LAPORAN TUGAS AKHIR PEMBUATAN

APLIKASI DATA SISWA



Di susun oleh:

Miftahul Hasanah

BAB I - PENDAHULUAN

1.1 Latar Belakang

Laravel adalah salah satu framework PHP yang banyak digunakan dalam pengembangan website karena memiliki struktur yang rapi dan mempermudah proses pembuatan aplikasi. Dalam sebuah aplikasi sekolah, data antar tabel perlu saling berhubungan agar informasi dapat dikelola dengan baik.

Pada proyek ini dibuat hubungan antara tabel siswa dan tabel kelas menggunakan framework Laravel. Jenis relasi yang digunakan yaitu One ToMany, dimana satu kelas dapat ditempati oleh banyak siswa, sedangkan satu siswa hanya terdaftar pada satu kelas saja. Dengan adanya relasi tersebut, data menjadi lebih terstruktur dan mudah ditampilkan pada aplikasi web.

1.2 Tujuan

1. Mengetahui konsep relasi tabel pada framework Laravel.
2. Membuat hubungan data antara tabel siswa dan tabel kelas.
3. Menampilkan hasil relasi data ke dalam halaman web Laravel.
4. Memahami penggunaan migration, model, controller, dan view dalam Laravel.

BAB I I-PEMBAHASAN

2.2Migration Tabel Kelas



```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
    Schema::create('kelas', function (Blueprint $table) {
        $table->id();
        $table->string('kelas');
        $table->timestamps();
    });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('kelas');
    }
};

```

1.1 gambar Migration Tabel Kelas

Kode migration tersebut digunakan untuk membuat tabel `kelas` pada database Laravel. Pada bagian `Schema::create('kelas')`, Laravel akan membuat tabel baru dengan nama `kelas`. Di dalam tabel tersebut terdapat kolom `id` yang berfungsi sebagai primary key dan dibuat otomatis menggunakan `$table->id()`.

Selanjutnya, `$table->string('kelas')` digunakan untuk membuat kolom `kelas` dengan tipe data string yang berfungsi menyimpan nama kelas. Kemudian `$table->timestamps()` digunakan untuk menambahkan kolom `created_at` dan `updated_at` secara otomatis, sehingga sistem dapat mencatat waktu data dibuat dan diperbarui.

Pada method `down()`, terdapat perintah `Schema::dropIfExists('kelas')` yang berfungsi untuk menghapus tabel `kelas` apabila migration di-rollback. Dengan demikian, migration ini berfungsi untuk mengatur struktur tabel `kelas` pada database Laravel.

2.2 Migration Tabel Siswa



```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
    Schema::create('siswas', function (Blueprint $table) {
    $table->id();
    $table->string('nama');
    $table->foreignId('kelas_id')->nullable()->constrained('kelas')->onDelete('cascade');
    $table->string('jurusan');
    $table->timestamps();
    });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('siswas');
    }
};

```

1.2 gambar Migration Tabel Siswa

Kode migration tersebut digunakan untuk membuat tabel siswas pada database Laravel. Tabel ini memiliki kolom id, nama, jurusan, dan kelas_id. Kolom kelas_id digunakan sebagai foreign key untuk menghubungkan tabel siswa dengan tabel kelas sehingga terbentuk relasi One To Many. Selain itu, terdapat timestamps() untuk mencatat waktu pembuatan dan perubahan data secara otomatis. Method down() berfungsi untuk menghapus tabel siswas jika migration di-rollback.

2.3 Relasi Model

relasi model siswa:



```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use App\Models\Kelas;

class Siswa extends Model
{
    protected $table = 'siswas';

    protected $fillable = [
        'nama',
        'kelas_id',
        'jurusan',
    ];

    public function kelas()
    {
        return $this->belongsTo(Kelas::class, 'kelas_id');
    }
}

```

1.3 gambar Relasi Model

Kode tersebut merupakan model Siswa pada Laravel yang digunakan untuk menghubungkan tabel siswas dengan aplikasi. Bagian $fillable berfungsi menentukan kolom yang dapat diisi, yaitu nama, kelas_id, dan jurusan.

Pada method kelas(), digunakan belongsTo(Kelas::class, 'kelas_id') untuk membuat relasi bahwa satu siswa hanya memiliki satu kelas. Model ini digunakan agar data siswa dapat terhubung dengan tabel kelas pada Laravel.

Relasi model kelas:



```php
<?php

namespace App\Models;

use App\Models\Siswa;
use Illuminate\Database\Eloquent\Model;

class Kelas extends Model
{
    protected $fillable = [
        'kelas',
        ];

    public function siswa()
{
    return $this->hasMany(Siswa::class, 'kelas_id');
}
}
```

1.3 gambar Relasi Model Kelas

Kode tersebut merupakan model `Kelas` pada Laravel yang digunakan untuk menghubungkan tabel `kelas` dengan tabel `siswas`. Bagian `$fillable` berfungsi menentukan kolom yang dapat diisi, yaitu `kelas`.

Pada method `siswa()`, digunakan `hasMany(Siswa::class, 'kelas_id')` untuk membuat relasi bahwa satu kelas dapat memiliki banyak siswa. Model ini digunakan agar data kelas dapat terhubung dengan data siswa pada Laravel.

2.5 Form Input Siswa

```blade
<form action="/siswa" method="POST">
    @csrf

    <input type="text" name="nama" placeholder="Nama siswa">

    <select name="kelas_id">
        <option value="">Pilih Kelas</option>
        @foreach ($kelas as $item)
            <option value="{{ $item->id }}">{{ $item->kelas }}</option>
        @endforeach
    </select>

    <input type="text" name="jurusan" placeholder="Jurusan">

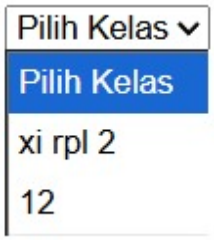
    <button type="submit">Simpan</button>
</form>
```

1.4 gambar From Input Siswa

Kode tersebut merupakan form input data siswa pada Laravel Blade. Form ini digunakan untuk menambahkan data siswa ke database melalui method `POST` ke route `/siswa`.

Bagian `@csrf` berfungsi sebagai keamanan form agar terhindar dari serangan CSRF. Input `nama` dan `jurusan` digunakan untuk mengisi data siswa, sedangkan `<select name="kelas_id">` digunakan untuk memilih kelas siswa.

Perulangan `@foreach($kelas as $item)` berfungsi menampilkan daftar kelas dari database ke dalam pilihan select. Tombol `Simpan` digunakan untuk mengirim data ke database.



2.6 Tampilan Data Siswa



Penjelasan: Pada halaman ini data siswa di gunakan untuk menampilkan nama siswa, jurusan, dan kelas siswa yang sudah terhubung dengan table kelas.

```blade
<h2>Data Siswa</h2>

<a href="/siswa/create">Tambah Data</a>

<table border="1">
<tr>
    <th>Nama</th>
    <th>Kelas</th>
    <th>Jurusan</th>
    <th>Aksi</th>
</tr>

@foreach($siswa as $s)
<tr>
    <td>{{ $s->nama }}</td>
    <td>{{ $s->kelas->kelas ?? '-' }}</td>
    <td>{{ $s->jurusan }}</td>
    <td>
```

Bagian @foreach($siswa as $s) digunakan untuk melakukan perulangan data siswa dari database. Kemudian {{ $s->kelas->kelas ?? '-' }} digunakan untuk menampilkan nama kelas berdasarkan relasi antara tabel siswa dan kelas. Link Tambah Data digunakan untuk menuju halaman tambah data siswa.

2.7 Tabel tampil data siswa

Hasil tampilan data siswa menunjukkan bahwa data yang telah berhasil disimpan ke dalam tabel siswas dapat ditampilkan dengan baik pada halaman web. Pada tabel ini, pengguna dapat melihat informasi siswa seperti nama, kelas, dan jurusan, serta melakukan aksi edit atau hapus data sesuai kebutuhan.

Data Siswa

Tambah Data

Nama	Kelas	Jurusan	Aksi
mip	-	rpl	Edit Hapus
rita	11	rpl	Edit Hapus
qira	10	TKJ	Edit Hapus

BAB I I- KESIMPULAN

Berdasarkan hasil pembuatan project, aplikasi pengelolaan data siswa berbasis Laravel dan MySQL telah berhasil dibuat dan berjalan sesuai dengan tujuan. Aplikasi ini mampu menjalankan proses CRUD, yaitu menambah, menampilkan, mengubah, dan menghapus data pada bagian data kelas maupun data siswa. Selain itu, aplikasi ini juga telah menerapkan hubungan antar tabel, yaitu antara tabel siswas dan tabel kelas dengan menggunakan foreign key kelas_id. Relasi tersebut membuat data siswa dapat terhubung langsung dengan data kelas, sehingga informasi yang ditampilkan menjadi lebih lengkap dan terstruktur.

Pada project ini, migration berfungsi untuk membentuk struktur tabel di database, model digunakan untuk mengatur data serta relasi antar tabel, controller berperan dalam mengelola alur proses aplikasi, route digunakan untuk menghubungkan alamat URL dengan fungsi pada controller, sedangkan Blade digunakan sebagai tampilan halaman web seperti form input dan tabel data. Dengan adanya pembagian fungsi tersebut, aplikasi menjadi lebih rapi, mudah dipahami, dan lebih mudah dikembangkan kembali.