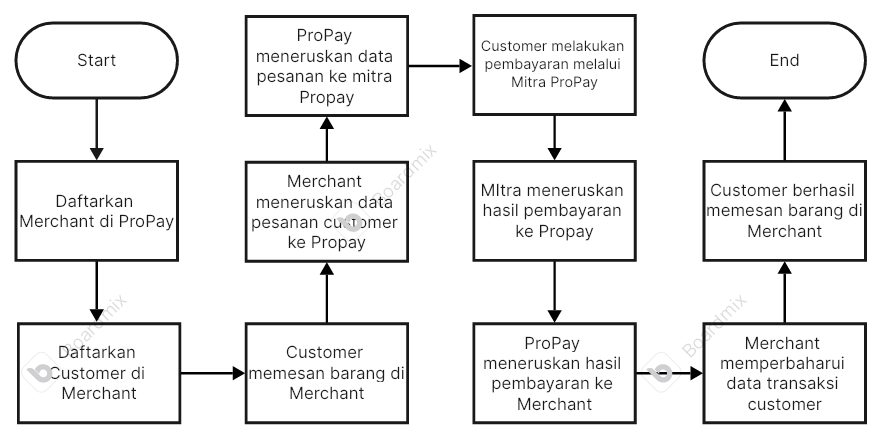
DOKUMENTASI PROYEK PROPAY PAYMENT GATEWAY

1. Pendahuluan
   1. Nama proyek: ProPay (Payment Gateway) dan Merchant
   2. Jumlah proyek spring boot: 2 proyek ( payment gateway dan merchant)
   3. Deskripsi: sebuah payment gateway yang membantu merchant untuk mengelola pembayaran yang dilakukan oleh customernya
2. Arsitektur proyek
   1. Alur proyek:  
      
   2. Arsitektur teknologi:
      1. Java versi 17
      2. Spring Boot versi 3.4.0
      3. Database PostgreSQL
   3. Dependency Spring Boot yang digunakan:
      1. Spring Data JPA: Untuk menghubungkan aplikasi Spring Boot ke Database
      2. PostgreSQL: Database untuk Menyimpan data aplikasi
      3. Spring Webflux: Untuk menghubungkan API antara 2 proyek
      4. Lombok: Untuk memudahkan penulisan kode program
   4. Struktur Proyek:
      1. Propay:

app

- src

- main

- java

- com.bagus.propay\_paymentgateway

- controller

- MerchantController (Endpoint Merchant)

- MerchantRequest (Request registrasi Merchant)

- PaymentCustomerRequest (Request callback pembayaran)

- PaymentRequest (Request informasi pembayaran)

- RegisterRequest (Request callback registrasi)

- VAController (Endpoint Virtual Account)

- VARequest (Request Virtual Account)

- VAResponse (Response informasi pembayaran)

- entity

- Merchant (Model Merchant)

- MerchantRepository (Repository Merchant)

- VA (Model Virtual Account)

- VARepository (Repository Virtual Account)

- service

- MerchantService (Service Merchant)

- VAService (Service Virtual Account)

- PropayPaymentGatewayApplication (Class utama)

- resources

- application.properties (Konfigurasi properti)

- pom.xml (Konfigurasi Spring Boot)

* + 1. Merchant:

app

- src

- main

- jPGW

- com.bagus.merchant\_paymentgateway

- controller

- CustomerController (Endpoint Controller)

- CustomerRequest (Request registrasi Customer)

- PaymentCustomerRequest (Request callback pembayaran)

- PGWController (Endpoint Payment Gateway)

- PGWRequest (Request callback registrasi merchant)

- TransactionController (Endpoint Transaction)

- TransactionRequest (Request Transaction)

- VARequest (Request input Virtual Account)

- entity

- Customer (Model Customer)

- CustomerRepository (Repository Customer)

- PGW (Model Payment Gateway)

- PGWRepository (Repository Payment Gateway)

- Transaction (Model Transaction)

- TrasanctionRepositoy (Repository Transaction)

- service

- CustomerService (Service Customer)

- PGWService (Service Payment Gateway)

- TransactionService (Service Transaction)

- MerchantPaymentGatewayApplication (Class utama)

- resources

- application.properties (Konfigurasi properti)

- pom.xml (Konfigurasi Spring Boot)

1. Persyaratan Sistem
   1. Software:
      1. Java versi 17
      2. Database PostgreSQL
      3. IntelliJ IDEA 2024.2.4 (Community Edition) atau versi Ultimate
      4. Postman untuk pengujian
   2. Konfigurasi:
      1. application.properties:

spring.datasource.url=jdbc:postgresql://localhost:5432/{nama\_database}

spring.datasource.username={username\_postgresql}

spring.datasource.password={password\_postgresql}

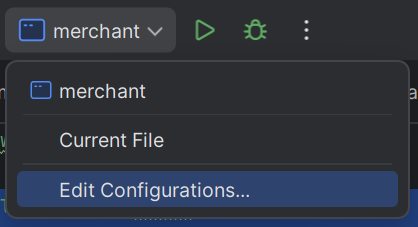
spring.jpa.hibernate.ddl-auto=create-drop

spring.jpa.generate-ddl=true

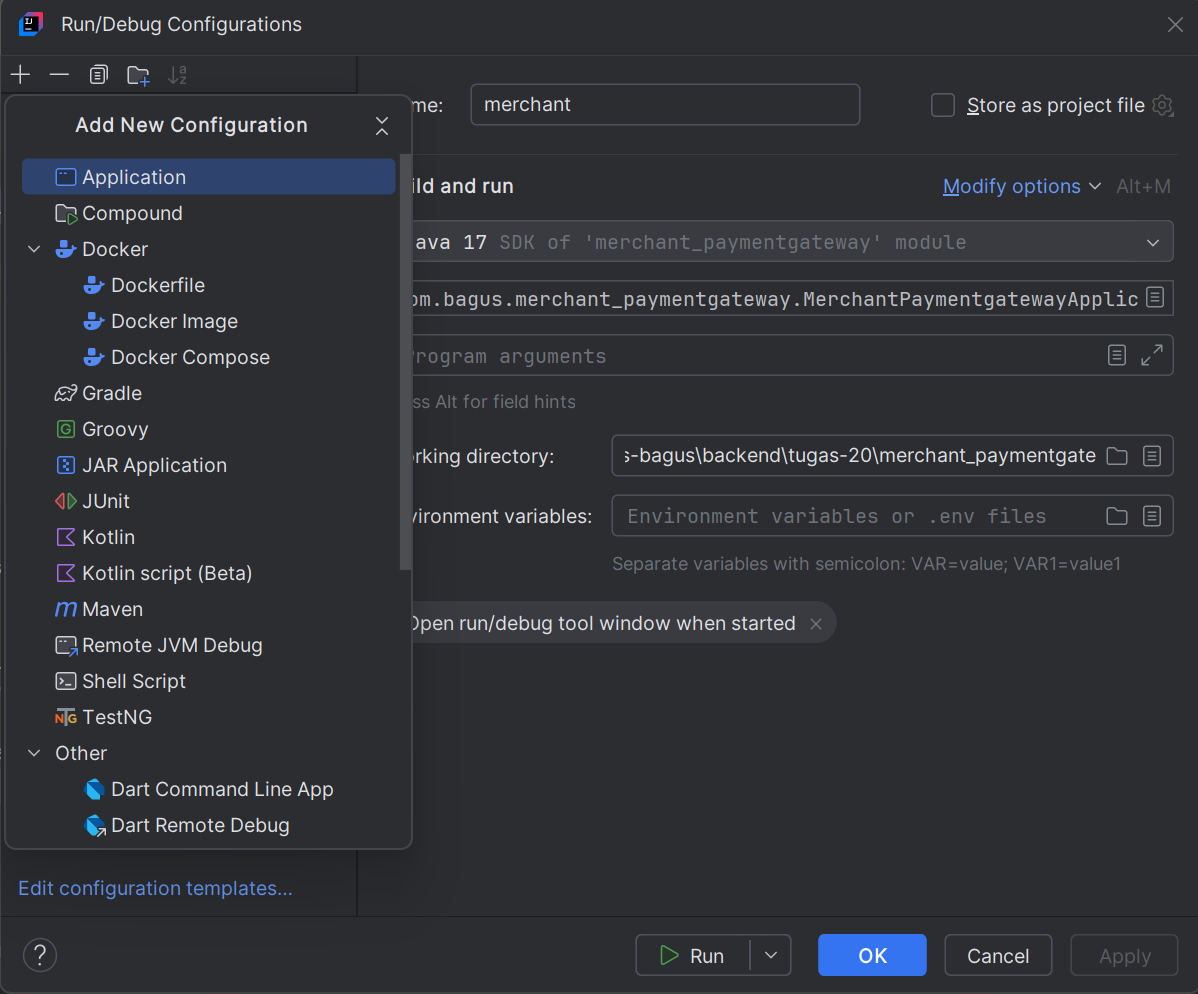
1. Cara Instalasi
   1. Unduh proyek melalui website:

<https://github.com/Prodemy-Gen-24-11/tugas-bagus/tree/master/backend/tugas-20>

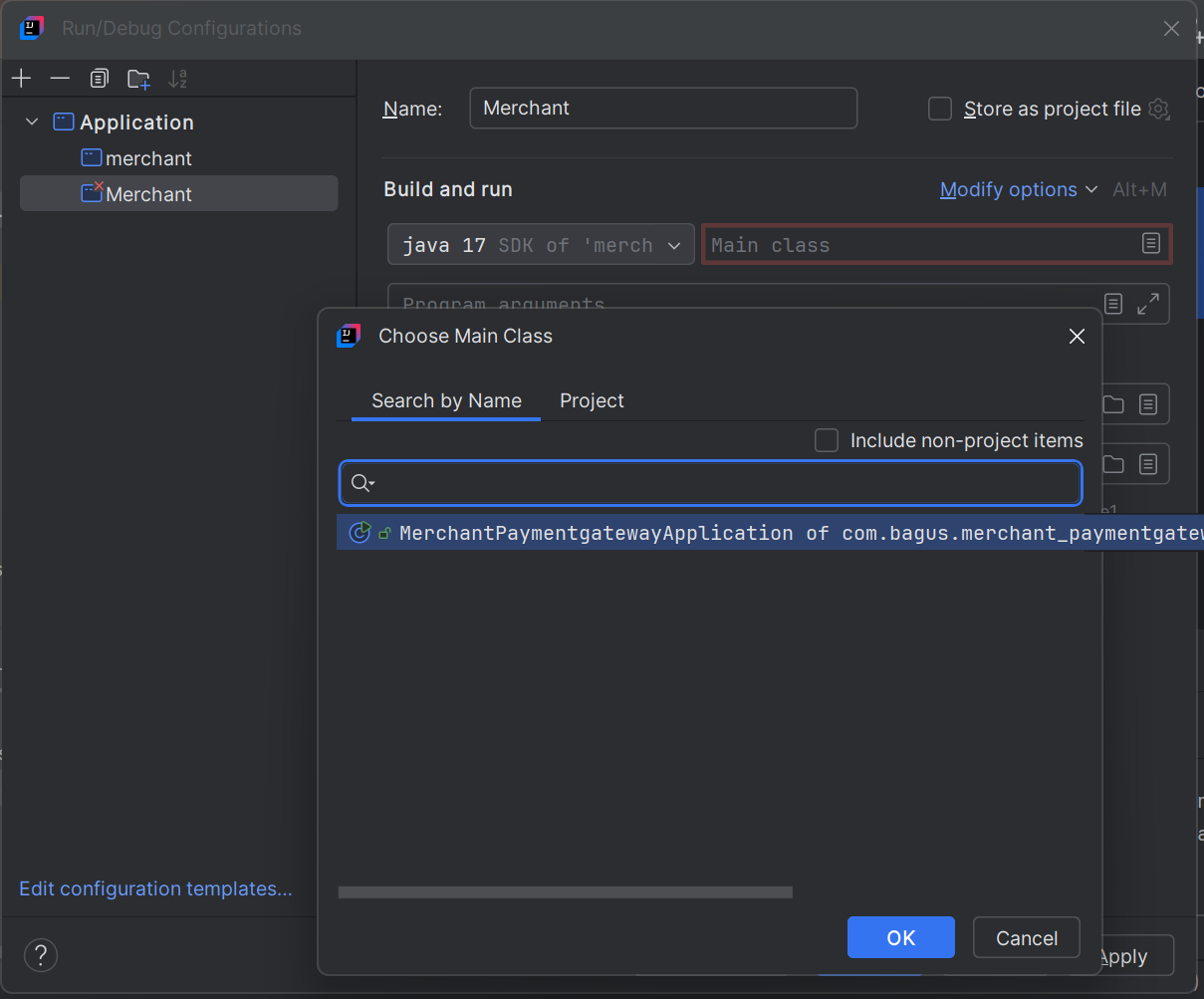
* 1. Buka menggunakan Intellij IDEA
  2. Buat konfigurasi Run/Debug baru dengan cara klik Edit Configuration:



* 1. Klik tanda + di kiri atas window dan pilih Application:



* 1. Ubah nama konfigurasi dan pilih Main Class yang terdeteksi oleh Intellij IDEA:



* 1. Klik Apply dan klik OK
  2. Klik tombol Run untuk menjalankan proyek:



* 1. Jalankan 2 proyek secara bersamaan

1. Penggunaan Aplikasi

Jalankan 2 proyek dan buka postman, berikut ini adalah daftar Endpoint yang dapat dibuka:

* 1. Register Customer
     1. URL : <http://localhost:8082/tokopedia/api/v1/customer/add>
     2. Method : POST
     3. Deskripsi : Register Customer pada endpoint yang dimiliki oleh Merchant
     4. Aksi : Menyimpan data customer di database
     5. Body :

{

"name":"nama\_customer"

}

* + 1. Response :

{

"id": 1,

"name": "nama\_customer"

}

* 1. Register Merchant
     1. URL : <http://localhost:8081/propay/api/v1/merchant/add>
     2. Method : POST
     3. Deskripsi : Register Merchant pada endpoint yang dimiliki oleh ProPay
     4. Aksi : Menyimpan data Merchant di database dan mengirimkan kode

unik Merchant ke endpoint callback milik Merchant. Kemudian pada Merchant akan menyimpan kode unik tersebut untuk membuat virtual account

* + 1. Body :

{

"name":"tokopedia",

"registerCallbackUri":"http://localhost:8082/tokopedia/api/v1/pgw/add"

}

* + 1. Response :

{

"id": 1,

"name": "tokopedia",

"ucode": "3102"

}

* 1. Input Transaction
     1. URL : <http://localhost:8082/tokopedia/api/v1/transaction/add>
     2. Method : POST
     3. Deskripsi : Input transaksi customer pada endpoint milik Merchant
     4. Aksi : Menyimpan data transaksi di database dan mengirimkan

virtual account pada endpoint milik ProPay. Kemudian Propay akan menyimpan data virtual account tersebut

* + 1. Body :

{

"userId":1,

"description":"topi hitam polos",

"amount":10000.0

}

* + 1. Response :

{

"id": 1,

"userId": 1,

"description": "topi hitam polos",

"amount": 10000.0,

"paid": false,

"vaNumber": "31021",

"paymentDate": null,

"createdAt": "2024-12-12T09:08:28.8861974"

}

* 1. Informasi Pembayaran
     1. URL : <http://localhost:8081/propay/api/v1/va/payment>
     2. Method : POST
     3. Deskripsi : Mitra ProPay memanggil endpoint milik ProPay untuk

menginformasikan bahwa pembayaran customer berhasil

* + 1. Aksi : Memperbaharui tanggal pembayaran pada database ProPay.

Kemudian memanggil endpoint callback milik Merchant untuk memberitahu Merchant bahwa customer telah berhasil melakukan pembayaran. Kemudian Merchant akan memperbaharui data transaksi pada database Merchant

* + 1. Body :

{

"vaNumber":"31021"

}

* + 1. Response :

{

"vaNumber": "31021",

"amount": 0.0

}

1. Logika Bisnis:
   1. Alur proyek berdasarkan endpoint:
      1. Merchant mendaftar pada endpoint milik ProPay dengan menyertakan endpoint callback untuk nantinya ProPay akan mengirimkan kode unik milik merchant
      2. Customer mendaftar pada endpoint milik Merchant
      3. Customer membeli produk di merchant dengan melalui endpoint milik Merchant. Kemudian merchant akan membuat virtual account untuk customer lalu menyimpan data transaksi serta memanggil endpoint milik ProPay untuk memasukan virtual account
      4. Customer dapat melakukan pembayaran melalui mitra ProPay. Kemudian mitra ProPay akan memanggil endpoint milik ProPay untuk menginformasikan bahwa customer telah membayar pesanannya. ProPay kemudian akan melakukan update pada data virtual account serta memanggil endpoint callback milik Merchant. Kemudian merchant akan melakukan update pada data transaksi dari customer
   2. Pembuatan kode unik Merchant

Propay akan membuat kode unik dari 4 nomor acak dengan menggunakan library Random, dengan sintaks String.format("%04d", new Random().nextInt(10000));

* 1. Pembuatan virtual account:

Virtual account merupakan gabungan dari kode unik milik Merchant yang didapatkan saat melakukan regristrasi pada endpoint ProPay ditambah dengan id dari customer. Misalnya kode unik Merchant adalah “1231” dan id dari customer adalah “1”, maka virtual account yang akan dihasilkan adalah “12311”.