Payment Gateway Integration Tutorial

Step-by-step guide to accepting payments in 5 steps

Version: 1.0

Author: Developer Relations Team

Category: Integration Guide | Payment Processing

Completion Time: 30-45 minutes

Last Updated: December 12, 2025

This hands-on tutorial walks you through integrating Monzo payment gateway into your application in 5 clear steps. By the end, you will accept card payments, process bank transfers, and handle webhooks for real-time payment notifications. Estimated completion time: 30-45 minutes.

Prerequisites

Before starting, ensure you have: Monzo Business account with API access, OAuth 2.0 credentials (Client ID and Secret) from Developer Portal, Node.js 18+ or Python 3.9+, HTTPS endpoint for webhooks (use ngrok for local testing), and basic REST API understanding.

Step 1: Set Up Development Environment

Install Dependencies

```
# Node.js
npm init -y
npm install express axios dotenv

# Python
python -m venv venv
source venv/bin/activate
pip install flask requests python-dotenv
```

Configure Environment Variables

```
MONZO_CLIENT_ID=oauth2client_00009A1BK4MH...
MONZO_CLIENT_SECRET=mnzconf.Z7pBfXE2rD...
MONZO_REDIRECT_URI=http://localhost:3000/callback
MONZO_WEBHOOK_SECRET=whsec_kXXDjfj2K...
MONZO_BASE_URL=https://api.monzo.com
```

> ***NOTE:*** *Use Monzo sandbox environment for testing. Sandbox accounts are pre-loaded with test funds and do not process real payments.*

Step 2: Implement OAuth Authentication

```
const express = require('express');
const axios = require('axios');
require('dotenv').config();

const app = express();

app.get('/connect', (req, res) => {
    const authUrl = new URL('https://auth.monzo.com/');
    authUrl.searchParams.append('client_id',
        process.env.MONZO_CLIENT_ID);
    authUrl.searchParams.append('redirect_uri',
        process.env.MONZO_REDIRECT_URI);
    authUrl.searchParams.append('response_type', 'code');
    res.redirect(authUrl.toString());
});

app.get('/callback', async (req, res) => {
    const { code } = req.query;
    const tokenResponse = await axios.post(
        process.env.MONZO_BASE_URL + '/oauth2/token',
        new URLSearchParams({
            grant_type: 'authorization_code',
            client_id: process.env.MONZO_CLIENT_ID,
            client_secret: process.env.MONZO_CLIENT_SECRET,
            redirect_uri: process.env.MONZO_REDIRECT_URI,
            code: code
        })
    );
    global.accessToken = tokenResponse.data.access_token;
    res.send('Authorization successful!');
});
```

WARNING: In production, store access tokens encrypted in a database. Never store tokens in global variables or client-side code.

Step 3: Create Payment Intent

```
app.post('/create-payment', express.json(),
    async (req, res) => {
    const { amount, currency, description } = req.body;

    const payment = await axios.post(
        process.env.MONZO_BASE_URL + '/payment-intents',
        {
            amount: Math.round(amount * 100),
            currency: currency.toUpperCase(),
            description: description || 'Payment'
        },
        {
            headers: {
                'Authorization': 'Bearer ' + global.accessToken
            }
        }
    );

    res.json({
        payment_intent_id: payment.data.id,
        client_secret: payment.data.client_secret
    });
});
```

Step 4: Set Up Webhook Handler

```
app.post('/webhooks/monzo',
    express.raw({ type: 'application/json' }),
    async (req, res) => {
    const signature = req.headers['x-monzo-signature'];
    const payload = req.body;

    const hmac = require('crypto').createHmac('sha256',
        process.env.MONZO_WEBHOOK_SECRET);
    const expectedSig = 'v1=' +
        hmac.update(payload).digest('hex');

    if (signature !== expectedSig) {
        return res.status(401).send('Invalid signature');
    }

    const event = JSON.parse(payload);

    if (event.type === 'payment.succeeded') {
        console.log('Payment successful:', event.data.id);
        // Fulfill order here
    }

    res.status(200).send('OK');
});
```

Step 5: Test Integration

1. Start your server: node app.js

2. Visit http://localhost:3000/connect to authorize

3. Create test payment using curl or Postman

```
curl -X POST http://localhost:3000/create-payment \
  -H "Content-Type: application/json" \
  -d '{"amount": 10.50, "currency": "GBP"}'
```

4. Use Monzo test cards:

Test Scenario	Card Number	Result
Success	4242 4242 4242 4242	Payment succeeds
Declined	4000 0000 0000 0002	Payment declined
3DS Required	4000 0025 0000 3155	Triggers 3D Secure

SUCCESS: Integration complete! You can now accept payments through Monzo. Next steps: Add proper error handling, implement token refresh, set up logging, and deploy to production.