BP - Slug, Canonical e Return ASAP

Nel *URI* vogliamo inserire il titolo dell'articolo che andiamo a visualizzare nel dettaglio.
Nel Model sviluppiamo una funzione:

```
public function url(){
	return route('article', [$this->ud, \Str::slug($this->title)]);
}
```

Nell' *href* che ci manda al dettaglio dell'articolo, utilizziamo la funzione *url()*:

```
<a href="{{ $article->url() }}"></a>
```

Nella *route* inseriamo la rotta *parametrica* aggiungendo **?** che rende facoltativo quel parametro, o ne rende accettabile uno rotto:

```
Route::get('/articolo/{article_id}/{article_title?}', [HomeController::class,
                                                       'article'])->name('article')
```

Se vogliamo che Google non tenga conto di questa cosa per favorire la SEO, nel *push('styles')* della pagina dell'articolo:

```
<link rel="canonical" href="{{ $article->url() }}">
```

Inoltre per evitare che un utente possa riscontrare *errori* accedendo a pagine di articoli non pubblicati o *non più esistenti,* nella funzione che gestisce la *show* dell'article:

```
public function article($article_id){
	$article = Article::where('id, $article_id')->where('published', true)->first();
	if($article)
		return view('article', compact('article'));

	return redirect('/home');
}
```

BP - UTF-8

Quando creiamo un DB se possiamo scegliere il tipo di *Encoding* scegliamo ***utf8mb4*** che a differenza dell ***utf8*** supporta anche le *emoticons*.

BP - Guarded != Fillable nel Model

Per non dover aggiornare il *$fillable* ogni volta nel *Model* si può utilizzare la logica inversa per cui se nessun campo e' ***$guarded*** tutti gli altri sono *$fillable*.

```
class Article extends Model{
	use HasFactory:

	protected $guarded = [];
}
```