1. サービスプロパイダー

ファサードのように Laravel の設計の中でついている『ある役割を持ったクラス』の事を『サービスプロパイダー』と呼ぶ。

laravel はサービスごとに『初期処理を定義』し、『実行』する仕組みを持っています。
その仕組みや、『実際に初期処理の実装を行うクラス』の事をサービスプロパイダーと言います。
リクエストが来るたびに config\app.php の provider 配列で定義されたプロバイダーが読み込まれます。(実際は、その各プロパイダーが使われるときに読み込まれます。

もっと細かく言えばサービスプロバイダーはサービスコンテナと呼ばれるものにクラスを登録する機能を持ったもの。
サービスコンテナはいわゆる DI コンテナと呼ばれているものです。
(サービスコンテナはクラス間の依存性を管理する為の仕組み?あとで調べる。)

リクエストが来るたびにルーティングで設定したアクションが実行されますが、そのアクションが実行される前に読み込まれて初期処理を行う役割が『サービスプロパイダー』です。

サービスプロパイダーにはファサードと同じ様に役割ごとに色々なサービスプロパイダーがあり、パスワードの暗号化など暗号化サービスなど暗号化サービスを提供する『EncryptionServiceProvider』などがある。

＊ちなみにファサードは『簡単にサービスプロパイダに設定されたクラスを使える機能』なので、

1.サービスプロパイダでサービスコンテナにクラスを登録し、
2.ファサードでは、その登録されたクラスが簡単に使える

っぽい。

2. プロパイダーファイル内にある register()と boot()について。
register()
サービスプロパイダへの登録を実施する。
それ以外の処理は書いてはいけない。
register()メソッドの定義は必須だけど何も無いなら空のまま置いておいてよい。

例
laravel/framwork/src/Illuminate/Auth/AuthServiceProvider/

```
    public function register()
    {
        $this->registerAuthenticator();
        $this->registerUserResolver();
        $this->registerAccessGate();
        $this->registerRequestRebindHandler();
        $this->registerEventRebindHandler();
    }
```

boot を実行する前にする処理

boot()
boot()ではそのサービス固有の初期処理を自由に実装できます。
サービス固有の初期処理が必要なければ、boot メソッドが無くても構いません。

boot()は、他の全てのサービスプロバイダーの regist()の実行を終えてから呼び出される。

3. DB アクセスを見る
SQL ログを取るようにした上で『app.blade.php』