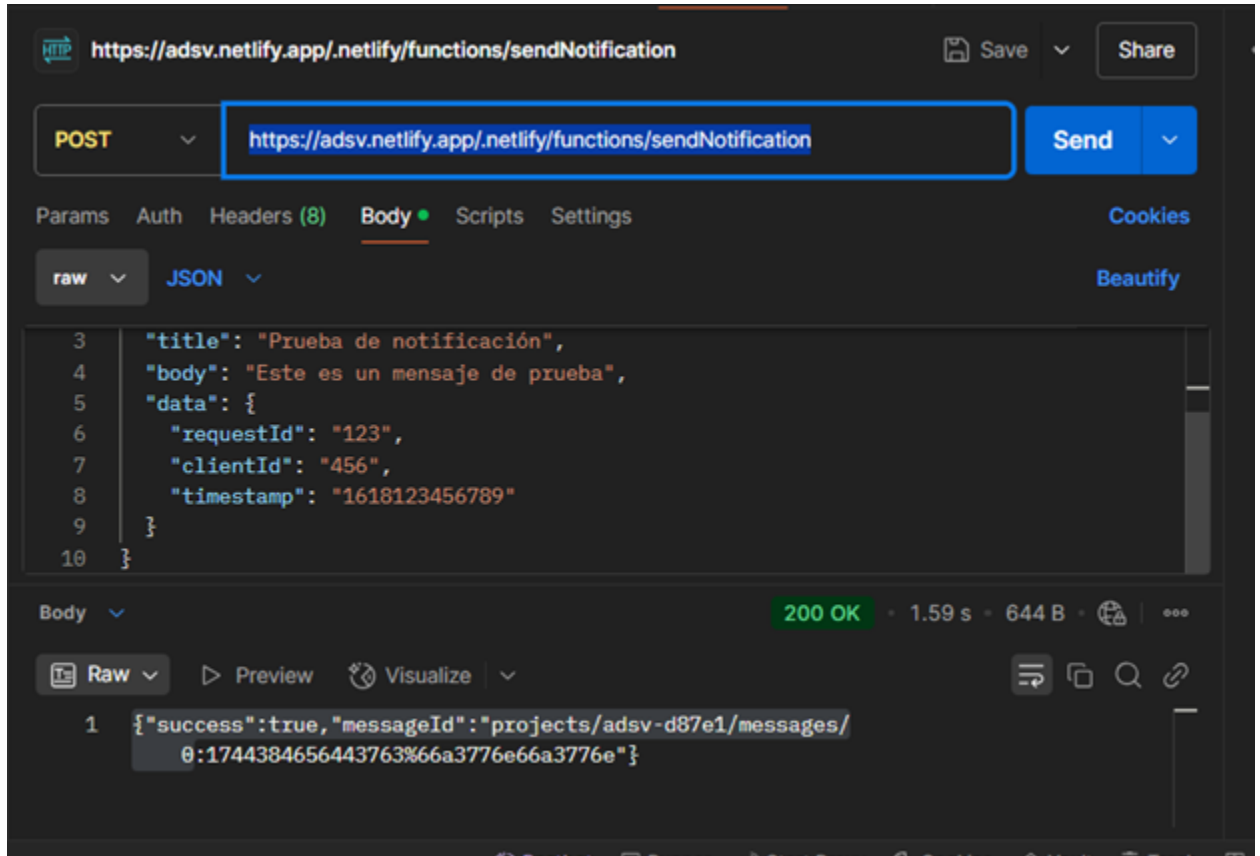
 <https://adsv-d87e1-default-rtdb.firebaseio.com>



<https://adsv-d87e1-default-rtdb.firebaseio.com/>

- ▶ — Tokens
- ▶ — User
- ▶ — active_workers
- ▶ — requests

Resultados



The screenshot shows a REST client interface with the URL `https://adsv.netlify.app/.netlify/functions/sendNotification`. The request is a POST with a JSON body. The response is a 200 OK status with a response time of 1.59 s and a body size of 644 B. The response body is a JSON object indicating success and providing a message ID.

```
POST https://adsv.netlify.app/.netlify/functions/sendNotification
```

Params Auth Headers (8) **Body** Scripts Settings Cookies Beautify

raw JSON

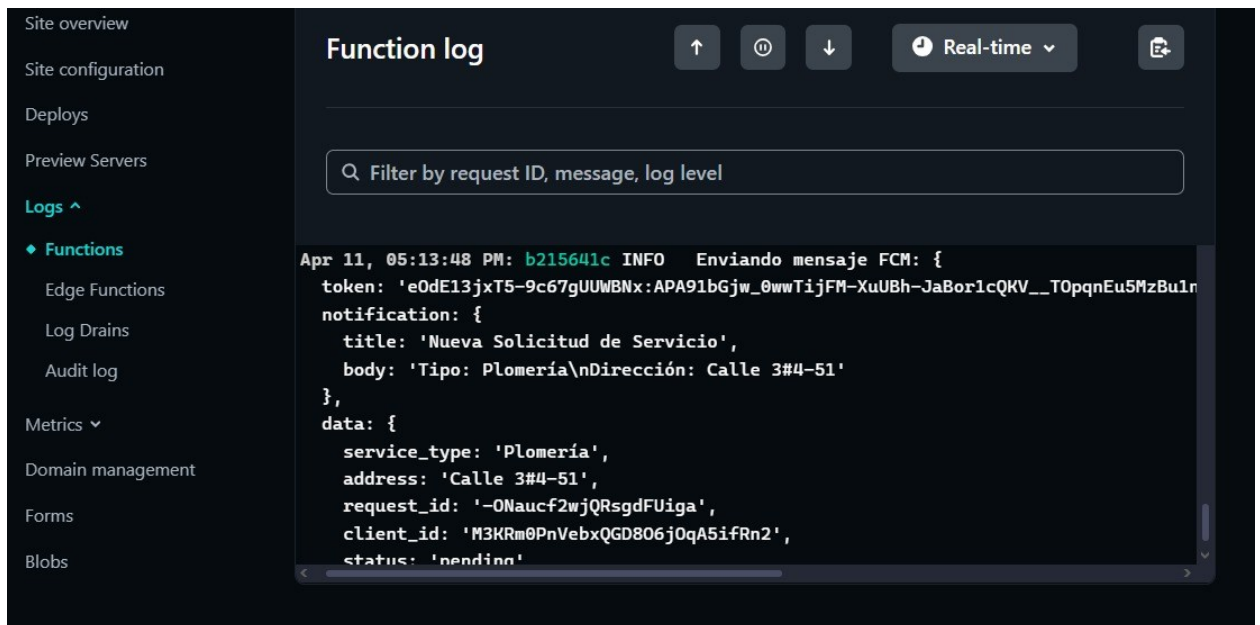
```
3  {"title": "Prueba de notificación",
4  "body": "Este es un mensaje de prueba",
5  "data": {
6    "requestId": "123",
7    "clientId": "456",
8    "timestamp": "1618123456789"
9  }
10 }
```

Body 200 OK · 1.59 s · 644 B

Raw Preview Visualize

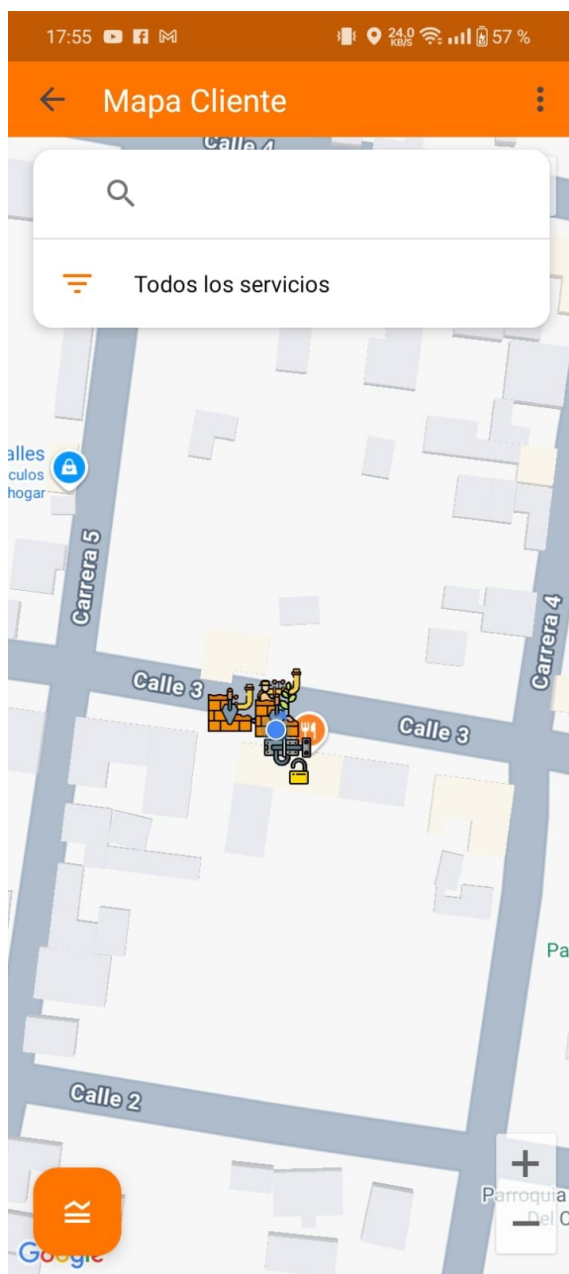
```
1  {"success":true,"messageId":"projects/adsv-d87e1/messages/
   0:1744384656443763%66a3776e66a3776e"}

```



The screenshot shows the Netlify Functions log interface. The left sidebar contains navigation links for Site overview, Site configuration, Deploys, Preview Servers, Logs, Functions, Edge Functions, Log Drains, Audit log, Metrics, Domain management, Forms, and Blobs. The main area displays the Function log for the 'sendNotification' function. The log shows a successful message being sent to FCM with the following details:

```
Apr 11, 05:13:48 PM: b215641c INFO Enviando mensaje FCM: {
  token: 'e0dE13jxT5-9c67gUUWBNx:APA91bGjw_0wwTijFM-XuUBh-JaBor1cQKV__T0pqnEu5MzBu1n',
  notification: {
    title: 'Nueva Solicitud de Servicio',
    body: 'Tipo: Plomería\nDirección: Calle 3#4-51'
  },
  data: {
    service_type: 'Plomería',
    address: 'Calle 3#4-51',
    request_id: '-0Naucf2wjQRsgdFUiga',
    client_id: 'M3KRm0PnVebxQGD806jOqA5ifRn2',
    status: 'pending'
  }
}
```



The screenshot shows the service request form in the app. At the top, there's a status bar with the time 17:55, battery level at 57%, and various icons. Below the status bar is a light purple header with a circular profile picture of a worker wearing a yellow hard hat. Below the profile picture is the text 'Nombre del Trabajador'. The main section is titled 'SOLICITAR SERVICIO'. It contains three input fields: 'Dirección' with the value 'Calle 3#4-51', 'Tipo de Servicio' with a dropdown menu showing 'Plomería', and 'Descripción del servicio' with the value 'Tubería Rota'. At the bottom, there's a large orange button labeled 'SIGUIENTE'.

Links

Repositorio en GitHub: <https://github.com/zXpect/ADSV.git>

Deploy Netlify: <https://adsv.netlify.app>

Nota: El envío de notificación exitoso depende el estado del token, es decir si el usuario trabajador está conectado y disponible, de lo contrario el envío no saldrá con éxito.