

LAPORAN TUGAS KELOMPOK 01 PEMROGRAMAN API

“ QRIS Payment Midtrans – Kasir Café ”



Dosen Pengampu :

Saiful Nur Budiman, S.Kom., M.Kom.

Disusun Oleh :

Isra Naswa Reyka Swahili (23104410006)

Agistha Ardha Sulisty P. (23104410009)

Zaki Zakaria Zakse (23104410010)

Jusafa Ido Ad'hareza (23104410022)

Jovanda Kelvin Wibawa P. (23104410035)

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK DAN INFORMATIKA
UNIVERSITAS ISLAM BALITAR

Oktober 2025

BAB I

PENDAHULUAN

1.1 Latar Belakang

Seiring dengan meningkatnya tren digitalisasi, sistem pembayaran non-tunai menjadi solusi praktis di berbagai bidang usaha, termasuk café dan restoran. Salah satu metode pembayaran yang paling banyak digunakan di Indonesia saat ini adalah QRIS (Quick Response Code Indonesian Standard) yang memungkinkan berbagai aplikasi e-wallet seperti GoPay, OVO, dan DANA digunakan dalam satu kode QR yang terintegrasi.

Dalam tugas ini, kami membuat sistem Kasir Café berbasis web dengan menggunakan framework Laravel versi 12, yang diintegrasikan dengan Midtrans API sebagai *payment gateway*. Sistem ini dijalankan di Laragon untuk mempermudah pengembangan lokal, dan menggunakan Ngrok sebagai alat *tunneling* agar server lokal dapat diakses publik oleh Midtrans.

Tujuan utama dari proyek ini adalah memahami cara kerja integrasi API pembayaran QRIS, proses autentikasi API key Midtrans, serta otomatisasi status transaksi pada sistem kasir digital.

1.2 Dasar Teori

a. Application Programming Interface (API)

API adalah antarmuka yang menghubungkan dua sistem agar dapat saling berkomunikasi. Dalam proyek ini, API digunakan oleh aplikasi Laravel untuk mengirim data transaksi ke server Midtrans dan menerima *callback* notifikasi status pembayaran.

b. Midtrans

Midtrans merupakan layanan *payment gateway* Indonesia yang menyediakan API untuk pembayaran menggunakan QRIS, GoPay, transfer bank, dan metode lain. Midtrans menyediakan mode Sandbox untuk simulasi transaksi sebelum sistem digunakan secara resmi.

c. Laravel 12

Laravel adalah *framework PHP* yang menggunakan arsitektur Model-View-Controller (MVC). Versi 12 dari Laravel membawa peningkatan pada efisiensi performa, keamanan, serta dukungan *route model binding* dan *job batching*. Dalam proyek ini, Laravel digunakan untuk mengelola logika transaksi, menampilkan antarmuka kasir, dan memproses notifikasi dari Midtrans.

d. Ngrok

Ngrok berfungsi untuk membuat *tunnel* yang menghubungkan server lokal ke internet. Dengan Ngrok, URL lokal (misalnya <http://kasircafe.test>) dapat diakses oleh Midtrans untuk mengirimkan notifikasi *callback* setelah transaksi selesai.

BAB II

IMPLEMENTASI PROGRAM

2.1 Langkah-langkah Penggerjaan

- 1) Membuat Proyek Laravel 12 di Laragon

laravel new project kasir-new/qrис

- 2) Menginstal Library Midtrans

composer require midtrans/midtrans-php

- 3) Menambahkan Konfigurasi Midtrans di .env

MIDTRANS_SERVER_KEY=SB-Mid-server-XXXX

MIDTRANS_CLIENT_KEY=SB-Mid-client-XXXX

MIDTRANS_IS_PRODUCTION=false

- 4) Membuat PaymentController untuk:

- Membuat transaksi ke API Midtrans.
- Menampilkan QRIS QR Code.
- Menangani *callback* notifikasi.

- 5) Membuat Tampilan (Blade Template) untuk halaman kasir dan QRIS.

- 6) Menjalankan Server Laravel di Laragon

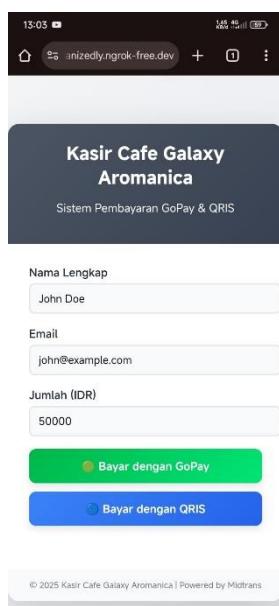
Akses lokal: *http://127.0.0.1:8000*

- 7) Menjalankan Ngrok untuk Koneksi Publik

- 8) Melakukan Pengujian Pembayaran di Sandbox Midtrans

Setelah pelanggan melakukan pembayaran, status transaksi otomatis berubah menjadi *Success* dan pesan “Pembayaran Berhasil” muncul di layar kasir.

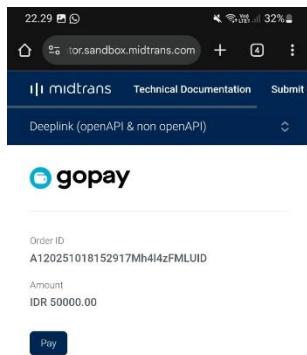
2.2 Tampilan Web



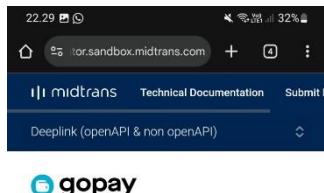
Setelah server aktif, pengguna diarahkan ke dashboard aplikasi kasir. Di halaman ini terdapat menu utama seperti nama lengkap, email, jumlah (IDR), dan tombol untuk memulai proses pembayaran. Tampilan ini menjadi titik awal bagi kasir untuk melayani pesanan pelanggan.



Setelah pelanggan selesai memilih menu dan kasir menekan tombol “Bayar dengan QRIS”, sistem mengirimkan data transaksi ke API Midtrans. Hasilnya, aplikasi menampilkan QR Code QRIS yang dapat dipindai langsung oleh pelanggan menggunakan aplikasi e-wallet seperti DANA.



Jika pelanggan memilih metode pembayaran GoPay, sistem menampilkan halaman transaksi yang memuat detail pembayaran melalui aplikasi GoPay. Tahap ini menunjukkan bahwa integrasi API Midtrans berjalan dengan baik dan sistem mampu menampilkan antarmuka pembayaran yang sesuai dengan metode yang dipilih.



Tahap ini memperlihatkan proses sistem ketika pengguna menekan tombol Pay setelah memilih metode pembayaran. Laravel memproses permintaan tersebut dan menunggu respons dari server Midtrans. Jika berhasil, sistem akan menampilkan status transaksi berikut QRIS atau konfirmasi pembayaran dari Midtrans.



```
C:\ngrok.exe -ngrok http 8001 + v
ngrok
* Create instant endpoints for local containers within Docker Desktop + https://ngrok.com/r/docker
Session Status      online
Account            jovankelvin18@gmail.com (Plan: Free)
Version             3.30.0
Region              Asia Pacific (ap)
Latency             228ms
Web Interface      http://127.0.0.1:4640
Forwarding          https://procoercion-caylee-organizedly.ngrok-free.dev -> http://localhost:8000
Connections         ttl     opn     rt1     rt5     p50     p99
                    11      8      0.00    0.01    0.49    3.58
HTTP Requests
22:34:30.136 +07 GET /favicon.ico      200 OK
22:34:17.555 +07 POST /payment/process 200 OK
22:34:11.498 +07 GET /favicon.ico      200 OK
22:34:09.757 +07 GET /                  200 OK
22:38:06.643 +07 GET /                  200 OK
22:29:16.371 +07 POST /payment/process 200 OK
22:29:11.156 +07 GET /favicon.ico      200 OK
22:29:09.587 +07 GET /                  200 OK
22:28:21.070 +07 GET /                  200 OK
22:28:14.614 +07 GET /favicon.ico      200 OK
```

Setelah pelanggan berhasil melakukan pembayaran, Midtrans mengirimkan callback ke server Laravel melalui Ngrok. Tampilan ini menunjukkan hasil di sisi server, di mana sistem menerima notifikasi bahwa transaksi telah berhasil. Status pembayaran otomatis berubah menjadi *success* di database. Tampilan ini menandai bahwa seluruh proses integrasi API — mulai dari permintaan transaksi, pembayaran, hingga verifikasi — telah berjalan dengan benar.

2.3 Hasil Uji Coba

- QRIS berhasil ditampilkan dan dapat dipindai oleh aplikasi e-wallet.
- Callback dari Midtrans diterima oleh Laravel melalui URL Ngrok.
- Status transaksi di database berubah otomatis menjadi *Berhasil*.
- Sistem menampilkan pesan “Pembayaran Berhasil” di halaman kasir café.

BAB III

PENUTUP

3.1 Kesimpulan

Berdasarkan hasil implementasi, sistem Kasir Café berbasis Laravel 12 dengan integrasi Midtrans API berjalan dengan baik di lingkungan Laragon. Proses pembayaran QRIS dapat dilakukan secara otomatis dan status transaksi berhasil diperbarui tanpa intervensi manual.

Integrasi Ngrok memungkinkan uji coba sistem secara real-time tanpa harus menggunakan hosting. Sistem ini dapat dikembangkan lebih lanjut untuk fitur seperti cetak struk otomatis, laporan penjualan harian, serta integrasi perangkat kasir IoT.

DAFTAR PUSTAKA

- Midtrans. (2025). *Midtrans API documentation*. Jakarta: PT Midtrans. Diakses dari <https://docs.midtrans.com> pada 19 Oktober 2025.
- Laravel. (2025). *Laravel 12 official documentation*. San Francisco: Laravel LLC. Diakses dari <https://laravel.com/docs> pada 19 Oktober 2025.
- Ngrok. (2025). *Secure tunnels to localhost*. Boston: Ngrok, Inc. Diakses dari <https://ngrok.com> pada 19 Oktober 2025.
- Laragon. (2025). *Laragon official documentation*. Hanoi: Le Nguyen Hoang. Diakses dari <https://laragon.org> pada 19 Oktober 2025.
- Bank Indonesia. (2023). *Quick response code Indonesian standard (QRIS)*. Jakarta: Bank Indonesia. Diakses dari <https://www.bi.go.id> pada 19 Oktober 2025.

DIAGRAM ALIR



Penjelasan:

1. Pengguna membuka halaman pembayaran

Proses dimulai ketika pengguna mengakses halaman pembayaran melalui browser. Halaman ini menampilkan form untuk mengisi data seperti nama, email, dan jumlah pembayaran.

2. Pengguna mengisi formulir dan memilih metode 'GoPay / QRIS'

Setelah form diisi, pengguna menekan tombol "GoPay / QRIS" untuk memulai proses pembayaran. Data dari form dikirim ke server Laravel menggunakan metode POST.

3. Laravel mengirim permintaan ke API Inti Midtrans

Laravel memproses data pengguna dan membuat permintaan (request) ke Midtrans Core API menggunakan Server Key yang telah dikonfigurasi di file .env. Permintaan ini berisi detail transaksi seperti order_id, gross_amount, dan payment_type (qrис).

4. Midtrans membuat transaksi dan mengirimkan kode QRIS

Midtrans memproses permintaan tersebut, lalu mengembalikan response JSON yang berisi data transaksi termasuk URL QRIS atau gambar kode QR yang harus ditampilkan ke pengguna.

5. Laravel menampilkan QRIS kepada pengguna

Laravel menampilkan QRIS ke halaman web sehingga pengguna dapat memindainya melalui aplikasi GoPay atau aplikasi pembayaran lain yang mendukung QRIS.

6. Pengguna melakukan pembayaran melalui QRIS / GoPay simulator

Pengguna menyelesaikan pembayaran dengan memindai kode QRIS tersebut. Di mode sandbox, pembayaran dilakukan melalui Midtrans Payment Simulator untuk keperluan pengujian.

7. Midtrans mengirim callback notifikasi ke Laravel

Setelah pembayaran selesai, Midtrans mengirimkan notifikasi callback ke endpoint Laravel (yang diakses melalui Ngrok selama proses pengembangan). Laravel menerima notifikasi tersebut dan memperbarui status transaksi menjadi “settlement” atau “success”.

8. Laravel memperbarui status transaksi menjadi berhasil

Laravel kemudian menyimpan status transaksi ke database (atau session) dan menampilkan pesan bahwa pembayaran berhasil.

9. Halaman menampilkan pesan “Pembayaran Berhasil”

Proses transaksi selesai. Pengguna menerima notifikasi bahwa pembayaran telah berhasil dilakukan.

LAMPIRAN CODING

➤ payment.controller.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Str;
use Illuminate\Support\Facades\Http;
use Illuminate\Support\Facades\Log;

class PaymentController extends Controller
{
    public function index()
    {
        return view('payment'); // View utama di resources/views/payment.blade.php
    }

    public function process(Request $request)
    {
        $request->validate([
            'name' => 'required|string',
            'email' => 'required|email',
            'amount' => 'required|integer|min:1000',
            'payment_type' => 'required|string|in:gopay,qrис',
        ]);

        $serverKey = config('services.midtrans.server_key'); // Pastikan sudah diatur di .env
        $orderId = 'ORDER-' . Str::uuid();

        $payload = [
            'transaction_details' => [
                'order_id' => $orderId,
                'gross_amount' => (int) $request->amount,
            ],
            'customer_details' => [
                'first_name' => $request->name,
                'email' => $request->email,
            ],
            'payment_type' => $request->payment_type,
        ];

        $response = Http::withHeaders([
            'Authorization' => 'Basic ' . base64_encode($serverKey . ':'),
            'Content-Type' => 'application/json',
            'Accept' => 'application/json',
        ])->post('https://api.sandbox.midtrans.com/v2/charge', $payload);
    }
}
```

```

if ($response->failed()) {
    Log::error('Midtrans charge failed', ['response' => $response->json()]);
    return response()->json([
        'status' => 'error',
        'status_message' => $response->json()['status_message'] ?? 'Payment failed',
        'data' => $response->json(),
    ], 500);
}

return response()->json([
    'status' => 'success',
    'data' => $response->json(),
]);
}
}

```

➤ payment.blade.php

```

<!DOCTYPE html>
<html lang="id">
<head>
<meta charset="utf-8" />
<title>Pembayaran Midtrans (GoPay & QRIS)</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<meta name="csrf-token" content="{{ csrf_token() }}" />
<style>
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    background: linear-gradient(135deg, #f3f4f6 0%, #e5e7eb 100%);
    margin: 0;
    padding: 0;
    color: #1f2937;
    min-height: 100vh;
    display: flex;
    justify-content: center;
    align-items: center;
}
.card {
    width: 100%;
    max-width: 480px;
    background: #fff;
    border-radius: 16px;
    box-shadow: 0 15px 50px rgba(0,0,0,0.15);
    overflow: hidden;
}
.header {
    background: linear-gradient(135deg, #4b5563 0%, #374151 100%);
    color: white;
    text-align: center;
}

```

```
padding: 25px;
}
.header h1 {
  margin: 0;
  font-size: 26px;
  font-weight: 800;
}
.content {
  padding: 30px;
}
label {
  display: block;
  font-weight: 600;
  margin-bottom: 5px;
}
input {
  width: 100%;
  padding: 10px 12px;
  margin-bottom: 18px;
  border: 2px solid #e5e7eb;
  border-radius: 8px;
  background-color: #f9fafb;
  font-size: 15px;
}
input:focus {
  border-color: #4b5563;
  outline: none;
  background-color: #fff;
  box-shadow: 0 0 0 3px rgba(75, 85, 99, 0.1);
}
.btn {
  width: 100%;
  padding: 14px;
  margin-bottom: 12px;
  border: none;
  border-radius: 10px;
  cursor: pointer;
  font-size: 16px;
  font-weight: 700;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  transition: transform 0.2s, box-shadow 0.2s;
}
.btn:hover {
  transform: translateY(-2px);
  box-shadow: 0 6px 20px rgba(0,0,0,0.15);
}
.btn-gopay {
  background: linear-gradient(135deg, #00b74a 0%, #00e676 100%);
  color: white;
}
```

```
.btn-qris {
    background: linear-gradient(135deg, #3b82f6 0%, #2563eb 100%);
    color: white;
}
.error {
    background-color: #fee2e2;
    border: 2px solid #ef4444;
    padding: 15px;
    border-radius: 8px;
    color: #991b1b;
    font-weight: 600;
    margin-top: 10px;
}
.loading {
    text-align: center;
    padding: 20px;
}
.loading-spinner {
    border: 4px solid rgba(75, 85, 99, 0.2);
    border-top: 4px solid #4b5563;
    border-radius: 50%;
    width: 40px;
    height: 40px;
    animation: spin 1s linear infinite;
    margin: 0 auto 10px;
}
@keyframes spin {
    100% { transform: rotate(360deg); }
}
#result {
    margin-top: 20px;
    text-align: center;
}
.footer {
    background: #f9fafb;
    padding: 15px;
    text-align: center;
    font-size: 12px;
    color: #6b7280;
}

```

</style>

</head>

<body>

<div class="card">

<div class="header">

<h1>Kasir Cafe Galaxy Aromanica</h1>

<p>Sistem Pembayaran GoPay & QRIS</p>

</div>

<div class="content">

```
<label>Nama Lengkap</label>
<input id="name" type="text" placeholder="Masukkan nama lengkap" />

<label>Email</label>
<input id="email" type="email" placeholder="Masukkan email" />

<label>Jumlah (IDR)</label>
<input id="amount" type="number" min="1000" placeholder="Minimal Rp 1.000" />

<button class="btn btn-gopay" onclick="pay('gopay')"><span> GoPay </span> Bayar dengan GoPay</button>
<button class="btn btn-qr" onclick="pay('qr')"><span> QRIS </span> Bayar dengan QRIS</button>

<div id="result"></div>
</div>

<div class="footer">
    © 2025 Kasir Cafe Galaxy Aromanica | Powered by Midtrans
</div>
</div>

<script>
async function pay(type) {
    const name = document.getElementById("name").value.trim();
    const email = document.getElementById("email").value.trim();
    const amount = parseInt(document.getElementById("amount").value);
    const resultDiv = document.getElementById("result");

    if (!name || !email || !amount || amount < 1000) {
        resultDiv.innerHTML = <div class="error">⚠ Mohon isi semua data dengan benar.</div>;
        return;
    }

    resultDiv.innerHTML = `
        <div class="loading">
            <div class="loading-spinner"></div>
            <div>Memproses pembayaran...</div>
        </div>
    `;

    try {
        const res = await fetch(window.location.origin + "/payment/process", {
            method: "POST",
            headers: {
                "Content-Type": "application/json",
                "X-CSRF-TOKEN": document.querySelector('meta[name="csrf-token"]').content,
            },
            body: JSON.stringify({ name, email, amount, payment_type: type }),
        });
    }
}
```

```

const data = await res.json();
console.log("Payment Response:", data);

if (data.status === "success") {
    if (type === "gopay" && data.data.actions) {
        const deeplink = data.data.actions.find(a => a.name === "deeplink-redirect");
        if (deeplink && deeplink.url) {
            window.location.href = deeplink.url;
        } else {
            resultDiv.innerHTML = <div class="error">⚠ Link pembayaran GoPay tidak tersedia.</div>;
        }
    } else if (type === "qrис" && data.data.qr_string) {
        const qrUrl = data.data.qr_string;
        resultDiv.innerHTML = `
            <p><b>QRIS</b> Scan QRIS untuk membayar:</p>
            
            <p style="margin-top:10px; color:#6b7280;">Gunakan aplikasi pembayaran seperti Dana, .</p>
        `;
    } else {
        resultDiv.innerHTML = <div class="error">⚠ Respons tidak dikenali.</div>;
    }
} else {
    resultDiv.innerHTML = <div class="error">⚠ ${data.status_message} || "Gagal memproses pembayaran."</div>;
}

} catch (err) {
    resultDiv.innerHTML = <div class="error">⚠ Gagal terhubung ke server: ${err.message}</div>;
}

</script>
</body>
</html>

```

➤ env.

```

APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:m3wDaK7JxmveOOaAsyOPn6N/klKVCtt2KzZP0H9iDk0=
APP_DEBUG=true
APP_URL=https://sun-disepalous-drew.ngrok-free.dev

APP_LOCALE=en
APP_FALLBACK_LOCALE=en
APP_FAKELOCALE=en_US

```

```
APP_MAINTENANCE_DRIVER=file
# APP_MAINTENANCE_STORE=database

PHP_CLI_SERVER_WORKERS=4

BCRYPT_ROUNDS=12

LOG_CHANNEL=stack
LOG_STACK=single
LOG_DEPRECATED_CHANNEL=null
LOG_LEVEL=debug

DB_CONNECTION=sqlite
# DB_HOST=127.0.0.1
# DB_PORT=3306
# DB_DATABASE=laravel
# DB_USERNAME=root
# DB_PASSWORD=

SESSION_DRIVER=database
SESSION_LIFETIME=120
SESSION_ENCRYPT=false
SESSION_PATH=/
SESSION_DOMAIN=null

BROADCAST_CONNECTION=log
FILESYSTEM_DISK=local
QUEUE_CONNECTION=database

CACHE_STORE=database
# CACHE_PREFIX=

MEMCACHED_HOST=127.0.0.1

REDIS_CLIENT=phpredis
REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_MAILER=log
MAIL_SCHEME=null
MAIL_HOST=127.0.0.1
MAIL_PORT=2525
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_FROM_ADDRESS="hello@example.com"
MAIL_FROM_NAME="${APP_NAME}"

AWS_ACCESS_KEY_ID=
```

```
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=
AWS_USE_PATH_STYLE_ENDPOINT=false

VITE_APP_NAME="${APP_NAME}"
```

```
MIDTRANS_SERVER_KEY=Mid-server-3WDAyaNaTfwKBcCnrTRJ7IBs
MIDTRANS_CLIENT_KEY=Mid-client-YY42AYLfiAeW4ylB
MIDTRANS_IS_PRODUCTION=false
```