Tugas Praktikum X

(Kelompok 8)

- M. Naufal Muafa (1301180091)
- Inggrid Resmi Benita (1301184107)
- Saskia Putri Ananda (1301184157)
- Anis Novitasari (1301184424)

TUGAS LAB

Source code yang baik adalah source code yang tidak hanya programmer pembuatnya yang bisa membaca. Semua programmer lain bisa membacanya.
Untuk itu setiap source code perlu diberi catatan/komentar (*):
- Pendahuluan : tujuan code, deskripsi singkat, rujukan dokumen DPPL
- Penjelasan data yang diolah
- Penjelasan setiap method
- Penjelasan line code yang berisi rumus/inti pemrosesan

Manfaatkan :
- slide kuliah
- hasil browsing internet terkait pembuatan dokumentasi code Java 'Javadoc'
- hasil browsing internet terkait 'Natural Docs' di web http://www.naturaldocs.org
- hasil browsing internet terkait dokumentasi bahasa pemrograman yang dipakai setiap kelompok

Buatlah : Dokumentasi Code dari code yang sudah berhasil dibuat selama ini di Lab,
dimana setiap anggota kelompok membuat satu dokumen Source Code.

Berikut ini method-method yang ada pada model Order

Method getUserRoleForEbookInfoPage($bookId)

```php
// Method ini digunakan untuk mengembalikan status user dalam halaman ebook_info
// @param $bookId : id buku
// @return : status user (seperti enum)
public static function getUserRoleForEbookInfoPage($bookId)
{
    // Status 1 : Sebagai publisher
    // Status 2 : Sebagai buyer yang blm membeli buku
    // Status 3 : Sebagai buyer yang sedang atau sudah membeli buku

    // mengambil user id dari session
    $userId = session('id');

    // mengambil id publisher dari id buku
    $publisherId = Book::getPublisherIdByBookId($bookId);

    // mengambil id user dari id publisher
    $publisherUserId = Publisher::getUserIdByPublisherId($publisherId);

    if ($publisherUserId == $userId) { // Jika publisher sedang membuka halaman ebook_info
        return 1;
    }
    else if (Order::whetherTheBuyerHasntPurchasedBook($userId, $bookId)) { // Jika buyer belum membeli buku
        return 2;
    }
    return 3; // jika buyer sedang atau sudah membeli buku
}
```

Method whetherTheBuyerHasntPurchasedBook($buyerId, $bookId)

```php
// Method ini digunakan untuk mengembalikan boolean apakah buyer belum membeli buku
// @param $buyerId : id buyer
// @param $bookId : id buku
// @return : boolean apakah buyer belum membeli buku
private static function whetherTheBuyerHasntPurchasedBook($buyerId, $bookId)
{
    if (Order::whetherTheBuyerHasAlreadyPurchasedBook($buyerId, $bookId) || Order::whetherTheBuyerIsBuyingBook($buyerId, $bookId)) {
        return false;
    }
    return true;
}
```

Method whetherTheBuyerHasAlreadyPurchasedBook($buyerId, $bookId)

```php
// Method ini digunakan untuk mengembalikan boolean apakah buyer sudah membeli buku
// @param $buyerId : id buyer
// @param $bookId  : id buku
// @return : boolean apakah buyer sudah membeli buku
private static function whetherTheBuyerHasAlreadyPurchasedBook($buyerId, $bookId)
{
    $count = DB::table('orders')
                    ->join('users', 'users.id', '=', 'orders.userId')
                    ->join('book_snapshots', 'book_snapshots.orderId', '=', 'orders.id')
                    ->where('users.id', $buyerId)
                    ->where('book_snapshots.bookId', $bookId)
                    ->where('status', 'success')
                    ->count(); // mendapatkan data banyak order yang sesuai dengan parameter
    if ($count != 0) { // jika tidak ada data
        return true;
    }
    return false; // jika ada data
}
```

Method whetherTheBuyerIsBuyingBook($buyerId, $bookId)

```php
// Method ini digunakan untuk mengembalikan boolean apakah buyer sedang membeli buku
// @param $buyerId : id buyer
// @param $bookId  : id buku
// @return : boolean apakah buyer sedang membeli buku
private static function whetherTheBuyerIsBuyingBook($buyerId, $bookId)
{
    $count = DB::table('orders')
                    ->join('users', 'users.id', '=', 'orders.userId')
                    ->join('book_snapshots', 'book_snapshots.orderId', '=', 'orders.id')
                    ->where('users.id', $buyerId)
                    ->where('book_snapshots.bookId', $bookId)
                    ->where('status', 'pending') // yang status ordernya 'pending'
                    ->count();  // mendapatkan data banyak order yang sesuai dengan parameter
    if ($count != 0) { // jika tidak ada data
        return true;
    }
    return false; // jika ada data
}
```

Method convertPriceToCurrencyFormat($price)

```php
// Method ini digunakan untuk mengonversi integer ke format rupiah
// @param $price : harga (integer)
// @return : string rupiah hasil konversi
private static function convertPriceToCurrencyFormat($price)
{
    return number_format($price,0,',','.');
}
```

Method getNewOrderId()

```php
// Method ini digunakan untuk membuat id order baru
// id order akan digunakan untuk menyimpan order baru
// @param $price : harga (integer)
// @return : string rupiah hasil konversi
private static function getNewOrderId()
{
    return DB::table('orders')->count() + 1;
}
```

Method createOrder($paymentMethod)

```php
// Method ini digunakan untuk membuat dan menyimpan order baru
// @param $paymentMethod : metode pembayaran (integer);
// @return : id order yang telah dibuat
public static function createOrder($paymentMethod)
{
    $createdAt = Carbon::now(); // menyimpan current time
    $dt = $createdAt->copy()->addHours(24); // menyimpan waktu tenggat pembayaran (24 jam dari current time)
    $dt->second = 0; // Membulatkan waktu tenggat
    $faker = Faker::create('id_ID'); // Membuat objek faker yang berhubungan dengan country indonesia
    $backCode = $faker->swiftBicNumber; // Membuat back code dari faker
    $orderId = Order::getNewOrderId(); // menyimpan id order baru
    $midtransOrderId = $orderId."-".$backCode; // menyimpan id order untuk dikirimkan ke PG
    $arrBookId = Cart::getUserCartBookId(); // mendapatkan id-id buku dalam keranjang belanja user
    $totalPrice = 0;  // inisialisasi total harga
    foreach ($arrBookId as $bookId) { // menghitung total harga
        $totalPrice += Book::getPrice($bookId->bookId);
    }
    $paymentCode = Order::getPaymentCode($paymentMethod, $orderId); // mendapatkan kode pembaran
    Cart::emptyUserCart(); // mengosongkan keranjang belanja user
    if ($paymentMethod == "1") { // Jika metode pembayarannya mengunakan BNI VA
        Order::postTransactionToMidtransWithBNIVAPayment($midtransOrderId, $totalPrice); // mengirim data order ke PG
    }
    else if ($paymentMethod == "2") { // Jika metode pembayarannya menggunakan indomaret      You, seconds ago • Uncommitted changes
        Order::postTransactionToMidtransWithIndomaretPayment($midtransOrderId, $totalPrice); // mengirim data order ke PG
    }
    Order::store($orderId, $backCode, $paymentMethod, $paymentCode, $dt, $createdAt); // menyimpan data order ke database
    BookSnapshot::storeBookSnaphshotsByArrBookIdAndOrderId($arrBookId, $orderId); // menyimpan data ebook ke tabel 'book snapshot'
    return $orderId;
}
```

Method updatePaymentCodeFromMidtrans($orderId)

```php
// Method ini digunakan untuk memperbarui payment code dari sebuah order
// @param $orderId : id order (integer);
public static function updatePaymentCodeFromMidtrans($orderId)
{
    $backCode = DB::table('orders')->where('id', $orderId)->pluck('backIdCode')[0]; // mendapatkan back code
    $midtransOrderId = $orderId."-".$backCode; // menyimpan order id yang ada di PG
    $paymentMethodId = DB::table('orders')->where('id', $orderId)->pluck('paymentId')[0]; // mendapatkan data paymentId
    $paymentCode = Order::getPaymentCodeFromMidtrans($midtransOrderId, $paymentMethodId); // mendapatkan data paymentCode dari PG
    DB::table('orders')->where('id', $orderId)->update([ // update payment code ke row order
        "paymentCode" => $paymentCode,
        "updated_at" => Carbon::now(),
    ]);
}
```

Method store($id, $backCode , $paymentId, $paymentCode, $expiredTime, $createdAt)

```php
// Method ini digunakan untuk memperbarui payment code dari sebuah order
// @param $id          : id order (integer);
// @param $backCode    : backCode dari order (string);
// @param $paymentId   : metode pembayaran (integer);
// @param $paymentCode : kode pembaayaran (string);
// @param $expiredTime : waktu tenggat pembayaran (timestamp);
// @param $createdAt   : waktu order dibuat (timestamp);
private static function store($id, $backCode , $paymentId, $paymentCode, $expiredTime, $createdAt)
{
    $userId = session('id'); // mengambil id user dari session
    DB::table('orders')->insert([ // memasukkan data order ke database      You, seconds ago • Uncommitted c
        "id" => $id,
        "paymentId" => $paymentId,
        "userId" => $userId,
        "backIdCode" => $backCode,
        "paymentCode" => $paymentCode,
        "expiredTime" => $expiredTime,
        "created_at" => $createdAt,
        "updated_at" => $createdAt,
    ]);
}
```

Method postTransactionToMidtransWithBNIVAPayment($midtransOrderId, $totalAmount)

```php
// Method ini digunakan untuk mengirim data order ke PG (Payment Gateway) dengan metode pembarannya menggunakan BNI VA
// @param $midtransOrderId : id order yang akan digunakan di PG (string);
// @param $totalAmount     : total dana dalam transaksi (integer);
public static function postTransactionToMidtransWithBNIVAPayment($midtransOrderId, $totalAmount)
{
    $curl = curl_init(); // inisiasi curl

    curl_setopt_array($curl, array(
    CURLOPT_URL => "https://api.sandbox.midtrans.com/v2/charge",
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_ENCODING => "",
    CURLOPT_MAXREDIRS => 10,
    CURLOPT_TIMEOUT => 0,
    CURLOPT_FOLLOWLOCATION => true,
    CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
    CURLOPT_CUSTOMREQUEST => "POST",
    CURLOPT_POSTFIELDS => "{ \"payment_type\": \"bank_transfer\", \"transaction_details\": {\"gross_amount\": {$totalAmount}, \"ord
    CURLOPT_HTTPHEADER => array(
        "Accept: application/json",
        "Content-Type: application/json",
        env("MIDTRANS_AUTHORIZATION")
    ),
    )); // mengirim data transaksi

    $response = curl_exec($curl);

    curl_close($curl); // menutup curl
    // echo $response;      You, 10 months ago • Redirect ke info pesanan ketika selesai melakukan…
}
```

Method postTransactionToMidtransWithIndomaretPayment($midtransOrderId, $totalAmount)

```php
// Method ini digunakan untuk mengirim data order ke PG (Payment Gateway) dengan metode pembarannya menggunakan Indomaret
// @param $midtransOrderId : id order yang akan digunakan di PG (string);
// @param $totalAmount     : total dana dalam transaksi (integer);
public static function postTransactionToMidtransWithIndomaretPayment($midtransOrderId, $totalAmount)
{
    $curl = curl_init(); // inisiasi curl

    curl_setopt_array($curl, array(
    CURLOPT_URL => "https://api.sandbox.midtrans.com/v2/charge",
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_ENCODING => "",
    CURLOPT_MAXREDIRS => 10,
    CURLOPT_TIMEOUT => 0,
    CURLOPT_FOLLOWLOCATION => true,
    CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
    CURLOPT_CUSTOMREQUEST => "POST",
    CURLOPT_POSTFIELDS =>"{
        \"payment_type\": \"cstore\",
        \"transaction_details\": {
            \"gross_amount\": {$totalAmount},
            \"order_id\": \"{$midtransOrderId}\"
        },
        \"customer_details\": {
            \"email\": \"noreply@example.com\",
            \"first_name\": \"Vervays\",
            \"last_name\": \"user\",
            \"phone\": \"+6281 1234 1234\"
        },
        \"item_details\": [
        {
           \"id\": \"item01\",
           \"price\": {$totalAmount},
           \"quantity\": 1,
           \"name\": \"Ebook1\"
        }
       ],
      \"cstore\": {
        \"store\": \"Indomaret\",
        \"message\": \"Message to display\"
      }
    }",
    CURLOPT_HTTPHEADER => array(
        "Accept: application/json",
        "Content-Type: application/json",
        env("MIDTRANS_AUTHORIZATION")
    ),
    )); // mengirim data transaksi

    $response = curl_exec($curl);

    curl_close($curl); // menutup curl      You, seconds ago • Uncommitted changes
    // echo $response;
}
```

Method getPaymentCode($paymentId, $orderId)

```php
// Method ini digunakan untuk membuat payment code berdasarkan parameter
// @param $paymentId : id pembayaran (integer);
// @param $orderId   : id order (integer);
// @return kode pembayaran
private static function getPaymentCode($paymentId, $orderId)
{
    if ($paymentId == 1) { // jika metode pembayarannya menggunakan BNI VA
        return "21".$orderId;
    }
    else if ($paymentId == 2) { // jika metode pembayarannya menggunakan indomaret
        return "22".$orderId;
    }
    else {
        return "23".$orderId;
    }
}
```

Method whetherTheTransactionIsPendingOrSuccess($bookId)

```php
// Method ini mengembalikan status transaksi
// @param $bookId : id buku (integer);
public static function whetherTheTransactionIsPendingOrSuccess($bookId)
{
    $userId = session('id');
    if (Order::whetherTheBuyerHasAlreadyPurchasedBook($userId, $bookId)) {
        return "success";
    }
    return "pending";
}
```

Method getRealStatus($orderId)

```php
// Method ini mengembalikan status transaksi
// Status transaksi didapatkan dari PG
// Method ini diapanggil di middleware
// @param $orderId : id order (integer);
public static function getRealStatus($orderId)
{
    $curl = curl_init(); // inisiasi curl

    curl_setopt_array($curl, array(
    CURLOPT_URL => "http://localhost:8000/transaction/$orderId",
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_ENCODING => "",
    CURLOPT_MAXREDIRS => 10,
    CURLOPT_TIMEOUT => 0,
    CURLOPT_FOLLOWLOCATION => true,
    CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
    CURLOPT_CUSTOMREQUEST => "GET",
    CURLOPT_POSTFIELDS =>"{\r\n    \"paymentId\" : 1,\r\n    \"paymentCode\" : 213\r\n}",
    CURLOPT_HTTPHEADER => array(
        "Content-Type: application/json"
    ),
    )); // Membuat request

    $response = curl_exec($curl); // Mendapatkan hasil request

    curl_close($curl); // menutup curl
    return json_decode($response, true)["status"]; //mengembalikan status order
}
```

Method updateStatus()

```php
// Method ini memperbarui status transaksi
// Status transaksi didapatkan dari PG
// Method ini diapanggil di middleware
public static function updateStatus()
{
    // Mendapatkan data order yang masih pending (dengan user yg sesuai)
    $orders = DB::table('orders')->where('status', 'pending')->where('userId', session('id'))->get();       You, 10 months a

    foreach ($orders as $order) { // untuk setiap order
        $midtransOrderId = $order->id."-".$order->backIdCode; // menyimpan order id
        $transactionStatus = Order::getTransactionStatusFromMidtrans($midtransOrderId); // menyimpan status transaksi
        if($transactionStatus == "settlement") { // jika transaksinya berhasil
            DB::table('orders')->where('id', $order->id)->update([ // update status transaksi mnjd 'success'
                "status" => "success",
                "updated_at" => Carbon::now(),
            ]);
            $arrBookId = DB::table('orders')
                                ->join('book_snapshots', 'orders.id', '=', 'book_snapshots.orderId')
                                ->where('orders.id', $order->id)
                                ->pluck('book_snapshots.bookId'); // mendapatkan id-id buku yang diorder
            foreach ($arrBookId as $bookId) { // untuk setiap id buku
                $publisherId = Book::getPublisherIdByBookId($bookId); // mendapatkan id publisher dari id buku
                $price = BookSnapshot::getPrice($bookId, $order->id); // mendapatkan harga buku
                Publisher::addBalance($publisherId, $price); // menambah saldo publisher sesuai harga buku
                Have::store(session('id'), $bookId); // menyimpan buku di tabel 'have' agar user dapat membaca buku
            }
        }
        else if($transactionStatus == "cancel" || $transactionStatus == "expire") { // jika transaksinya gagal
            DB::table('orders')->where('id', $order->id)->update([ // update status transaksi mnjd 'failed'
                "status" => "failed",
                "updated_at" => Carbon::now(),
            ]);
        }
    }
}
```

Model Order digunakan untuk mengambil, menghapus, menambah, dan memperbarui data pada tabel 'order'.

Link repo : https://github.com/Kelompok8RPLIF4203/modul10

#Selamat bekerja

RUBRIKASI
A : 81..100 : Dokumen Source Code lengkap (> 80%) dengan penjelasan (*) sejumlah anggota kelompok
AB : 71..80 : Dokumen Source Code lengkap (antara 71..80%) dengan penjelasan (*) sejumlah anggota kelompok
B : 66..70 : Dokumen Source Code lengkap (antara 66..70%) dengan penjelasan (*) sejumlah anggota kelompok
BC : 61..65 : Dokumen Source Code lengkap (antara 61..65%) dengan penjelasan (*) sejumlah anggota kelompok
C : 51..60 : Dokumen Source Code lengkap (antara 51..60%) dengan penjelasan (*) sejumlah anggota kelompok
D : 41..50 : Dokumen Source Code lengkap dengan penjelasan (*) anggota kelompok tidak lengkap
E : 0..40 : tidak hadir