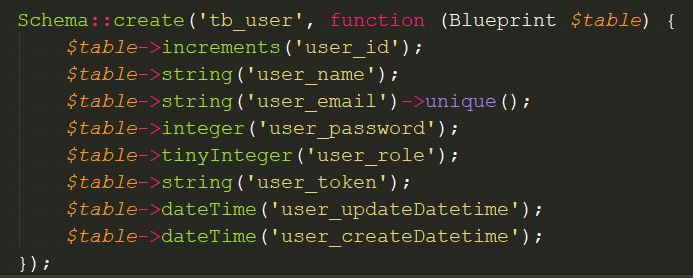
Chức năng - Exp

1. Login – Login – Auth Api Passport

Chạy migration để install 1 số table của passport, user

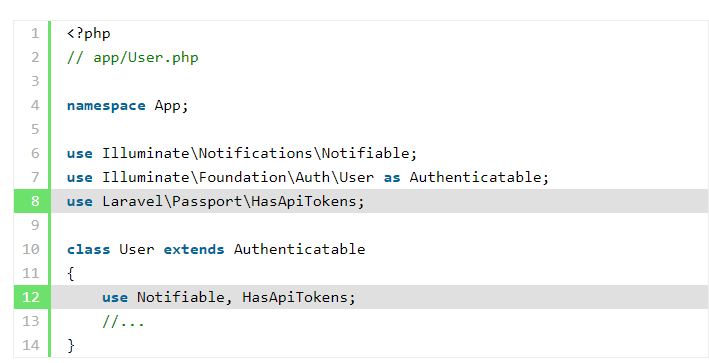


Tạo controller Auth để viết function login, logout

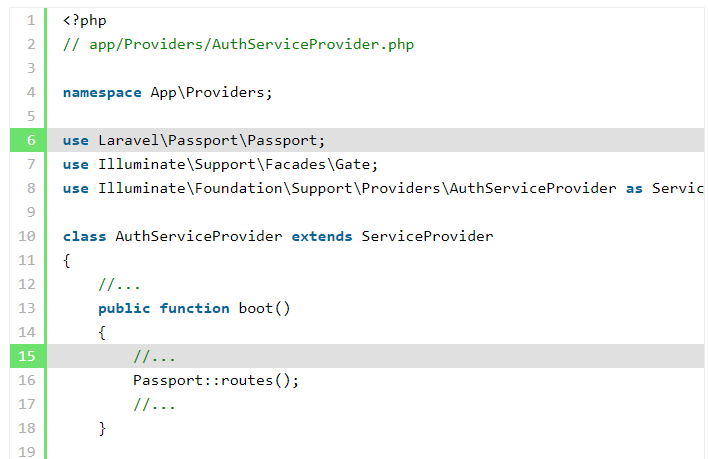
Add thư viện passport vào model

+ use Laravel\Passport\HasApiTokens;

+ use HasApiTokens;



Mở class **App\Providers\AuthServiceProvider**, thêm  **Passport::routes()** vào method **boot()**



Chỉnh route cho auth api

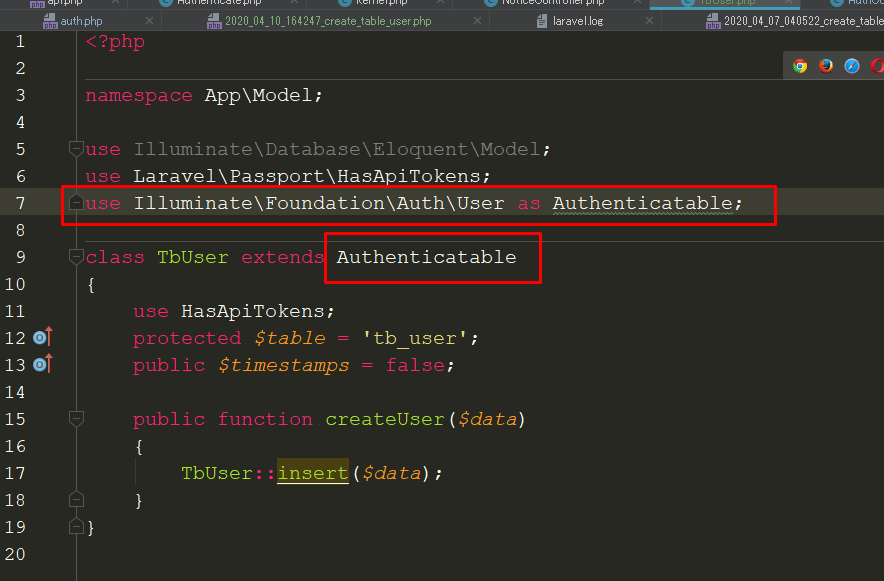


Với

+ prefix: thêm path trên url, bỏ cũng được

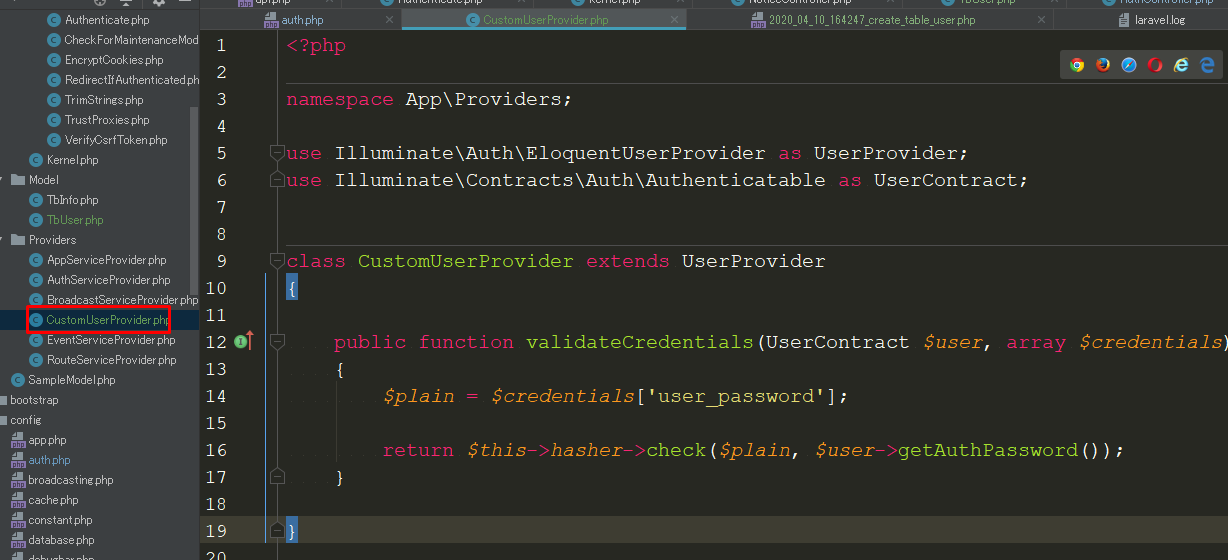
+ middleware auth:api : Check xem có login không, có thì pass, không thì error

1. **Change extends table khi sử dụng authen của laravel**

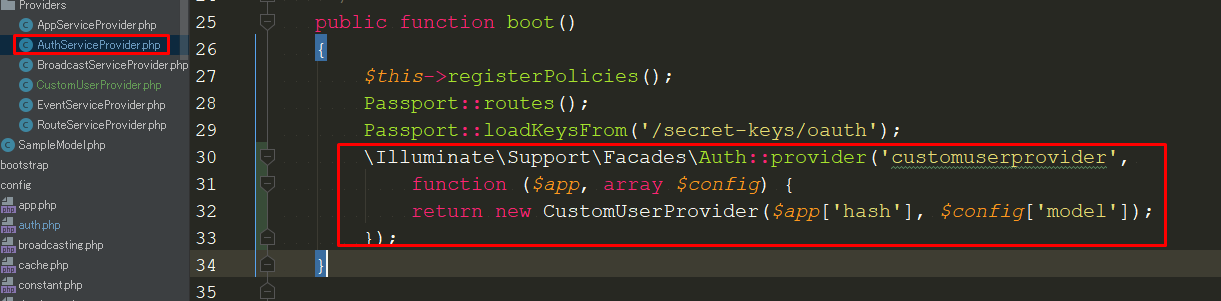


1. **Change provider của laravel**

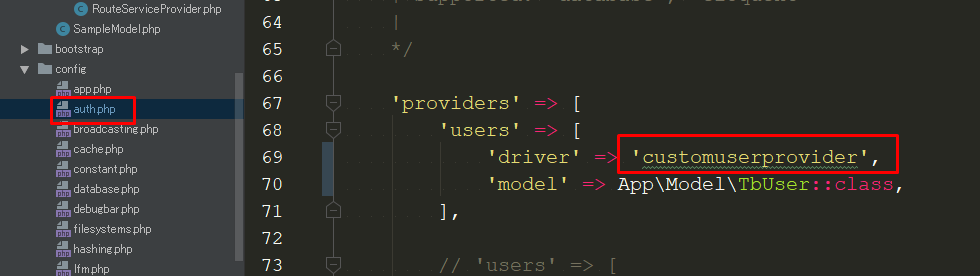
Laravel mặc định sẽ sử dụng table user của họ, khi mình đổi theo db của mình thì cần override lại mặc định của họ.



Sau đó add provider vừa được tạo vào AuthServiceProvider để khai báo



Sau đó add name provider vừa khai báo vào config.auth



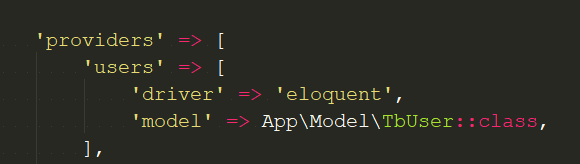
Note:

Config/auth.php:

+ Guard cung cấp session guard hoặc token guard.

* Session guard duy trì trạng thái người dùng trong mỗi lần request bằng cookie
* Token guard xác thực người dùng bằng cách kiểm tra token hợp lệ trong mỗi lần request (So sánh token người dung push lên và db)

+ Providers lấy ra dữ liệu người dùng từ phía back-end (Nôm na là query với db)



Khi login, server tạo session, client lưu session id bằng cookie, khi cookie hết hạn thì logout.

Vậy nên muốn ghi nhớ login thì server sẽ set thêm time sống ứng với cookie của user đó.

NOTE: Khi chạy nhầm composer update thì revert file, sau đó chạy composer install lại, khi đó nó sẽ đọc data từ composer.lock ra để cài đặt lại.

1. Zip Folder

Source Common function:

*$zipFile* = (isset(*$data*['zipName']) ? *$data*['zipName'] : *date*('Ymd-his')) . '.zip';  
*$zip* = new ZipArchive();  
*$zip*->open(*$zipFile*, ZipArchive::*CREATE* | ZipArchive::*OVERWRITE*);  
  
*$paths* = isset(*$data*['zipPath']) ? *$data*['zipPath'] : storage\_path();  
foreach (*$paths* as *$path*) {  
 *$files* = new RecursiveIteratorIterator(new RecursiveDirectoryIterator(*$path*));  
 foreach (*$files* as *$name* => *$file*) {  
 if (!*$file*->isDir()) {  
 *$filePath* = *$file*->getRealPath();  
 *$relativePath* = *substr*(*$filePath*, *strlen*(*$path*) + 1);  
 *$zip*->addFile(*$filePath*, *$relativePath*);  
 }  
 }  
}  
*$zip*->close();

Call Common function

*$this*->Common->zipFolderFile(*$request*->all());  
return *$this*->Common->ApiReturnSuccess();