

Parcours Développeur d'application - PHP / Symfony

Créer un web service exposant une API

Objectifs

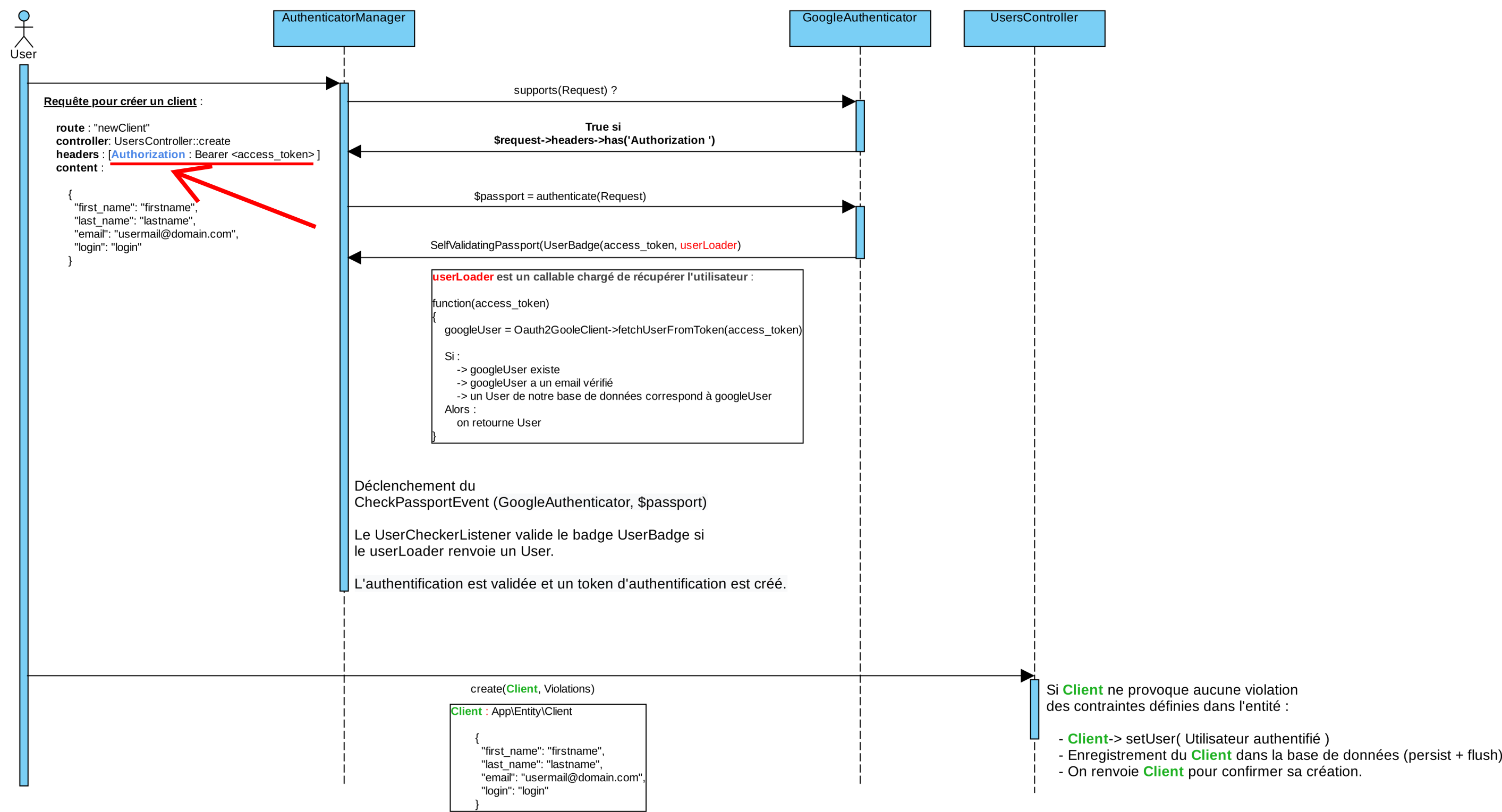
API permettant aux entreprises de consulter un catalogue de smartphones et gérer des clients.

- consulter la liste des produits BileMo ;
- consulter les détails d'un produit BileMo ;
- consulter la liste des utilisateurs inscrits liés à un client sur le site web ;
- consulter le détail d'un utilisateur inscrit lié à un client ;
- ajouter un nouvel utilisateur lié à un client ;
- supprimer un utilisateur ajouté par un client.

Seuls les clients référencés peuvent accéder aux API.

Les clients de l'API doivent être authentifiés via OAuth ou JWT.

Un access token sera nécessaire



Get your access token (need a Google account)

GET /getToken 



Smartphones



GET /phones  


GET /phone/{id}  



Your clients

GET /clients  

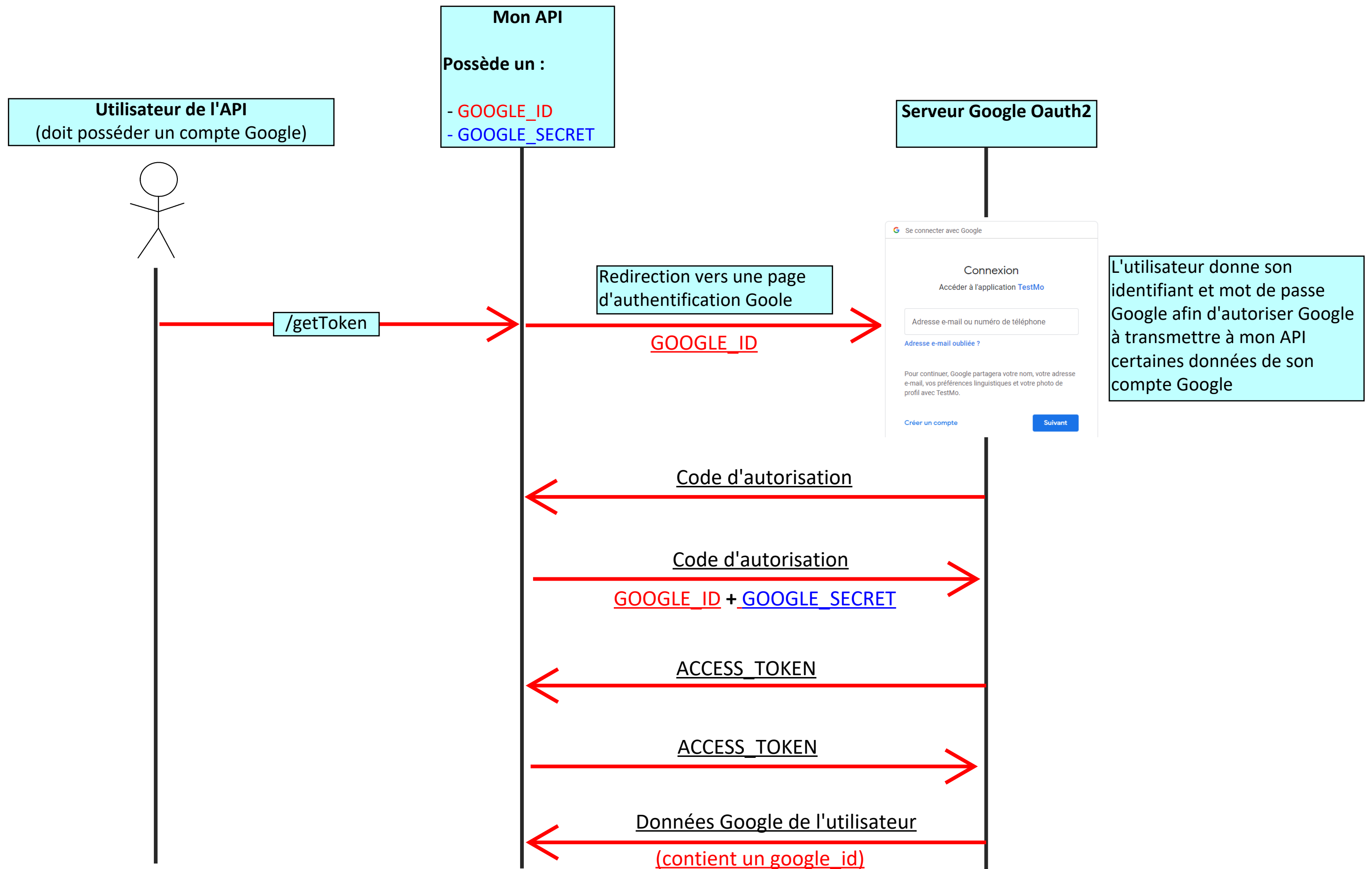
GET /client/{id}  

PUT /client/{id}  

DELETE /client/{id}  

POST /client/new  

Principe de l'authentification OAuth



On vérifie dans notre base de données si un user possède ce google_id

id	email	roles	password	token	name	google_id
1	kawaMobile@domain.com	["ROLE_USER"]	\$2y\$13\$76iSV...	gxu7Z2Hp...	Kawa Mobiles	57416552...
2	DualTechno@domain.fr	["ROLE_USER"]	\$2y\$13\$hojD...	VzQyoPxD...	Dual Tech	31923695...
3	aquaPhone@domain.com	["ROLE_USER"]	\$2y\$13\$4Sc8r...	HgvF75U7...	Aqua Phone	55971448...



Si oui, on transmet l'access token à l'utilisateur

Service permettant d'indiquer à l'utilisateur son access token (route /getToken)

```
public function get()
{
    $userRepository = $this->em->getRepository(User::class);
    $accessToken = $this->client->getAccessToken();
    $accessTokenValue = $accessToken->getToken();

    /** @var GoogleUser $googleUser */
    $googleUser = $this->client->fetchUserFromToken($accessToken);

    if ($googleUser->toArray()['email_verified']) {

        /** @var User $existingUser */
        $existingUser = $userRepository->findOneBy(['googleId' => $googleUser->getId()]);

        if ($existingUser) {
            $existingUser->setToken($accessTokenValue);
            $this->em->flush();
        } else {
            // Any Google user is allowed to register to test this API
            $newUser = new User();
            $newUser->setGoogleId($googleUser->getId())
                ->setEmail($googleUser->getEmail())
                ->setName($googleUser->getName())
                ->setRoles(["ROLE_USER"])
                ->setToken($accessTokenValue);
            $this->em->persist($newUser);
            $this->em->flush();
            // throw new ResourceNotFoundException("You are not registered in our database");
        }
        return $accessTokenValue;
    }
}
```

Retour page 2

Méthodologie

En local

Création d'une nouvelle branche <nom branche>

On publie cette branche sur le dépôt distant

On se place sur la nouvelle branche

Sur la nouvelle branche, on enchaîne le cycle :

git add .

git commit -m <nom_commit>

git push origin <nom branche>

Sur le dépôt distant (GitHub)

On soumet la Pull Request

On approuve la Pull Request

On la merge sur la branche <main>

On supprime l'ancienne branche

En local

On se place sur la branche <main>

On récupère la dernière version de la branche <main>

On supprime la branche <nom branche>

Liste de mes commits

<https://github.com/Benlasc/BileMo/commits/main>

Librairies utilisées

Authentication Oauth2 avec Google -> knpuniversity/oauth2-client-bundle
-> thephpleague/oauth2-google

Création de l'API REST -> FriendsOfSymfony/FOSRestBundle

Serializer -> jms/serializer-bundle

Documentation de l'API -> nelmio/NelmioApiDocBundle