COMPONENTS

I ***components*** sono dei blocchi di codice che possiamo richiamare quando ci serve nei nostri documenti.
Esempio una navbar o un footer non verrà ripetuto in ogni pagina nella nostra view, ma andrà' richiamato.

Creiamo la cartella *components* all'interno di *views*.

```
/views/components
```

Al suo interno creiamo un file *layout.blade.php* che conterrà tutto l'HTML di base del documento (*head* e *body)*. Sara' la nostra cornice di lavoro.

```
<html>
<head>
</head>
<body>

      {{$slot}}

<script></script>
</body>
</html>
```

Con ***{{slot}}*** mettiamo un segnaposto per indicare lo spazio che le pagine utilizzeranno quando richiameranno il Layout.

I file nelle cartelle views adesso richiameranno quello che e' presente nel file *Layout* attraverso i Tag:

```
<x-layout>

      <paginaHtml/>

</x-layout>
```

<paginaHTML> sara' idealmente all posto di *{{slot}}*
Una volta creata la pagina di Layout, all'interno di *components*, possiamo creare altri *component* e collegarli al Layout, in modo che possano essere ripetuti semplicemente richiamando il Layout e non tutti gli altri.

Esempio la **Navbar** (e il **Footer**).
Con lo stesso procedimento creiamo il file *navbar.blade.php* con all'interno l'HTML corrispondente.
Nel Layout adesso andiamo ad inserire il tag

```
<x-navbar/>
```

NB. Il tag e' scritto nella sua forma ridotta perché' al suo interno non deve essere inserito altro codice.

Con la stessa logica possiamo anche creare dei *components* con altri elementi di un HTML, come ad esempio delle *cards.*
Trasferiamo una card in un nuovo documento *card.blade.php* e nella pagina nel quale vogliamo inserirle mettiamo il tag per richiamarle.

In questo caso vogliamo ciclare con un foreach tutte le cards con i contenuti di un array. Nella pagina nelle views:

```
@foreach ($projects as $project)
    <x-cards
        name="{{project['name']}}"
        surname="{{project['surname']}}"
        description="{{project['description']}}"
        author=”Massimiliano Falcone”
    />
@endforeach
```

In cards.blade.php riempiremo gli spazi che vogliamo riempire di contenuto dell'array in questo modo:

```
@props(['name', 'surname', 'description']}


{{$name}}
{{$surname}}
{{$description}}
{{author}}
```

***NB** Se dichiariamo le proprietà' che vogliamo siano specificate all'interno tramite **@props** facciamo una **dichiarazione di proprietà**. Ogni variabile non specificata dall'esterno non sarà' accettata.*

INSERIRE LOGICA NEI COMPONENTS

Quando inseriamo logica nei Components stiamo dando *logica frontend.* (Quella backend nei controller).

Esempio, vogliamo aggiungere una classe alla navbar che triggeri quando ci troviamo sulla pagina corrispondente.

```
<li class="nav-item"><a class="nav-link js-scroll-trigger
{{Route::currentRouteName()=='about' ? 'active : " "}}">About</a></li>
<li class="nav-item"><a class="nav-link js-scroll-trigger
{{Route::currentRouteName()=='contact' ? 'active : " "}}">Contact</a></li>
```

In questo modo abbiamo aggiunto un *operatore ternario (if)*.

SCAFFOLDING (old version)

il file *app.blade.php* e' il Layout del nuovo metodo.
Al posto del comando *{{$slot}}* utilizziamo:

```
@yield('content)
```

Nel *welcome.blade.php* la sintassi e' diversa:

```
@extends('layouts.app')

@section('content')

    <codicehtml/>

@endsection
```

Per le Navbar/Footer ecc creo una cartella */resources/views/**includes*** , al suo interno *navbar.blade.php* nel quale inserisco il codice della Nav.

Per richiamarlo in *app.blade.php*

```
@include('includes.navbar')
```