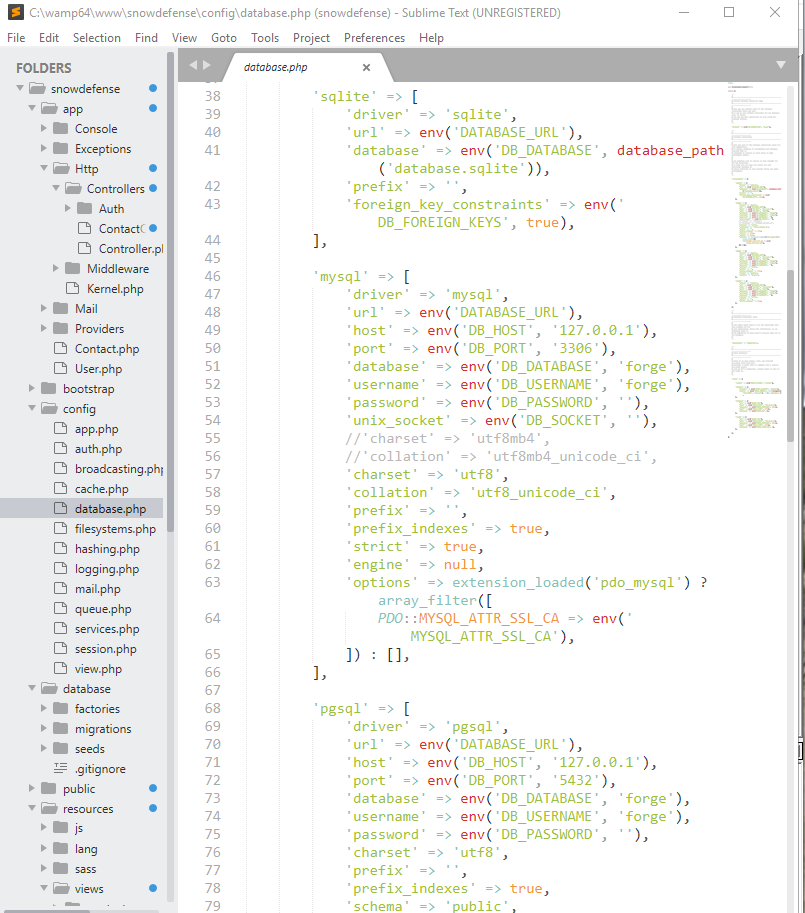
LARAVEL – how to fix database issue so migration error won’t occur:

(notice charset and collation, the commented ones are the old ones, new code below it)



ALSO, READ THIS:

<https://phplaravel.wordpress.com/2016/07/06/using-mysql-utf8mb4-character-set-and-collation-with-laravel/>

**THE PROBLEM**

MySQL has long provided a utf8 character set.  More recently (as of mysql 5.5.3), utf8mb4 character set and corresponding collations have become available.  Utf8mb4 allows for expansive character support and is generally better than it’s regular utf8 alternative.

All of my recent projects are using utf8mb4 character sets and utf8mb4\_general\_ci collations.

I’ve recently began learning Laravel (Version 5.2 (I’m loving it!)) and encountered a problem when I have my app using mysql database with utf8mb4 character set and utf8mb4\_general\_ci collation as shown below from ***/config/database.php*.**

 'mysql' => [

'driver' => 'mysql',

'host' => env('DB\_HOST', 'localhost'),

'port' => env('DB\_PORT', '3306'),

'database' => env('DB\_DATABASE', 'forge'),

'username' => env('DB\_USERNAME', 'forge'),

'password' => env('DB\_PASSWORD', ''),

'charset' => 'utf8mb4',

'collation' => 'utf8mb4\_general\_ci',

'prefix' => '',

'strict' => false,

'engine' => null,

],

When I run the artisan migrate command php artisan migrate I am presented with the following error and the tables are not completely set up.

[Illuminate\Database\QueryException]

SQLSTATE[42000]: Syntax error or access violation: 1071 Specified key was t

oo long; max key length is 767 bytes (SQL: alter table `users` add unique `

users\_email\_unique`(`email`))

[PDOException]

SQLSTATE[42000]: Syntax error or access violation: 1071 Specified key was t

oo long; max key length is 767 bytes

**THE SOLUTION**

Thankfully, the solution was easy to come up with.  And, since I didn’t find the solution after a quick web search, I decided to register here and make a quick blog post for others that encounter the same problem.

In a nutshell, since mysql reserves 4 bytes per utf8mb4 character, the Laravel provided default of 255 characters for the field length exceeds mysql’s key length limit.

When using utf8mb4 the maximum length of any varchar length field is 191 bytes.  Thus, the SQL schema generated by Laravel needs to have any varchar fields maximum length set to 191.

To do this is very easy.  I feel like I shouldn’t be editing the framework’s code, and there probably is a better solution that doesn’t require one to do so, but, I am in a hurry (as always) and the following simple fix works.

**Step 1:** If you’ve already tried running php artisan migrate from the terminal and received the error messages shown above, go ahead and try running php artisan migrate:rollback (warning: this will drop tables so if you have any custom tables or data in your database, save it first).  If you are informed by artisan that there isn’t anything to rollback, go ahead and drop the partially created tables (via mysql cli, phpmyadmin, whatever).

**Step 2:** Open ***/vendor/laravel/framework/src/Illuminate/Database/Schema/BluePrint.php*** in your favorite text editor.

**Step 3:** Change the instances of 255 to 191 (currently lines 439 and 451 for this version of Laravel) as shown below:

/\*\*

\* Create a new char column on the table.

\*

\* @param string $column

\* @param int $length

\* @return \Illuminate\Support\Fluent

\*/

public function char($column, $length = 191)

{

return $this->addColumn('char', $column, compact('length'));

}

/\*\*

\* Create a new string column on the table.

\*

\* @param string $column

\* @param int $length

\* @return \Illuminate\Support\Fluent

\*/

public function string($column, $length = 191)

{

return $this->addColumn('string', $column, compact('length'));

}

**Step 4:** Save the file.

**Step 5:** Return to your terminal and run the migrate command php artisan migrate.

**Step 6:** Success!  This fixes this problem and any problem in the future when a table requires a varchar field with more than 191 characters.

**IN CONCLUSION**

I feel like this is a pretty major thing, as utf8mb4 is an improvement over the utf8 implementation and allows for full utf8 character support, and Laravel should’ve accounted for this, but maybe there is a good reason utf8mb4 maximum key lengths aren’t implemented or checked for.  If there is reasoning and you know why, please leave a comment below.

If I’ve helped you, please leave a comment and let me know!  I don’t write often, as I’m not sure I’m good at writing or sharing information, but to be honest, it has been pretty enjoyable making this post.  Maybe I’ll write more articles in the future!

**HAPPY DEVELOPING!**