

## A. Identitas Job Sheet

### IDENTITAS JOB SHEET

Perguruan Tinggi	: Politeknik Negeri Bengkalis	Pertemuan Ke	:
Jurusan/Program Studi	: Teknik Informatika	Job Ke	: 2
Kode Mata Kuliah	:	Halaman	: 28
Nama Mata Kuliah	: Pemograman Framework	Mulai Berlaku	:

## B. Komponen Job Sheet

### 2.1. Judul Job

### JOB II

### Pengenalan Controller, Middleware, Model, Migration, Route dan GitHub

### 2.2. Tujuan

1. Mahasiswa mampu mengenal apa itu controller, middleware, model, migration, route, dan github.
2. Mahasiswa mampu membuat controller, model, migration, dan route.
3. Mahasiswa mampu memebuat repositori github dan melakukan push terhadap perubahan kode yang dilakukan.
4. Mahasiswa dapat memahami bagaimana controller, middleware, model, migration, dan route bekerja dalam Laravel.

### 2.3. Dasar Teori

#### 2.3.1. Controller

Controller di dalam laravel, merupakan sebuah tempat dimana logika dari aplikasi laravel sendiri diletakkan. Controller biasanya digunakan untuk mengatur atau memproses alur kerja seperti menerima permintaan (request) dari user, meresponse data, dan mengirimkan balasan (response) tersebut. Sebagai contoh, jika user mengunjungi halaman tertentu dalam situs website anda, maka controller akan memutuskan apa yang akan dikerjakan seperti menampilkan halaman website, memproses form, dan lain sebagainya.

Untuk membuat sebuah controller dalam laravel, Anda bisa menggunakan perintah “**php artisan make:controller NamaController**” dalam terminal.

### 2.3.2. Middleware

Middleware dalam laravel merupakan sebuah lapisan pengaman atau filter yang dapat digunakan untuk memproses sebuah request dari user yang masuk ke dalam aplikasi. Anda bisa menggunakan middleware tersebut untuk memastikan bahwa hanya user tertentu yang dapat mengakses halaman tertentu. Sebagai contoh, user yang sudah melakukan login ke dalam halaman admin tidak bisa masuk ke dalam halaman user biasa). Middleware ini biasanya digunakan untuk hal-hal seperti otentikasi, pengaturan hak akses untuk masing-masing user, validasi, dan hal lainnya. Untuk membuat middleware dalam laravel, Anda bisa menggunakan perintah ”**php artisan make:middleware NamaMiddleware**” dalam terminal.

### 2.3.3. Model

Model didalam laravel merupakan sebuah representasi dari tabel yang ada di dalam database. Di dalam laravel sendiri, model digunakan untuk melakukan interaksi dengan tabel dan data yang ada di dalamnya. Setiap model biasanya terhubung dengan satu tabel dan melalui model ini anda bisa melakukan operasi database seperti mengambil data, menyimpan data baru, memperbarui data, atau bahkan melakukan penghapusan data. Laravel sendiri menggunakan Eloquent ORM (Object-Relational Mapping) untuk memudahkan dalam mengelola sebuah data. Dengan Eloquent ORM tersebut, anda bisa mengakses data di database seperti mengakses sebuah properti objek. Untuk membuat model dalam laravel, Anda bisa menggunakan perintah “**php artisan make:model NamaModel**” dalam terminal.

### 2.3.4. Migration

Migration di dalam laravel, adalah sebuah fitur yang dapat digunakan untuk mengelola perubahan struktur database secara terorganisir. Dengan migration ini, anda bisa membuat, mengubah, atau menghapus tabel dan kolom di database tanpa perlu menulis langsung atau berinteraksi langsung dengan SQL. Migration sendiri membantu anda untuk menjaga versi database agar tetap konsisten antara tim pengembang (dalam kerja tim) dan mudah untuk melakukan sebuah rollback (membatalkan perubahan) jika diperlukan. Untuk

membuat migration dalam laravel, Anda bisa menggunakan perintah “**php artisan make:migration nama\_migration**” dalam terminal.

### 2.3.5. Route

Route di dalam laravel, adalah sebuah cara laravel untuk mengarahkan URL ke controller atau action yang akan dijalankan. Ketika user mengakses URL tertentu, maka route akan memetakan URL tersebut ke controller atau fungsi yang sesuai. Sebagai contoh, ketika user mencoba untuk mengunjungi URL "/profile/1", maka route akan memberitahu aplikasi untuk menjalankan method "show()" dari controller dan menampilkan profil user dengan nomor ID adalah 1.

Dari penjelasan diatas, dapat disimpulkan bahwa Route adalah penghubung URL dengan Controller, sementara Middleware bekerja di belakang layar untuk memproses request sebelum atau sesudah controller dijalankan. Selain itu, Model digunakan untuk berinteraksi dengan data di tabel database, sementara Migration digunakan untuk mengatur struktur tabel tersebut. Keduanya bekerja sama untuk mempermudah pengelolaan data dan perubahan database dalam aplikasi laravel. Anda dapat membuat model dan migration secara bersamaan, caranya adalah anda perlu menggunakan perintah “**php artisan make:model NamaModel -m**”, maka laravel akan membuat sebuah file model dan migrationnya sehingga Anda tidak perlu membuat model dan migration secara satu persatu.

### 2.3.6. GitHub

GitHub adalah sebuah platform berbasis web yang digunakan untuk menyimpan dan mengelola kode menggunakan Git. Selain itu, GitHub memungkinkan kolaborasi antar tim pengembangan dengan fitur-fitur seperti version control, tracking perubahan, dan berbagi kode secara aman. Git adalah sebuah sistem version control yang memungkinkan anda untuk melacak perubahan pada kode secara lokal di perangkat anda. Dengan Git, kamu bisa membuat versi kode yang berbeda, kemudian kembali kepada versi sebelumnya, dan bekerja sama dengan orang lain tanpa khawatir akan konflik kode.

## 2.4. Alat dan Bahan

Alat dan bahan yang diperlukan dalam praktikum ini adalah:

1. Komputer/laptop dengan minimal versi windows 7.
2. Code Editor Visual Studio Code.
3. XAMPP.

#### 4. Browser.

### 2.5. Implementasi

Selanjutnya, Anda akan membuat sebuah **model** dan **migration** untuk **Product** serta menampilkan data Product tersebut ke dalam halaman user biasa dan admin. Untuk membuat model dan migration dari Product, ketikkan perintah dibawah ini dalam terminal Anda:

```
php artisan make:model Product -m
```

Setelah membuat model dan migration Product, Anda perlu menambahkan beberapa baris kode ke dalam model **Product.php** dan file migration **2024\_09\_18\_123105\_create\_products\_table.php** (sesuaikan dengan nama file migration dalam project Anda) dengan kode yang ada dibawah ini:

#### Product.php:

```
1 namespace App\Models;
2
3 use Illuminate\Database\Eloquent\Factories\HasFactory;
4 use Illuminate\Database\Eloquent\Model;
5
6 class Product extends Model
7 {
8     use HasFactory;
9
10    protected $fillable = [
11        'name', 'price', 'category', 'description', 'image'
12    ];
13 }
```

#### 2024\_09\_18\_123105\_create\_products\_table.php:

```
1 use Illuminate\Database\Migrations\Migration;
2 use Illuminate\Database\Schema\Blueprint;
3 use Illuminate\Support\Facades\Schema;
4
5 return new class extends Migration
6 {
7     /**
8      * Run the migrations.
9      */
10    public function up(): void
11    {
12        Schema::create('products', function (Blueprint $table) {
13            $table->id();
14            $table->string('name');
15            $table->bigInteger('price');
16            $table->string('category');
17            $table->text('description');
18            $table->string('image');
19            $table->timestamps();
20        });
21    }
22
23    /**
24     * Reverse the migrations.
25     */
26    public function down(): void
27    {
28        Schema::dropIfExists('products');
29    }
30};
```

Selanjutnya, Anda akan menambahkan controller baru dengan nama AdminController untuk mengatur fungsi-fungsi yang akan digunakan oleh admin. Untuk membuat controller tersebut, ketikkan perintah dibawah ini pada terminal Anda:

```
php artisan make:controller Admin/AdminController
```

Setelah berhasil menambahkan **AdminController.php**, tulislah kode dibawah ini dalam controller tersebut yang berlokasi di dalam **app/Http/Controllers/Admin/AdminController.php**:

```
 1 namespace App\Http\Controllers\Admin;
 2
 3 use App\Http\Controllers\Controller;
 4 use Illuminate\Http\Request;
 5 use App\Models\Product;
 6 use App\Models\User;
 7
 8 class AdminController extends Controller
 9 {
10     public function dashboard()
11     {
12         $products = Product::count();
13         $users = User::count();
14
15         return view('pages.admin.index', compact('products', 'users'));
16     }
17 }
```

Setelah membuat **AdminController.php**, selanjutnya Anda akan membuat controller untuk admin dapat mengelola data Product dengan nama **ProductController.php**. Untuk membuat controller tersebut, ketikkan perintah dibawah ini dalam terminal Anda:

```
php artisan make:controller Admin/ProductController
```

Langkah selanjutnya, tulislah kode dibawah ini dalam **ProductController.php** yang berlokasi di dalam **app/Http/Controllers/Admin/ProductController.php**:

```
 1 namespace App\Http\Controllers\Admin;
 2
 3 use App\Http\Controllers\Controller;
 4 use Illuminate\Http\Request;
 5 use App\Models\Product;
 6
 7 class ProductController extends Controller
 8 {
 9     public function index()
10     {
11         $products = Product::all();
12
13         return view('pages.admin.product.index', compact('products'));
14     }
15 }
```

Untuk langkah selanjutnya, Anda akan mengubah tampilan admin dan menambahkan view baru bagi admin dan user biasa untuk menampilkan data produk dari database. Pertama, Anda akan fokus dalam bagian admin dulu. Tambahkan kode yang diberi tanda warna kuning dibawah ini, pada bagian **sidebar.blade.php** yang berlokasi di dalam **resource/views/layouts/admin/sidebar.blade.php**: [tulis sintak dengan rapih]



```
1 <div class="main-sidebar sidebar-style-2">
2     <aside id="sidebar-wrapper">
3         <div class="sidebar-brand">
4             <a href="#">Teknik Informatika | RPL</a>
5         </div>
6         <div class="sidebar-brand sidebar-brand-sm">
7             <a href="#">RPL</a>
8         </div>
9         <ul class="sidebar-menu">
10            <li class="menu-header">Menu</li>
11            <li class="{{ Route::is('admin.dashboard') ? 'active' : '' }}"><a class="nav-link"
12                href="{{ route('admin.dashboard') }}><i class="fas fa-home"></i> <span>Dashboard</span></a>
13                </li>
14                <li class="{{ Route::is('admin.product') ? 'active' : '' }}><a class="nav-link"
15                    href="{{ route('admin.product') }}><i class="fas fa-box"></i> <span>Produk</span></a></li>
16            </ul>
17        </aside>
18    </div>
```

Selanjutnya, Anda akan menambahkan file untuk halaman index dari produk di dalam bagian admin. Sebelum menambahkan file tersebut, terlebih dahulu buatlah sebuah folder baru dengan nama **product** di dalam **resources/views/pages/admin**, kemudian buatlah sebuah file baru dengan nama **index.php** di dalam **resources/views/pages/admin/product/index.blade.php** lalu tulislah kode dibawah ini:

```
@extends('layouts.admin.main')
@section('title', 'Admin Product')
@section('content')
<div class="main-content">
    <section class="section">
        <div class="section-header">
            <h1>Produk</h1>
            <div class="section-header-breadcrumb">
                <div class="breadcrumb-item active"><a href="{{ route('admin.dashboard') }}>Dashboard</a></div>
```

```

        <div class="breadcrumb-item">Produk</div>
    </div>
</div>

<a href="#" class="btn btn-icon icon-left btn-primary"><i class="fas fa-plus"></i> Produk</a>

<div class="card-body">
    <div class="table-responsive">
        <table class="table table-bordered table-md">
            <tr>
                <th>#</th>
                <th>Nama Produk</th>
                <th>Harga Produk</th>
                <th>Stok</th>
                <th>Action</th>
            </tr>
            @php
                $no = 0
            @endphp
            @forelse ($products as $item)
                <tr>
                    <td>{{ $item->name }}</td>
                    <td>{{ $item->price }} Points</td>
                    <td>{{ $item->stock }}</td>
                    <td>
                        <a href="#" class="badge badge-info">Detail</a>
                        <a href="#" class="badge badge-warning"> Edit </a>
                        <a href="" class="badge badge-danger"> Hapus </a>
                    </td>
                </tr>
            @empty
                <td colspan="5" class="text-center">Data Produk Kosong</td>
            @endforelse
        </table>
    </div>
</div>
</div>
@endsection

```

Setelah itu, Anda juga perlu mengubah halaman dashboard admin atau halaman pertama yang ditampilkan ketika admin berhasil melakukan login ke dalam website. Untuk mengubah halaman dashboard admin tersebut, Anda perlu masuk ke dalam file **index.blade.php** yang berlokasi di dalam **resources/views/pages/admin/index.blade.php** kemudian Ubahlah kode sebelumnya menjadi kode yang ada dibawah ini yang diberi dengan tanda warna kuning:

```

@extends('layouts.admin.main')
@section('title', 'Admin Dashboard')
@section('content')
    <div class="main-content">
        <section class="section">
            <div class="section-header">
                <h1>Dashboard</h1>

```

```

<div class="section-header-breadcrumb">
    <div class="breadcrumb-item active"><a href="#">Dashboard</a></div>
</div>

<div class="row">
    <div class="col-lg-3 col-md-6 col-sm-6 col-12">
        <div class="card card-statistic-1">
            <div class="card-icon bg-primary">
                <i class="far fa-user"></i>
            </div>
            <div class="card-wrap">
                <div class="card-header">
                    <h4>Total Pengguna</h4>
                </div>
                <div class="card-body">
                    {{ $users }}
                </div>
            </div>
        </div>
    </div>
    <div class="col-lg-3 col-md-6 col-sm-6 col-12">
        <div class="card card-statistic-1">
            <div class="card-icon bg-danger">
                <i class="far fa-newspaper"></i>
            </div>
            <div class="card-wrap">
                <div class="card-header">
                    <h4>Total Produk</h4>
                </div>
                <div class="card-body">
                    {{ $products }}
                </div>
            </div>
        </div>
    </div>
</div>
</section>
</div>

@endsection

```

Setelah menambahkan dan memperbarui halaman pada bagian admin, Anda selanjutnya akan melakukan hal yang sama pada bagian user. Pertama-tama, Anda akan membuat sebuah controller baru dengan nama **UserController.php** dengan mengetikkan perintah dibawah ini dalam terminal Anda:

```
php artisan make:controller User/UserController
```

Setelah berhasil menambahkan **UserController.php**, tulislah kode dibawah ini dalam controller tersebut yang berlokasi dalam **app/Http/Controllers/User/UserController.php**:

```
1 namespace App\Http\Controllers\User;
2
3 use App\Http\Controllers\Controller;
4 use Illuminate\Http\Request;
5 use App\Models\Product;
6
7 class UserController extends Controller
8 {
9     public function index()
10    {
11        $products = Product::all();
12
13        return view('pages.user.index', compact('products'));
14    }
15 }
```

Langkah selanjutnya, Anda akan mengubah tampilan utama dari user ketika user berhasil melakukan proses login. Masuk ke dalam file **index.blade.php** yang berlokasi dalam **resources/views/pages/user/index.blade.php** kemudian ubahlah kode yang sebelumnya menjadi kode yang diberi tanda warna kuning dibawah ini:

```
@extends('layouts.user.main')
@section('content')
<!-- start banner Area -->
<section class="banner-area">
    <div class="container">
        <div class="row fullscreen align-items-center justify-content-start">
            <div class="col-lg-12">
                <div class="">
                    <!-- single-slide -->
                    <div class="row">
                        <div class="col-lg-5 col-md-6">
                            <div class="banner-content">
                                <h1>Nike New <br>Collection!</h1>
                                <p>Lorem ipsum dolor sit amet, consectetur
adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore
magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation.</p>
                            </div>
                        </div>
                        <div class="col-lg-7">
                            <div class="banner-img">
                                
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</section>
```

```

        </div>
    </div>
</section>
<!-- End banner Area -->

<!-- start product Area -->
<section class="section_gap">
    <!-- single product slide -->>
    <div class="container">
        <div class="row justify-content-center">
            <div class="col-lg-6 text-center">
                <div class="section-title">
                    <h1>Latest Products</h1>
                    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
                </div>
            </div>
        </div>
        <div class="row">
            <!-- single product -->
            @forelse ($products as $item)
                <div class="col-lg-3 col-md-6">
                    <div class="single-product">
                        
                        <div class="product-details">
                            <h6></h6>
                            <div class="price">
                                <h6>{{ $item->price }}</h6>
                            </div>
                            <div class="prd-bottom">
                                <a href="#" class="social-info">
                                    <span class="ti-bag"></span>
                                    <p class="hover-text">Beli</p>
                                </a>
                                <a href="#" class="social-info">
                                    <span class="lnr lnr-move"></span>
                                    <p class="hover-text">Detail</p>
                                </a>
                            </div>
                        </div>
                    </div>
                @empty
                    <div class="col-lg-12 col-md-12">
                        <div class="single-product">
                            <h3 class="text-center">Tidak ada produk</h3>
                        </div>
                    </div>
                @endforelse
            </div>
        </div>
    </div>
</section>
<!-- end product Area -->

@endsection

```

Langkah terakhir, Anda akan memperbarui kode route yang berada di dalam file **web.php**. Tulislah kode dibawah ini yang diberi tanda warna kuning:

```
use App\Http\Controllers\Auth\AuthController;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\Admin\AdminController;
use App\Http\Controllers\Admin\ProductController;
use App\Http\Controllers\User\UserController;

// Guest Route
Route::group(['middleware' => 'guest'], function() {
    Route::get('/', function () {
        return view('welcome');
    });

    Route::get('/register', [AuthController::class, 'register'])->name('register');
    Route::post('/post-register', [AuthController::class, 'post_register'])->name('post.register');

    Route::post('/post-login', [AuthController::class, 'login']);
})->middleware('guest');

// Admin Route
Route::group(['middleware' => 'admin'], function() {
    Route::get('/admin', [AdminController::class, 'dashboard'])->name('admin.dashboard');

    // Product Route
    Route::get('/product', [ProductController::class, 'index'])->name('admin.product');

    Route::get('/admin-logout', [AuthController::class, 'admin_logout'])->name('admin.logout');
})->middleware('admin');

// User Route
Route::group(['middleware' => 'web'], function() {
    Route::get('/user', [UserController::class, 'index'])->name('user.dashboard');

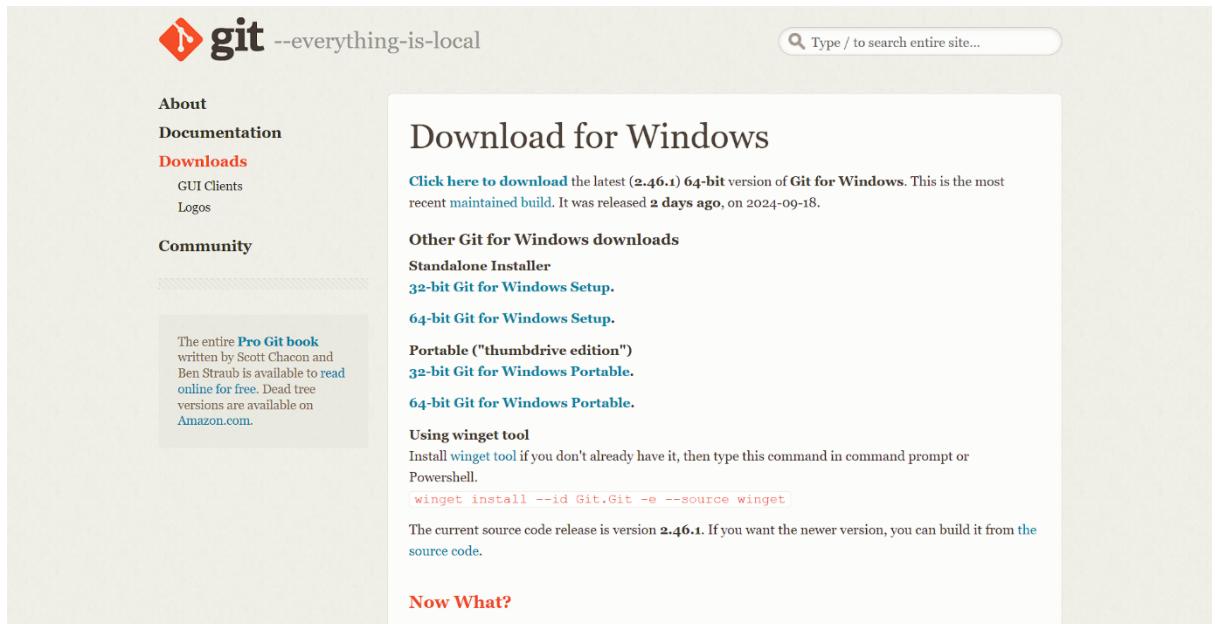
    Route::get('/user-logout', [AuthController::class, 'user_logout'])->name('user.logout');
})->middleware('web');
```

## Tugas

Selanjutnya pindah kan data halaman Distributor yang sudah dibuat sebelumnya pada halaman **Admin**.

Terakhir, Anda perlu membuat sebuah folder baru yang akan menyimpan gambar produk yang akan ditampilkan ke dalam halaman website. Untuk membuat folder baru tersebut, buatlah di dalam folder **public** dengan nama folder adalah **images**. Untuk bagian ini, Anda belum bisa menambahkan data produk baru dan hanya bisa menampilkan data produk saja. Dikarenakan belum ada data produk, maka yang akan tampil pada website adalah sebuah tulisan yang mengatakan data kosong atau tidak ada. Pada bagian selanjutnya, Anda akan mempelajari bagaimana caranya untuk melakukan proses CRUD dalam laravel dengan menggunakan fungsi yang sudah disediakan oleh laravel.

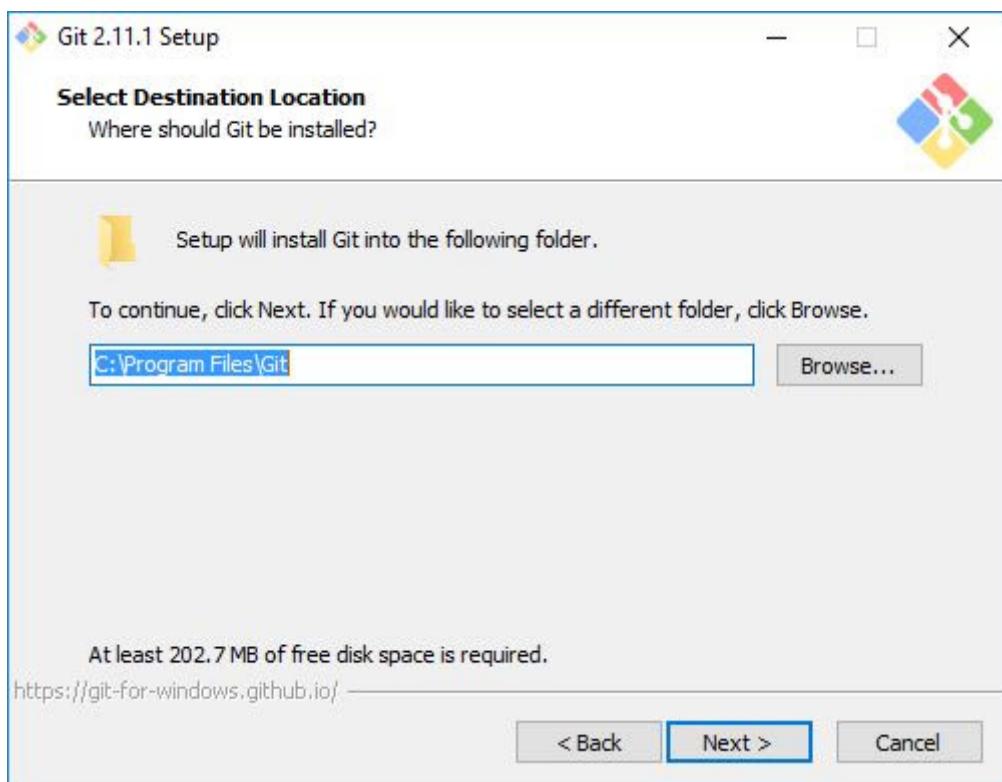
Sebelum mengakhiri modul ini, Anda akan membuat sebuah repositori baru dalam github lalu mengupload project yang sudah Anda buat ke dalam repositori tersebut. Untuk itu, pertama-tama Anda perlu melakukan instalasi git ke dalam perangkat Anda. Instalasi git yang akan dilakukan pada bagian ini adalah menggunakan windows, untuk itu jika Anda menggunakan MacOs atau Linux untuk dapat menyesuaikan. Untuk memulai instalasi git, Anda perlu mendownload installer nya pada link berikut ini <https://git-scm.com/download/win> kemudian anda pilih arsitektur windows yang sesuai dengan yang ada dalam perangkat Anda. Pada gambar dibawah ini, arsitektur yang dipilih adalah 64-bit:



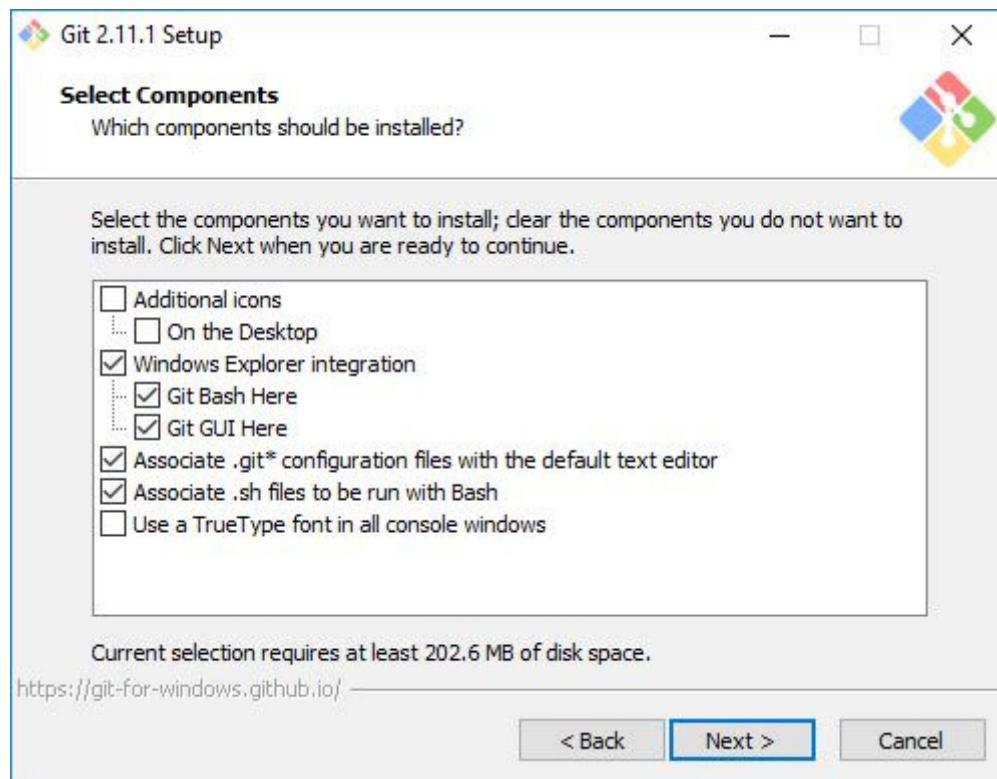
Setelah file installer git berhasil di download, Anda dapat menjalankan file installer tersebut untuk melanjutkan ke dalam tahap berikutnya. Ketika file installer tersebut dijalankan, maka akan muncul sebuah windows yang berisi informasi tentang lisensi git. Klik “Next” untuk melanjutkan ke bagian selanjutnya.



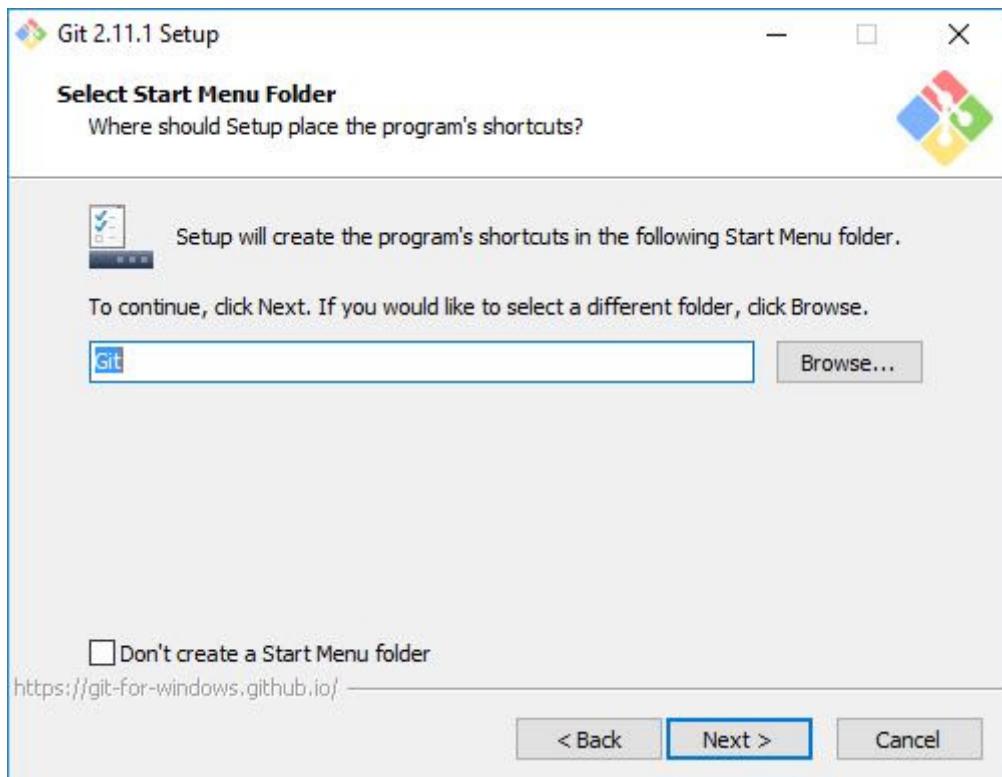
Setelah Anda mengklik tombol next maka akan muncul windows seperti pada gambar dibawah ini. Untuk lokasi tempat git akan diinstall, biarkan saja apa adanya lalu klik tombol next.



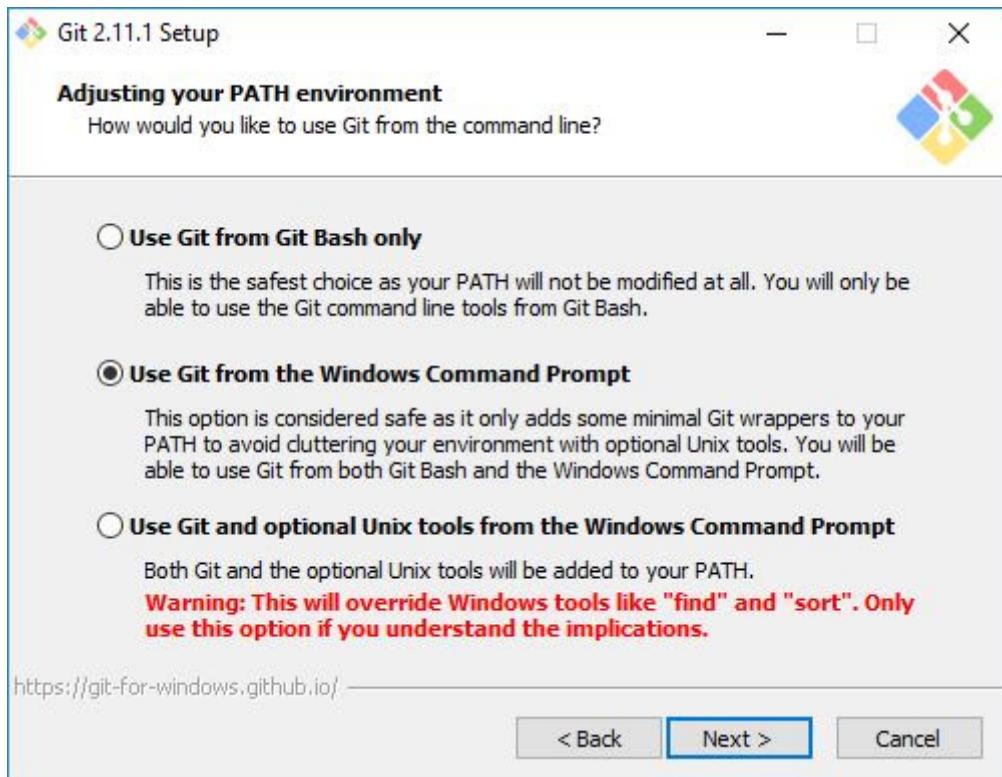
Setelah Anda mengklik tombol next, maka akan muncul tampilan terkait pemilihan komponen. Pada bagian ini, biarkan saja apa adanya dan klik tombol next.



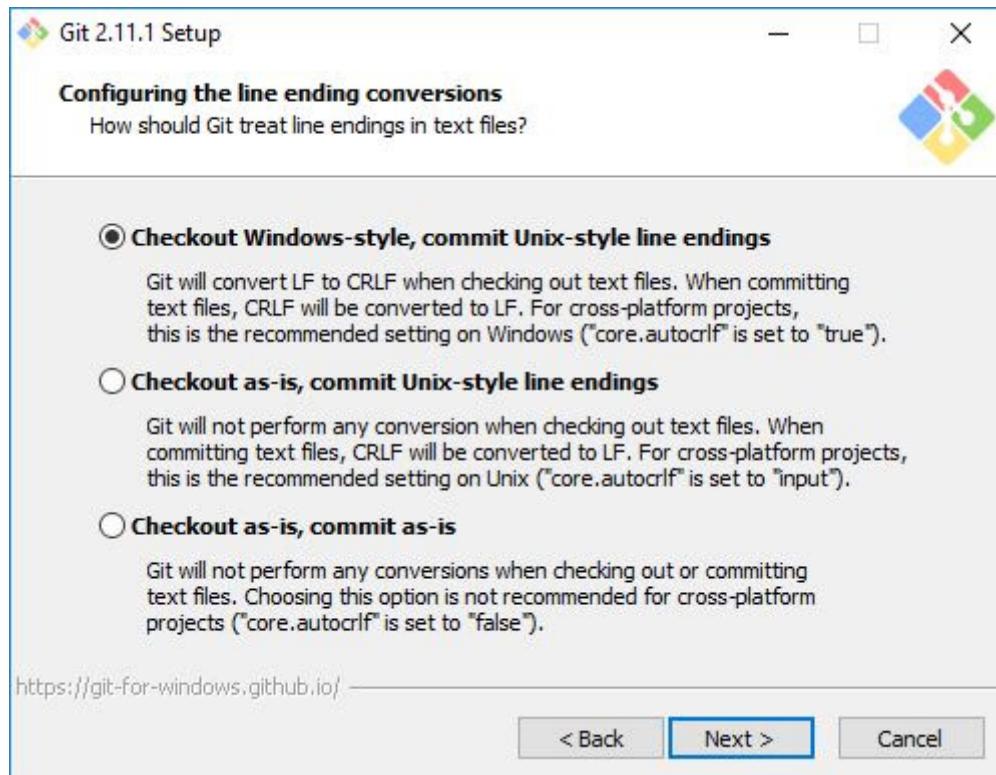
Ketika Anda mengklik tombol next pada bagian sebelumnya, akan muncul sebuah windows yang berisi pemilihan start direktori. Anda cukup klik tombol next saja karena Anda tidak perlu mengubah start direktori tersebut.



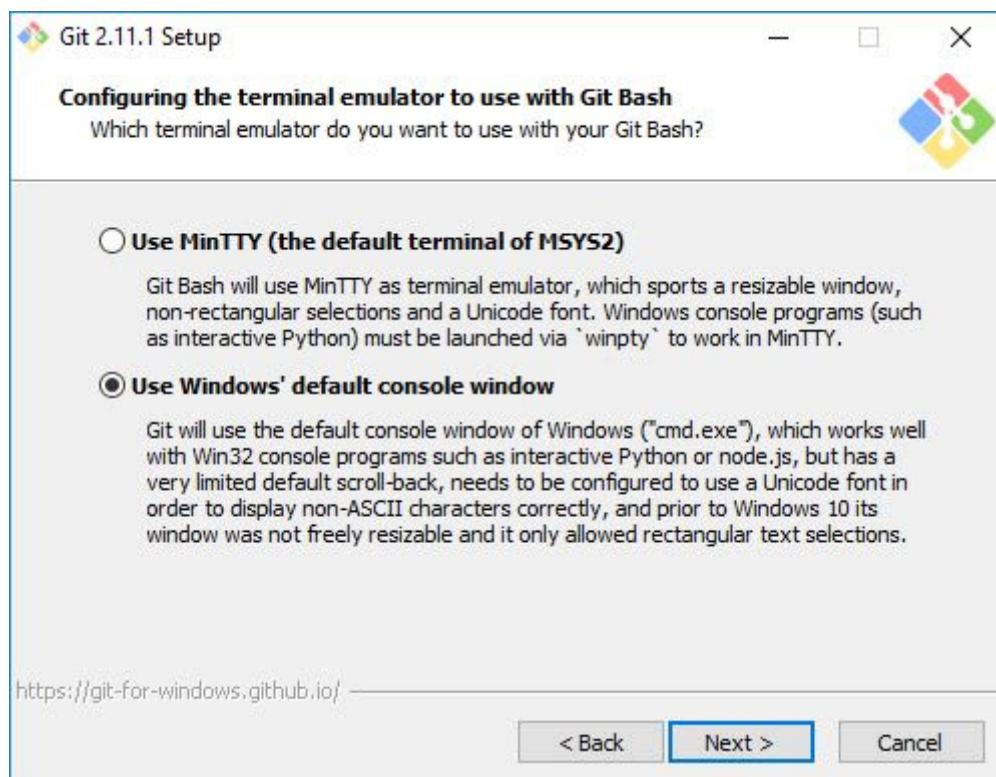
Setelah Anda mengklik next, akan muncul sebuah windows yang berisi pengaturan path environment. Pada bagian ini, Anda pilih opsi kedua yang ditunjukkan pada gambar dibawah ini agar cmd pada windows Anda mengenali perintah git, kemudian Anda klik next.



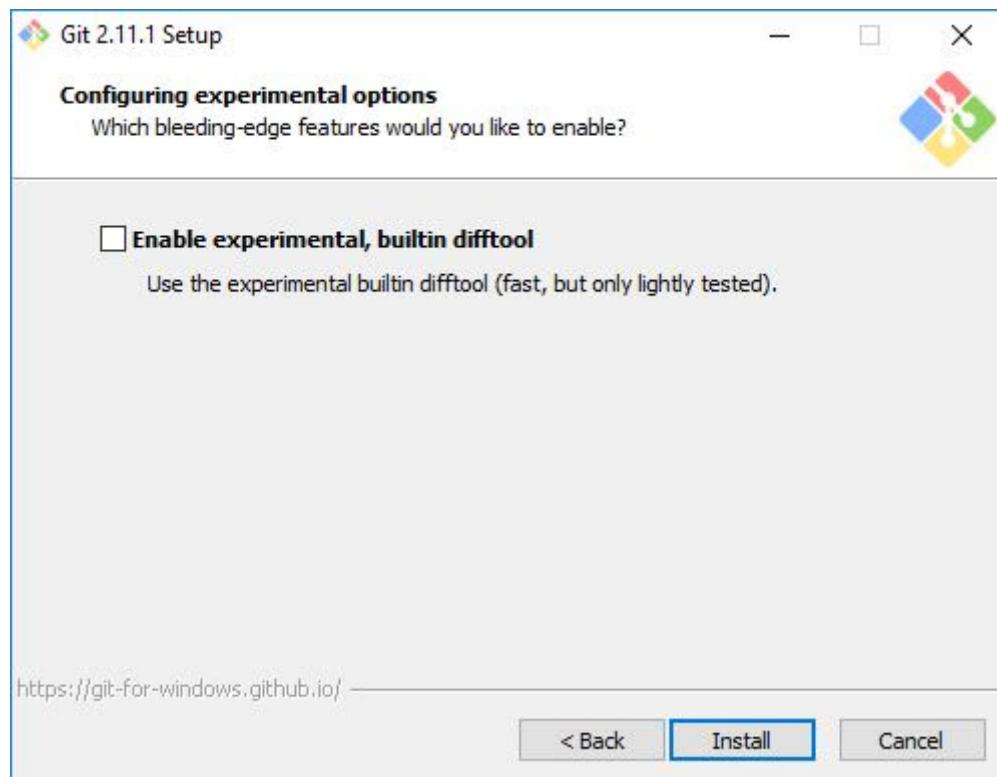
Setelah Anda mengklik tombol next pada bagian sebelumnya, akan muncul windows seperti gambar dibawah ini. Biarkan saja kemudian Anda klik tombol next lagi.



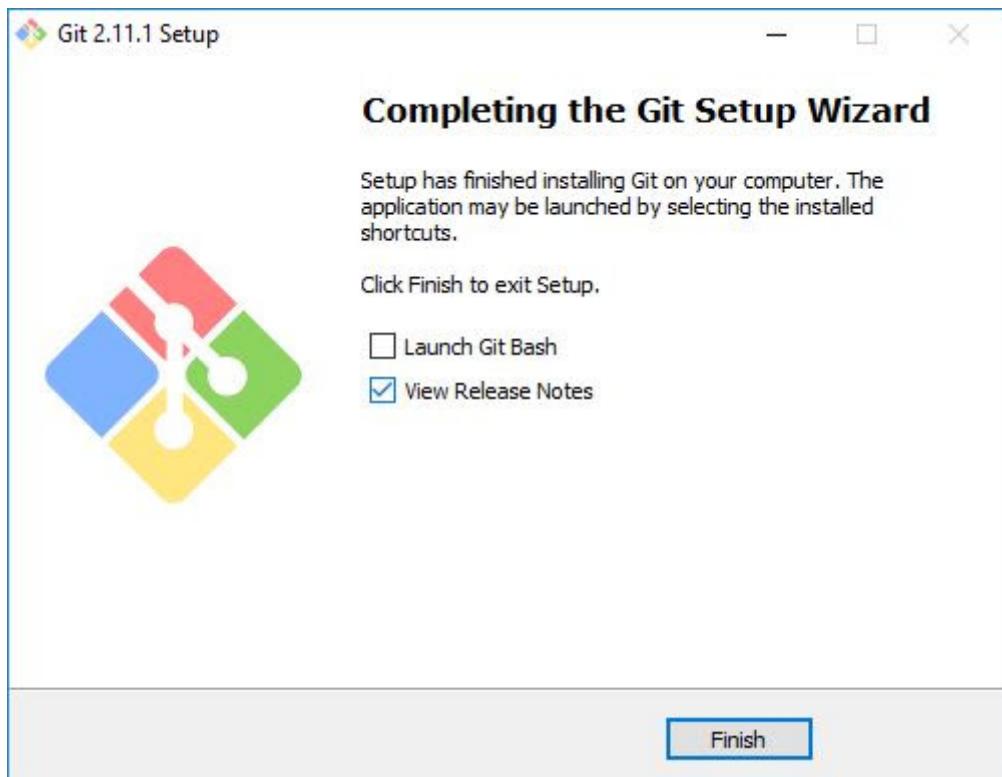
Setelah itu, akan muncul sebuah windows seperti dibawah ini yang berisi pemilihan emulator terminal. Pada bagian ini, Anda pilih opsi kedua kemudian klik tombol next.



Setelah itu akan muncul lagi sebuah windows yang berisi pemilihan experimental, Anda cukup klik install saja untuk melakukan proses instalasi git.



Jika proses instalasi berhasil, maka akan muncul sebuah windows seperti gambar dibawah ini yang berisi sebuah informasi bahwa git berhasil di install ke dalam perangkat Anda.



Untuk memastikan apakah git benar-benar sudah di install ke dalam perangkat Anda, Anda dapat membuka terminal lalu ketikkan perintah “**git –version**”. Jika git berhasil di install, maka akan muncul pesan berikut ini dalam terminal Anda.

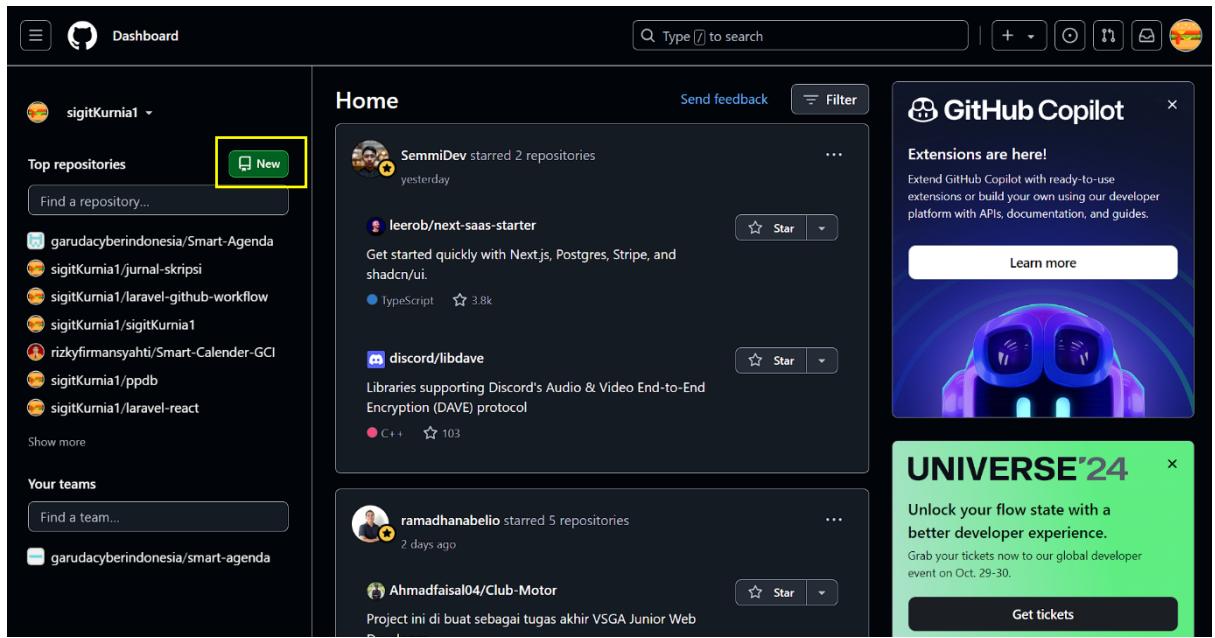
A screenshot of a Windows Command Prompt window titled "Command Prompt". The window shows the following text:

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

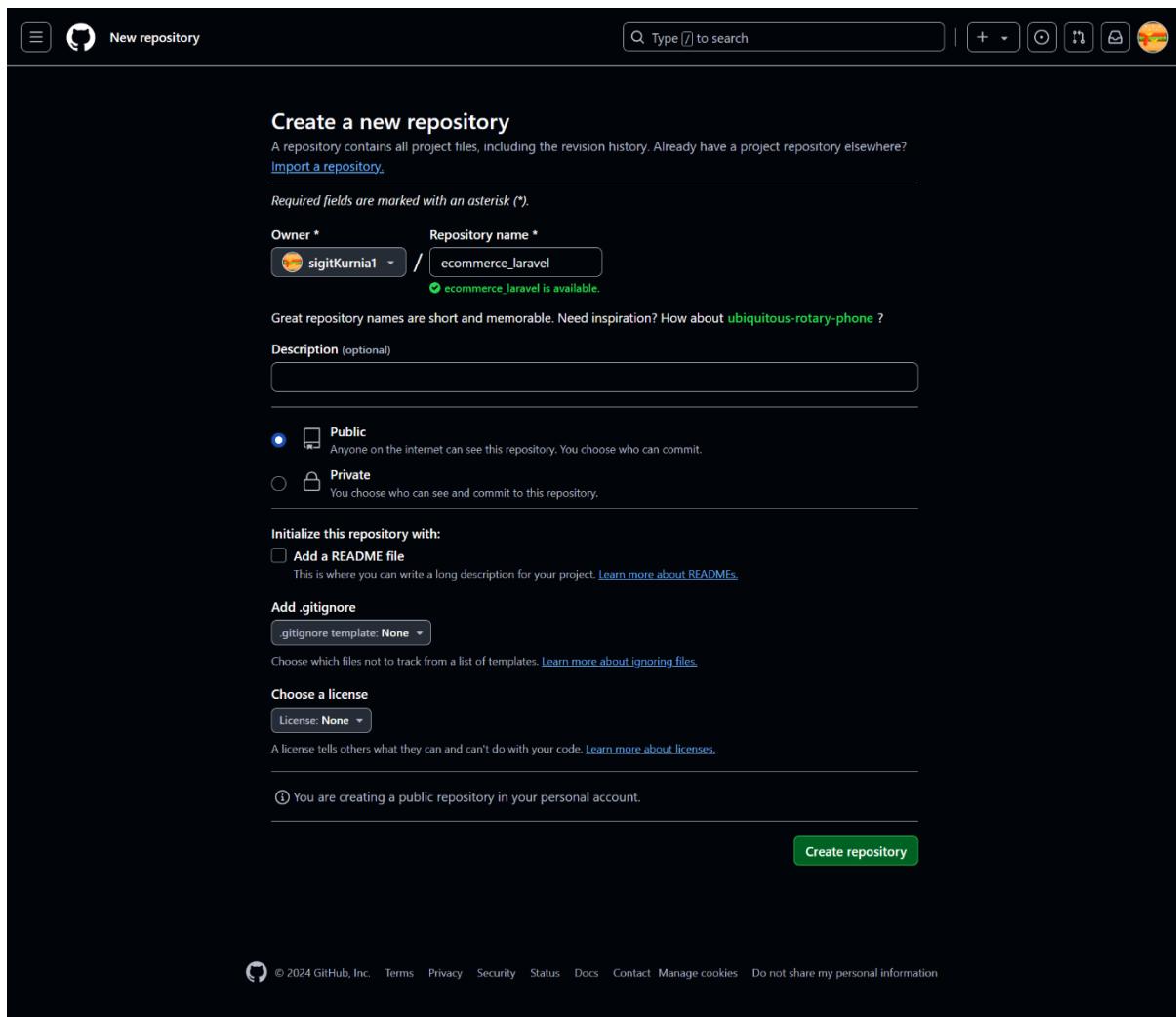
C:\Users\Development>git --version
git version 2.11.1.windows.1

C:\Users\Development>
```

Setelah git berhasil di install ke dalam perangkat Anda, selanjutnya Anda akan membuat sebuah repositori baru galam github namun sebelumnya Anda harus membuat akun github jika Anda belum mendaftar. Anda dapat melakukan proses pendaftaran akun baru github dengan mengunjungi link berikut ini [https://github.com/signup?ref\\_cta=Sign+up&ref\\_loc=header+logged+out&ref\\_page=%2F&source=header-home](https://github.com/signup?ref_cta=Sign+up&ref_loc=header+logged+out&ref_page=%2F&source=header-home). Ikuti arahan yang diberikan dengan teliti dan jangan sampai ada yang salah. Jika sudah berhasil, masuk ke dalam akun github lalu klik tombol new pada gambar dibawah ini untuk dapat membuat repositori github baru:



Jika Anda mengklik tombol new, maka Anda akan diarahkan ke dalam halaman yang berisi informasi dari repositori yang akan Anda buat tersebut. Isilah informasi tersebut dengan benar dan sesuai dengan yang ada pada gambar dibawah ini. Jika sudah mengisi informasi repositori baru yang akan ditambahkan tersebut, klik tombol “**Create Repository**”:



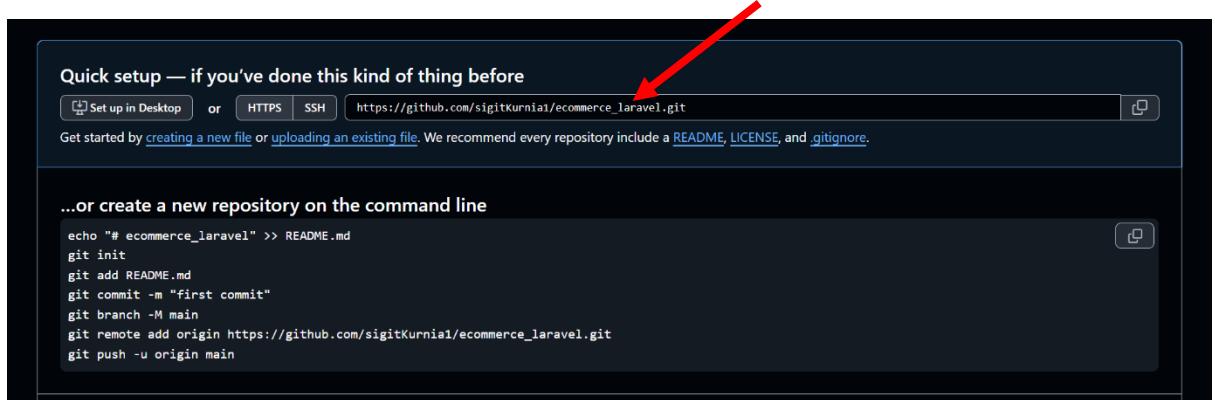
Jika sudah maka akan muncul tampilan seperti gambar berikut ini. Karena repositori ini masih kosong, maka yang muncul hanya sebuah panduan yang dapat Anda gunakan untuk menambahkan sebuah file ke dalam repositori baru tersebut. Selanjutnya Anda akan mengupload project laravel Anda ke dalam repositori tersebut, untuk itu buka terminal dalam direktori project laravel Anda kemudian ikuti langkah-langkah nya dengan teliti. Di dalam terminal tersebut, ketikkan perintah seperti dibawah ini:

```
PS D:\Project\Dir\Modul Laravel 11\ecommerce-laravel> git init
Initialized empty Git repository in D:/Project/Dir/Modul Laravel 11/ecommerce-laravel/.git/
PS D:\Project\Dir\Modul Laravel 11\ecommerce-laravel> []
```

Selanjutnya, Anda akan menghubungkan project Anda dengan repositori github dengan mengetikkan perintah berikut ini dalam terminal Anda:

```
Remote add origin [link repository anda] ecommerce_laravel.git
```

Perlu di ingat, bahwa Anda perlu menyesuaikan link repositori github yang Anda miliki. Anda bisa melihat link tersebut pada gambar di bawah ini:



Setelah itu, Anda ketikkan perintah dibawah ini dalam terminal Anda untuk menambahkan file yang akan di upload ke dalam repositori yang Anda buat:

```
PS D:\Project\Dir\Modul Laravel 11\ecommerce-laravel> git add .
```

Selanjutnya, Anda akan melakukan commit sebelum mengupload (push) ke dalam repositori github yang baru saja Anda buat. Usahakan pesan commit yang dibuat se-informatif mungkin untuk mengikuti best-practice nya. Untuk melakukan commit, ketikkan perintah dibawah ini dalam terminal Anda:

```
PS D:\Project\Dir\Modul Laravel 11\ecommerce-laravel> git commit -m "First Commit"
```

Setelah itu, Anda akan melakukan perpindahan branch yang awalnya berada pada branch **Master** menjadi branch **Main**. Untuk melakukan hal tersebut, ketikkan perintah dibawah ini dalam terminal Anda:

```
PS D:\Project\Dir\Modul Laravel 11\ecommerce-laravel> git branch -M main
PS D:\Project\Dir\Modul Laravel 11\ecommerce-laravel>
```

Terakhir, Anda akan mengupload project laravel Anda dengan mengetikkan perintah dibawah ini dalam terminal Anda. Sebelum melakukan push ke dalam repositori github, pastikan Anda memiliki koneksi internet yang baik karena jika tidak maka Anda tidak dapat melakukan proses push (upload) project laravel Anda ke dalam repositori github:

```
PS D:\Project\Dir\Modul Laravel 11\ecommerce-laravel> git push -u origin main
Enumerating objects: 5311, done.
Counting objects: 100% (5311/5311), done.
Delta compression using up to 12 threads
Compressing objects: 100% (5279/5279), done.
Writing objects: 100% (5311/5311), 31.79 MiB | 512.00 KiB/s, done.
Total 5311 (delta 955), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (955/955), done.
To https://github.com/sigitKurnia1/ecommerce_laravel.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS D:\Project\Dir\Modul Laravel 11\ecommerce-laravel> []
```

Sampai disini, Anda berhasil melakukan proses upload project laravel Anda ke dalam repositori github dan jika Anda kembali ke halaman repositori github yang sudah Anda buat, maka akan muncul tampilan seperti gambar dibawah ini yang menandakan bahwa project laravel berhasil diupload ke dalam github: