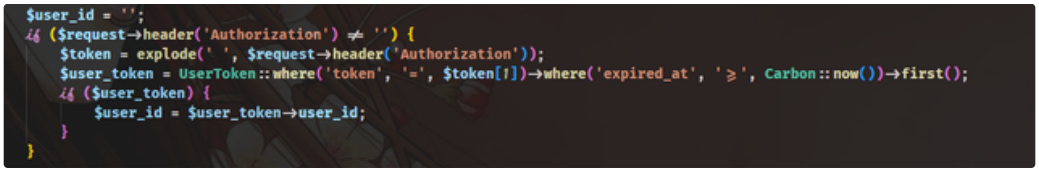
User Profile Show

Ruta: (get) api/v1/user/profile (middlewares que observan esta ruta [`AuthenticationChannel`], [`Authentication`]), la cual apunta al controlador UserController, método profile.

Primero de todo se extrae el token a raíz del user token proporcionado en el header, la lógica de la extracción del user/compañía ligada al user es la misma que la clase del middleware a la cual hace referencia el alias Authentication por lo tanto revisarlo si no se conoce este paso (img 1) que se repite en todos los método que se necesita que la verificación del user, compañía del usuario que realiza la consulta, operaciones de create/delete, etc.



img 1

Muestra la información contenida de las tablas relacionadas con la información básica del usuario en cuestión las cuales son :

lp_users → Data del user.

lp_companies → Data de la compañia a la que pertenece el usuario.

lp_files → Data relacionada con la imagen del usuario y de la compañia.

lp_user_points → data realcionada con los puntos que tiene un usuario.

lp_user_wallet_virtuals → data relacionada con la cantidad disponible para gastar que tiene un usuario.

En (Img 2) creamos en las variables la futura data que usaremos para la extracción de la información generando un único registro de usuario igualando buscando que en la tabla users la id del usuario sea igual del que hizo la consulta y a partir de este único registro a través de la fk del company y file sacar el registro que coincida de todos los registros guardados en la variable sql_file y sql_company, para extraer además la información de la imagen de la compañía se realiza el mismo paso que con usuario y file, accedemos a la fk de file en compañía y extraemos la info de la pk en file a la cual hace referencia, dichas relaciones entre tablas se pueden apreciar en la generacion de la respuesta (img 3 e img 4).

```php
$data_user = DB::table('lp_users')
    ->select(
        'lp_users.id',
        'lp_users.name',
        'lp_users.surname',
        'lp_users.email',
        'lp_users.image_id',
        'lp_users.registration_date',
        'lp_users.company_id',
        'lp_users.social_register',
        'lp_users.profile_completed'
    )
    ->whereNull('lp_users.deleted_at')
    ->where('lp_users.id', '=', $user_id)
    ->first();

// Precompiled SQL
$sql_file = DB::table('lp_files')
    ->select(
        'id',
        'name',
        'url'
    )
    ->whereNull('deleted_at');

$sql_company = DB::table('lp_companies')
    ->select(
        'id',
        'name',
        'description',
        'short_description',
        'image_id'
    )
    ->whereNull('deleted_at');
```

img 2

```php
$array['data'] = array();

$array_user = array();
$array_user['id'] = $data_user->id;
$array_user['name'] = $data_user->name;
$array_user['surname'] = $data_user->surname;
$array_user['email'] = $data_user->email;
$array_user['registration_date'] = $data_user->registration_date;
$array_user['social_register'] = $data_user->social_register;
$array_user['profile_completed'] = $data_user->profile_completed;

$array_user['file']['image'] = array();
if ($data_user->image_id != '' && $data_user->image_id != null) {
    $sql_user_image = clone $sql_file;
    $data_user_image = $sql_user_image->where('lp_files.id', $data_user->image_id)->first()
    if ($data_user_image) {
        $array_file = array();
        $array_file['id'] = $data_user_image->id;
        $array_file['name'] = $data_user_image->name;
        $array_file['url'] = Storage::url($data_user_image->url);
        $array_user['file']['image'][] = $array_file;
    }
}

$array_user['company'] = null;
if ($data_user->company_id != '' && $data_user->company_id != null) {
    $sql_user_company = clone $sql_company;
    $data_user_company = $sql_user_company->where('id', $data_user->company_id)->first();

    if ($data_user_company) {
        $array_company = array();
        $array_company['id'] = $data_user_company->id;
        $array_company['name'] = $data_user_company->name;
        $array_company['description'] = $data_user_company->description;
        $array_company['short_description'] = $data_user_company->short_description;
```

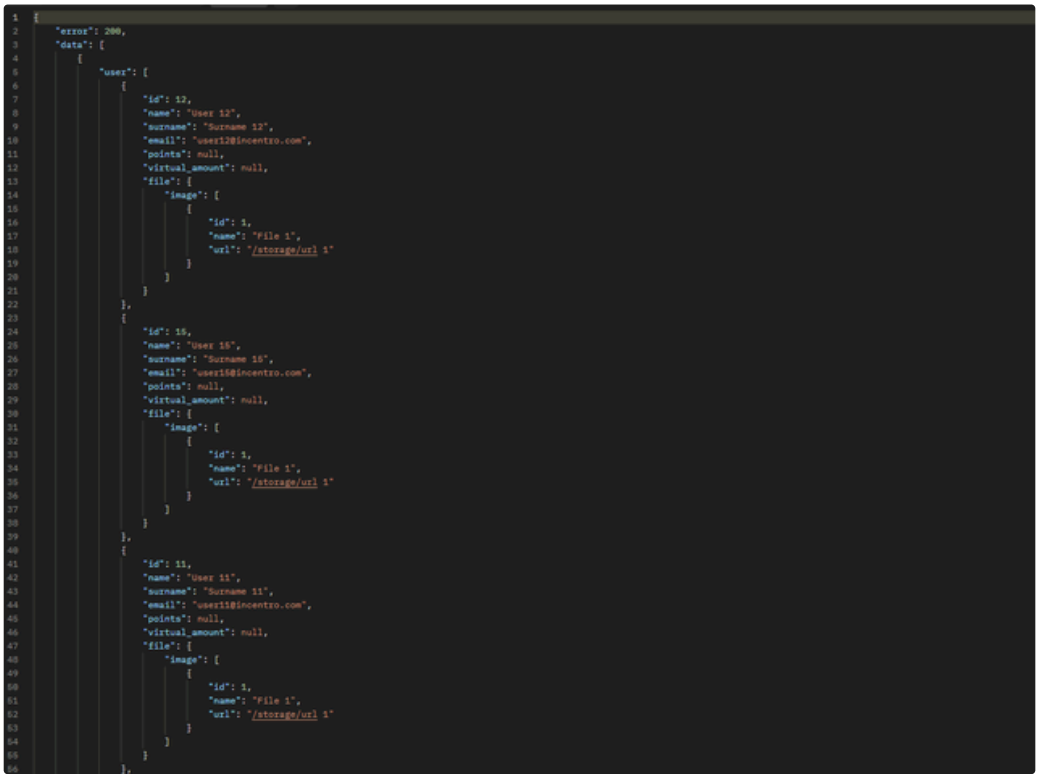
img 3

```php
        $array_company['file']['image'] = array();
        if ($data_user_company->image_id != '' && $data_user_company->image_id != null) {
            $sql_image_clone = clone $sql_file;
            $data_image = $sql_image_clone->where('lp_files.id', $data_user_company->image_id)->first();
            if ($data_image) {
                $array_image = array();
                $array_image['id'] = $data_image->id;
                $array_image['name'] = $data_image->name;
                $array_image['url'] = Storage::url($data_image->url);
                $array_company['file']['image'][] = $array_image;
            }
        }

        $array_user['company'] = $array_company;
    }
}

$array['data'][0]['user'][] = $array_user;


DB::commit();
tch (\Exception $e) {
DB::rollBack();
$response['status'] = $e->getCode() ?: 500;
$response['message'] = $e->getMessage() ?: 'Error getting user profile.';
Log::error('Error in profile method: ' . $e->getMessage());
```

img 4

Por último añadir que las validaciones de entrada son las genéricas, que se encuentre registro antes de iterar, que los valores no sean null etc, por último adicionalmente a la respuesta en caso de ok/ko que recibirá el front, se realiza un escrito de error en el Log de Laravel para el backend con referencia al donde se ha producido el error

En (img 5) se puede ver un ejemplo de una consulta realizada correctamente con el formato del json que es enviado como respuesta tras el ok.



```
{
    "error": 200,
    "data": [
        {
            "user": [
                {
                    "id": 12,
                    "name": "User 12",
                    "surname": "Surname 12",
                    "email": "user12@incentro.com",
                    "points": null,
                    "virtual_amount": null,
                    "file": {
                        "image": [
                            {
                                "id": 1,
                                "name": "File 1",
                                "url": "/storage/url 1"
                            }
                        ]
                    }
                },
                {
                    "id": 15,
                    "name": "User 15",
                    "surname": "Surname 15",
                    "email": "user15@incentro.com",
                    "points": null,
                    "virtual_amount": null,
                    "file": {
                        "image": [
                            {
                                "id": 1,
                                "name": "File 1",
                                "url": "/storage/url 1"
                            }
                        ]
                    }
                },
                {
                    "id": 11,
                    "name": "User 11",
                    "surname": "Surname 11",
                    "email": "user11@incentro.com",
                    "points": null,
                    "virtual_amount": null,
                    "file": {
                        "image": [
                            {
                                "id": 1,
                                "name": "File 1",
                                "url": "/storage/url 1"
                            }
                        ]
                    }
                },
```


img 5