BUILDING DEGLI ASSETS

Tramite il building degli assets, possiamo velocizzare i caricamenti del nostro sito importando delle cartelle che riorganizzano i nostri Css e Js.

Diamo i seguenti comandi nella bash:

- `composer laravel/ui`
  Scarichiamo lo scaffolding necessario di Bootstrap e Vue

- `php artisan ui bootstrap --auth`
  Scarichiamo lo scaffolding frontend. Aggiungendo *--auth* scarichiamo anche la logica dell'authentication (Login, Registration ecc)

- `npm install`
  Installiamo sul nostro progetto npm (solo dopo aver installato Node) che ci consente di gestire i pacchetti di Javascript. Quello che installerà lo recupera da package.json creando a sua volta un file package-lock.json con le librerie installate.
  Crea la cartella *node_modules* con tutte le librerie di Js.

- `npm run dev`
  Abbiamo creato in /public la cartella *css* e la cartella *js* con all'interno di entrambe un file *app.css/js*

I file app sono dei file di riferimento che *contengono* un indirizzamento di tutto il js e il css necessario a Bootstrap che si trova nella cartella ***resources***.
Nella cartella ***sass*** gestiremo il *css,* nella cartella ***js*** gestiremo il ***js.***

All'interno delle suddette cartelle posso creare i file *style.css* e *script.js,* e a seguire indichiamo nel file *app* corrispondente ***@import 'styles';*** per creare il collegamento e ***require('./scripts');***

Questo passaggio e' importante perché' il file ***webpack.mix.js*** builda gli assets e crea un unico file css, e un unico file js: quelli contenuti nella cartella *Public.*

- `npm run dev`
  Va lanciato ogni volta che dobbiamo aggiornare i file css e js (buildare gli assets)

- `npm run watch`
  Builda gli assets, ma rimane in ascolto, in modo da poter effettuare modifiche e poterle vedere in tempo reale sul nostro server (php artisan serve). *CMD+C* close.

- `npm run prod`
  Compatta il codice nei file in *public* per rendere la lettura del browser più rapida.
  Va lanciato quando finiamo il nostro lavoro.

Richiamiamo i nostri css e js sull'html nello stesso modo, ma specificando la posizione degli assets corrispondenti.

```
<link rel="stylesheet" href="{{asset('/css/app.css')}}">

<script src="{{asset('/js/app.js')}}"></script>
```

AUTHORIZATION

- *php artisan ui bootstrap* ***--auth***

Con il comando *--auth* abbiamo implementato in *web.php* tutte le rotte di registrazione (login, ecc) e inoltre ha creato un *HomeController.php* oltre ad una cartella ***Auth*** con ogni Controller dedicato ad ogni azione legata alla registrazione di uno User.

In */Http/Models* e' anche presente un file User.php con all'interno una *class User.*

In /resources/views/auth troviamo l'HTML necessario per i vari Login, Register, Verify ecc.

L'*HomeController.php* è stato creato con l'obiettivo di contenere la logica che verrà' eseguita solo se l'Utente è un *Utente Registrato*.
La funziona ***__construct*** al suo interno ha un ***middleware('auth')*** che blocca le request che non sono fatte da un utente registrato.