**PRAKTIKUM**

**PEMROGRAMAN FRAMEWORK**

**MODUL 10**

**Laravel Third Party Packages**

****

**Dosen Pengampu :**

Purnama Anaking, S.Kom., M.Kom.

**Oleh :**

Aisyah Nabila Zahra 1204210122

**PROGRAM STUDI S1 SISTEM INFORMASI**

**FAKULTAS TEKNOLOGI TEKNOLOGI DAN BISNIS**

**INSTITUT TEKNOLOGI TELKOM SURABAYA**

**2023**

**Laravel Third Party Packages**

1. **Penerapan Laravel DataTables**

Untuk menerapkan *data table*, pertama dengan dapat menambahkan data *dummy* banyak 200 melalui file **EmployeeSeeder**.

**A screen shot of a computer program

Description automatically generated**

Jika kode program pada file **EmployeeSeeder** sudah ditambahkan dapat, menjalankan file *seeder* melalui perintah artisan sebagai berikut:

**A screen shot of a computer

Description automatically generated**

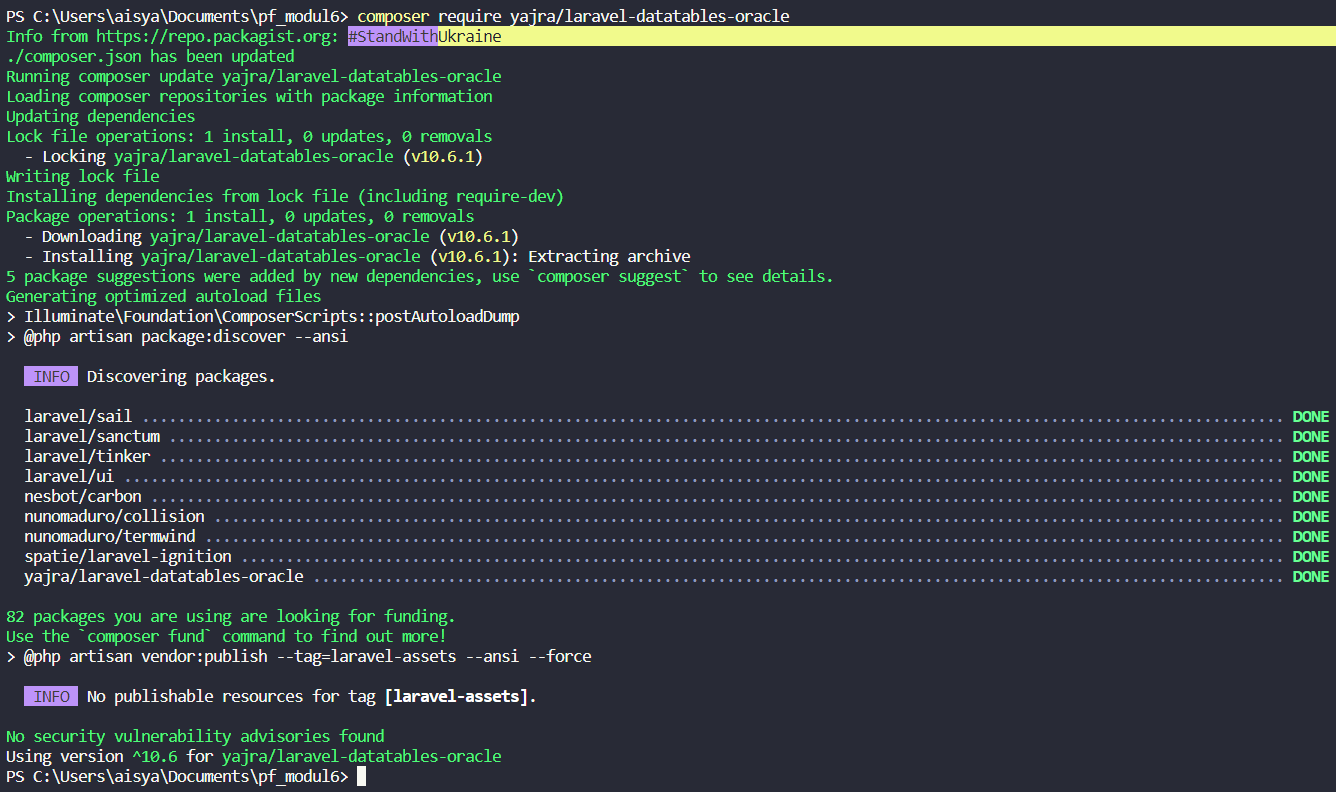
Perintah tersebut akan menambahkan data *entri* baru ke tabel *employees* sebaganya 200 data *entries.*

A screenshot of a computer

Description automatically generated

**1.1 Basic Data Tables**

Penerapan *basic data table*, dimulai dari meng-*install* *packages* **Laravel Data Tables**melalui **Composer**.

****

Kemudian install asset-aset Data Tables yang dibutuhkan melalui npm:

**A screen shot of a computer

Description automatically generated**

Masuk ke file **/resources/sass/app.scss** kemudian tambahkan asset-aset DataTables, sehingga hasilnya seperti berikut:

A screenshot of a computer code

Description automatically generated

Kemudian masuk ke file **/resources/js/bootstrap.js** dan import JQuery:

A screen shot of a computer

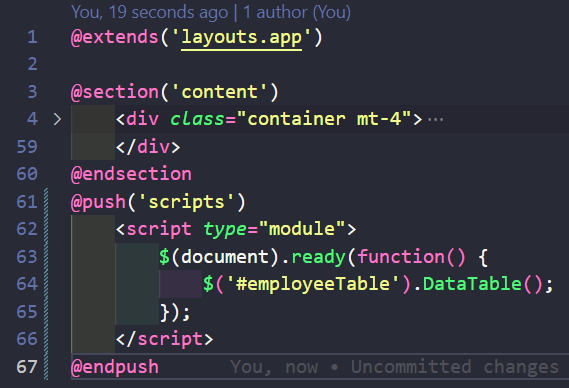
Description automatically generated

Berikutnya buka file /resources/views/layouts/app.blade.php. Kemudian menambahkan blade directive **stack()** dengan nama “script”, seperti gambar berikut:

A screen shot of a computer program

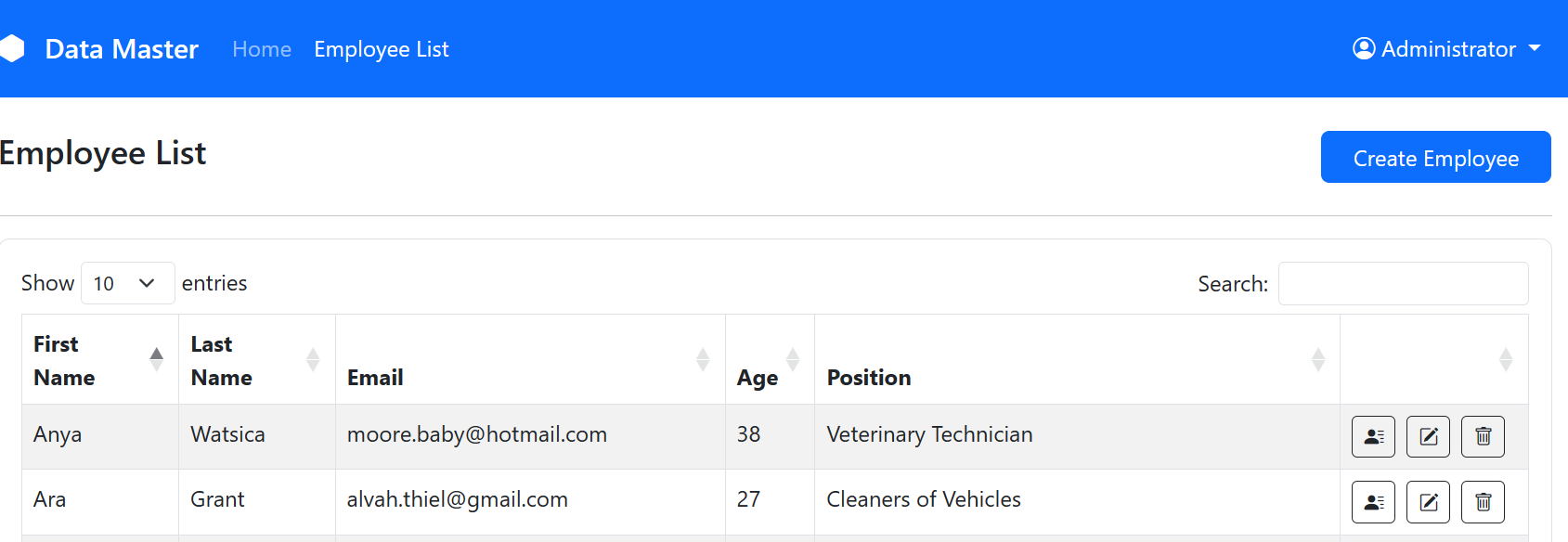
Description automatically generated

Selanjutnya buka file view **index.blade.php**, kemudian tambahkan block blade directive **push()**, sebagai berikut:



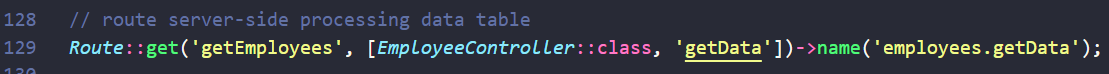
Terakhir akses halamana employee list melalui browser, untuk memastikan jika tampilan DataTables berhasil.

**Hasil Tampilan:**

****

**1.2 Server-side Processing DataTables**

Untuk menerapkan *server-side processing* DataTables dapat dimulai dari membuat route terlebih dahulu, berikut kode program untuk route:

****

Pada file **EmployeeController** ditambahkan methode **getData()**, dengan kode program sebagai berikut:

A screen shot of a computer program

Description automatically generated

Kemudian, pada file yang sama di fungsi index terdapat perubahan kode program. Dimana kode query eloquent untuk mengambil data employee sudah tidak diperlukan lagi. Karena sudah digantikan oleh method getData(). Sehingga hasil hasil kode program yang ada sebagai berikut:

A computer screen shot of a list

Description automatically generated

Kemudian masuk ke dalam file view /resources/views/employee/index.blade.php. Pada kode program tersebut bagian tag <table> ditambahkan kode menjadi sebagai berikut:

A screen shot of a computer

Description automatically generated

Kemudian pada bagian perulangan dengan tag <tbody> bisa dihapus karena data akan di load secara asynchronous dan ditangani oleh DataTable.

Berikutnya masuk ke file view /resources/views/employee/index.blade.php. Kemudian tambahkan *script* DataTable sebagai berikut yang diletakkan dibawah section:

A screen shot of a computer program

Description automatically generated

Jika sudah selesai, akses halaman employee list melalui browser dan hasilnya akan memunculkan tampilan DataTables untuk data employee dengan teknik *server-side processing*. Dengan hasil tampilan sebagai berikut:

**Hasil tampilan:**

**A screenshot of a computer

Description automatically generated**

1. **Penerapa Laravel Sweet Alert**

Untuk menerapkan Laravel Sweet Alert, dapat dimulai dengan menginstal package SweetAlert melalui composer.

**A screenshot of a computer program

Description automatically generated**

Kemudian buka file app.blade.php dan tambahkan blade directive **include()** untuk menerapkan *package SweetAlert* yang diletakkan diantara **vite()** dan **stack()**.

A computer screen shot of a program

Description automatically generated

Untuk membuat file konfigurasi dan asek publik yang diperlukan oleh paket Laravel SweetAlert ke dalam direktori proyek. Jalankan perintah Publish asset dari SweetAlert via Artisan:

**A screen shot of a computer

Description automatically generated**

Berikutnya buka file **EmployeeController** dan tambahkan method SweetAlertpada method **store()**, **update()**,dan **destroy()**. Dengan kode program sebagai berikut:

A screenshot of a computer program

Description automatically generated

Kode fungsi store

A screen shot of a computer program

Description automatically generated

kode program untuk fungsi update

A computer screen shot of code

Description automatically generated

kode program untuk fungsi destroy

Setelah menambahkan bagian fungsi, buka file view /resources/views/employee/actions.blade.php. Kemudian sesuaikan kode program dengan tag <button>. Tambahkan *class* “btn-delete” dan atribut “data-name”. Dengan hasil penulisan kode program sebagai berikut:

A computer screen shot of text

Description automatically generated

Berikutnya pada file view **index.blade.php** kode dengan tag <table>. Tambahkan class “datatable”. Dengan hasil penulisan kode program sebagai berikut:



Berikutnya pada file view yang sama tambahkan kode program untuk bagian block blade directive **push()**, dengan kode program sebagai berikut:

A computer screen shot of a program code

Description automatically generated  
Kemudian pada bagian file EmployeeController, tambahkan kode program yang memanggil *method* confirmDelete() dari SweetAlert.

A computer screen shot of a program

Description automatically generated

Berikutnya akses fitur **create, edit,** dan **delete** melalui browser. Jika kode berjalana dengan baik atau berhasil. Contohnya pada fitur delete akan muncul tampilan pop up windows dari SweetAlert, yang menjadi tanda *feedback* sistem untuk fitur yang sedang diakses oleh user. Contoh hasil implmentasi sebagai berikut:

**Hasil tampilan:**

**A screenshot of a computer

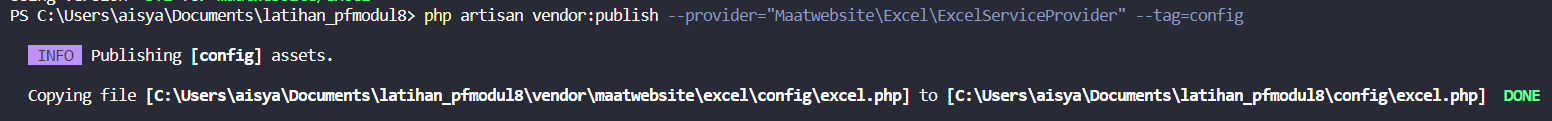
Description automatically generated**

1. **Penerapan Laravel Excel**

Hal pertama yang dilakukan untuk mengakses Laravel excel melalui composer, dapat dimulai dengan menginstal Laravel Excel. Berikut hasil implementasi perintah yang dijalankan:

**A computer screen shot of a program

Description automatically generated**

Kemudian, jalankan perintah publish untuk file konfigurasi yang akan disimpan ke dalam direktori proyek. Beriku kode perintah yang dijalankan.****

Buka file index.blade.php dan tambahkan button untuk fitur export data employee menjadi excel. Berikut hasil penulisan kode program:

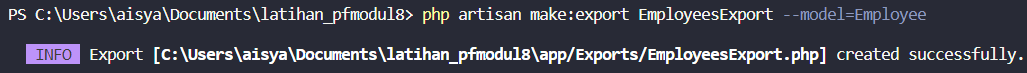
A computer screen shot of a program code

Description automatically generated

Buat route baru **/exportExcel** yang mengarah ke method exportExcel() pada EmployeeController. Berikut hasil penulisan route baru:



Berikutnya buat file EmployeeExport:

****

Kemudian masuk ke file yang baru saja degenerate dan tambahkan kode program untuk fungsi style dan view. Dengan kode program sebagai berikut:

A screenshot of a computer program

Description automatically generated

Buat file view baru untuk export excel dengan nama **export\_excel.blade.php** pada direktori /resources/views/employee. Dengan isi kode program sebagai berikut:

A screenshot of a computer program

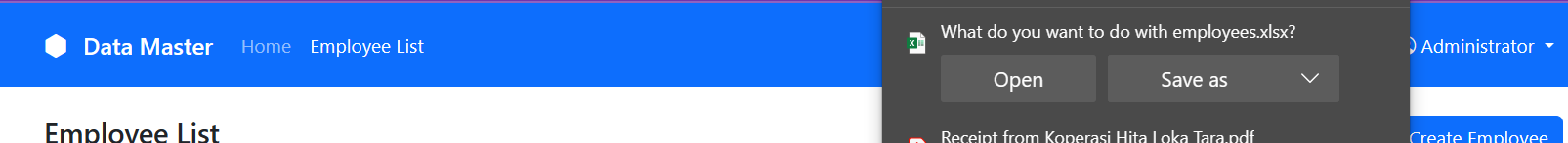
Description automatically generated

Berikutnya masuk kembali ke file **EmployeeController**, kemudian tambahkan method exportExcel(). Dengan kode program sebagai berikut:

A screenshot of a computer screen

Description automatically generated

Selanjutnya, akses halaman employee list, dan klik button “to excel” untuk export data employee menjadi file excel. Berikut hasil implementasi:



1. **Penerapan Laravel DomPdf**

Untuk menerapkan fitur export pdf, terlebih dahulu untuk intal Laravel DomPdf melalui composer, berikut hasil implementasi:

**A screen shot of a computer program

Description automatically generated**

Kemudian jalankan perintah publish, untuk mempublis file konfigurasi dengan menjalankan perintah “php artisan vendor:publish –provider= “Barryvdh\DomPDF\ServiceProvider” dengan hasil sebagai berikut:

****

Selanjutnya buka file view /resources/views/employee/index.blade.php. Kemudian tambahkan button untuk fitur export data menjadi file pdf. Berikut kode program file view yang digunakan:

A screen shot of a computer program

Description automatically generated

Kemudian buat route baru untuk fitur export pdf, yang mengarah ke method exportPdf() pada file EmployeeController(). Dengan kode program sebagai berikut:



Selanjutnya, buat file view baru dengan nama **export\_pdf.blade.php** pada direktori /resources/views/employee, yang berisi kode program sebagai berikut:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

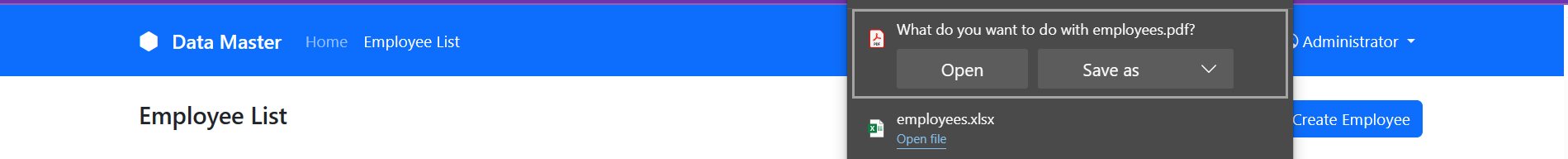
Description automatically generated

Masuk kembali ke file EmployeeController, dan buat methode **exportPdf()** yang didalamnya ditambahkan kode program sebagai berikut:

A computer screen shot of a program code

Description automatically generated

Berikut hasil akses fitur “to PDF”:



1. **Memindahkan dan Memisahkan Kode Javascript**

Untuk mengimplementasikan pemindahakan kode javascript yang ada pada file view dan mengelompokkannya dengan file javascript lainnya dapat dimulai dari membuat file js baru. Contohnya yaitu membuat file **employee.js** pada direktori /resources/js/. Kemudian hapus block kode yang ada pada blade directive **push()** pada file direktori /resources/views/index.blade.php. Kemudian pindahkan kode javascript teresbut ke file employee.js. Berikut hasil penulisan kode program pada **employee.js**:

A screen shot of a computer program

Description automatically generated

Berikutnya tambahkan registrasi untuk file employee.js pada file app.js yang ada pada direktori /resources/js/app.js. Dengan implementasi sebagai berikut:

A screen shot of a computer code

Description automatically generated

**Link Repositori Github:** [aisyahnabila/pf\_modul10: Praktikum 10, menerapkan semua package yang ada pada praktikum 9 (github.com)](https://github.com/aisyahnabila/pf_modul10)