**Laravel Multi Tenant App**

1. Database.php: Set multiple connection as per the requirement.

'connections' => [

'master' => [

'driver' => 'mysql',

'url' => env('DATABASE\_URL'),

'host' => env('DB\_HOST', '127.0.0.1'),

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

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

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

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

'unix\_socket' => env('DB\_SOCKET', ''),

'charset' => 'utf8mb4',

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

'prefix' => '',

'prefix\_indexes' => *true*,

'strict' => *false*,

'engine' => 'InnoDB',

'options' => *extension\_loaded*('pdo\_mysql') ? *array\_filter*([

PDO::MYSQL\_ATTR\_SSL\_CA => env('MYSQL\_ATTR\_SSL\_CA'),

]) : [],

],

'tenant' => [

'driver' => 'mysql',

'url' => env('DATABASE\_URL'),

'host' => env('DB\_HOST', '127.0.0.1'),

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

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

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

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

'unix\_socket' => env('DB\_SOCKET', ''),

'charset' => 'utf8mb4',

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

'prefix' => '',

'prefix\_indexes' => *true*,

'strict' => *false*,

'engine' => 'InnoDB',

'options' => *extension\_loaded*('pdo\_mysql') ? *array\_filter*([

PDO::MYSQL\_ATTR\_SSL\_CA => env('MYSQL\_ATTR\_SSL\_CA'),

]) : [],

],

],

1. Database migration: Create Migration respective to connections.

Master, Tenant:

Ex: Create Table

Schema::*connection*('master')->create('master\_login', *function* (Blueprint $table) {

1. Database Models: Specify connection to use in model.

Master Login Table Ex:

*<?php*

*namespace* DynamicGiftPromotions;

*use* Illuminate\Database\Eloquent\Model;

*use* Illuminate\Foundation\Auth\User *as* Authenticatable;

*use* Illuminate\Notifications\Notifiable;

*use* phpDocumentor\Reflection\Types\Self\_;

*class* MasterLogin *extends* Authenticatable

{

*use* Notifiable;

*protected* $connection = 'master';

*protected* $table = 'master\_login';

*/\*\**

*\* The attributes that are mass assignable.*

*\**

*\** ***@var*** *array*

*\*/*

*protected* $guarded = [];

*/\*\**

*\* The attributes that should be hidden for arrays.*

*\**

*\** ***@var*** *array*

*\*/*

*protected* $hidden = [

'password', 'remember\_token', 'database\_password',

];

*//user is super admin or not.*

*public function* isSuperAdmin()

{

*return* $this->user\_role === config('params.user\_role.super\_admin');

}

}

1. Artisan Console Commands: Run artisan command to create tenant database.

Create Tenant DB:

*<?php*

*namespace* DynamicGiftPromotions\Console\Commands;

*use* Illuminate\Console\Command;

*use* Illuminate\Support\Facades\DB;

*use* Illuminate\Support\Facades\Config;

*class* CreateTenantDatabase *extends* Command

{

*/\*\**

*\* The name and signature of the console command.*

*\**

*\** ***@var*** *string*

*\*/*

*protected* $signature = 'Create:TenantDB {db\_name} {db\_username}';

*/\*\**

*\* The console command description.*

*\**

*\** ***@var*** *string*

*\*/*

*protected* $description = 'Command to create tenant database.';

*/\*\**

*\* Create a new command instance.*

*\**

*\** ***@return*** *void*

*\*/*

*public function* \_\_construct()

{

*parent*::*\_\_construct*();

}

*/\*\**

*\* Execute the console command.*

*\**

*\** ***@return*** *mixed*

*\*/*

*public function* handle()

{

*try* {

$db\_name = $this->argument('db\_name');

$username = $this->argument('db\_username');

$password = '12345678';

$this->createDatabaseSchema($db\_name);

$this->createDatabaseUser($username, $password);

$this->grantPrivilegeToUser($db\_name, $username);

$this->configureConnection($username, $password, $db\_name);

*//$this->sqlModeChange();*

$this->call('migrate', *array*('--database' => 'tenant', '--path' => 'database/migrations/tenant'));

} *catch* (Exception $ex) {

*echo* $ex->getMessage();

*exit*;

}

}

*/\*\**

*\* Creates a new database schema.*

*\** ***@param*** *string $schemaName The new schema name.*

*\** ***@return*** *bool*

*\*/*

*private function* createDatabaseSchema($schemaName)

{

*//`statement` method from the connection class for create database*

*return* DB::connection('master')->statement('CREATE DATABASE ' . $schemaName . ' DEFAULT CHARACTER SET utf8;');

}

*/\*\**

*\* Creates a new database user.*

*\** ***@param*** *string $username ,$password.*

*\** ***@return*** *bool*

*\*/*

*private function* createDatabaseUser($username, $password)

{

*return* DB::connection('master')->statement('CREATE USER \'' . $username . '\'@\'localhost\' IDENTIFIED BY \'' . $password . '\';');

}

*/\*\**

*\* Grant privilege to created user*

*\** ***@param*** *string $username ,$password.*

*\** ***@return*** *bool*

*\*/*

*private function* grantPrivilegeToUser($db\_name, $username)

{

*return* DB::connection('master')->statement('GRANT ALL PRIVILEGES ON ' . $db\_name . '.\* TO \'' . $username . '\'@\'localhost\' WITH GRANT OPTION;');

*//return DB::connection('master')->statement("GRANT ALL PRIVILEGES ON \* . \* TO \'" . $username . "\'@\'localhost\' WITH GRANT OPTION MAX\_QUERIES\_PER\_HOUR 0 MAX\_CONNECTIONS\_PER\_HOUR 0 MAX\_UPDATES\_PER\_HOUR 0 MAX\_USER\_CONNECTIONS 0;");*

}

*/\*\**

*\* Configures a tenant's database connection.*

*\** ***@param*** *string $db\_name The database name.*

*\** ***@return*** *void*

*\*/*

*private function* configureConnection($username, $password, $db\_name)

{

Config::set('database.connections.tenant',

[

"host" => '127.0.0.1',

"database" => $db\_name,

"username" => $username,

"password" => $password,

"driver" => 'mysql',

]);

DB::*purge*('tenant');

DB::setDefaultConnection('tenant');

}

*//allow all Sql function like : group by etc..*

*// private function sqlModeChange()*

*// {*

*// return DB::connection('master')->statement('SET GLOBAL sql\_mode=\'NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_AUTO\_CREATE\_USER,NO\_ENGINE\_SUBSTITUTION\';');*

*// }*

}

1. Laravel queue Setup: Setup Laravel Queue in master and managed all emails links respective to database.

'database' => [

'driver' => 'database',

'connection' => 'master',

'table' => 'jobs',

'queue' => 'default',

'retry\_after' => 90,

],

1. Config: Auth.php : Setup table to use for authenticate and forgot password.

'guards' => [

'web' => [

'driver' => 'session',

'provider' => 'master',

],

'api' => [

'driver' => 'token',

'provider' => 'master',

'hash' => *false*,

],

],

'providers' => [

'users' => [

'driver' => 'eloquent',

'model' => DynamicGiftPromotions\User::*class*,

],

'master' => [

'driver' => 'eloquent',

'model' => DynamicGiftPromotions\MasterLogin::*class*,

],

],

1. Connection Middleware: Set database connection respective to user on every http request.

*<?php*

*namespace* DynamicGiftPromotions\Http\Middleware;

*use* Closure;

*use* Illuminate\Support\Facades\Config;

*use* Illuminate\Support\Facades\DB;

*class* SetDatabaseConnection

{

*/\*\**

*\* Handle an incoming request.*

*\* added to web middleware in kernel.php*

*\**

*\** ***@param*** *\Illuminate\Http\Request $request*

*\** ***@param*** *\Closure $next*

*\** ***@return*** *mixed*

*\*/*

*public function* handle($request, Closure $next)

{

*if* (auth()->user()) {

Config::set('database.connections.tenant',

[

"host" => '127.0.0.1',

"database" => auth()->user()->database\_name,

"username" => auth()->user()->database\_username,

"password" => auth()->user()->database\_password,

"driver" => 'mysql',

]);

DB::*purge*('tenant');

DB::setDefaultConnection('tenant');

}

*return* $next($request);

}

}