

Documentation technique

ECF : Groupe hôtelier


Florent Malo

Session Juin 2022

[Lien Ondrive du fichier](#) :

A – Spécification Technique

Environnement en Locale

- Serveur : Apache2
- Os : windows
- Xampp
- Composer dans sa dernière version
- Php via  v7.4.3
- Php PDO
- Nodejs : via npm pour le build de la partie Front-end

Front-end :

- Bootstrap v5
- webpack-encore
- CSS3 : Sass
- JavaScript

Versionning

- github: <https://github.com/FofoMalo/hypnosEcf>
- workflow continue intégration :

Back-end :

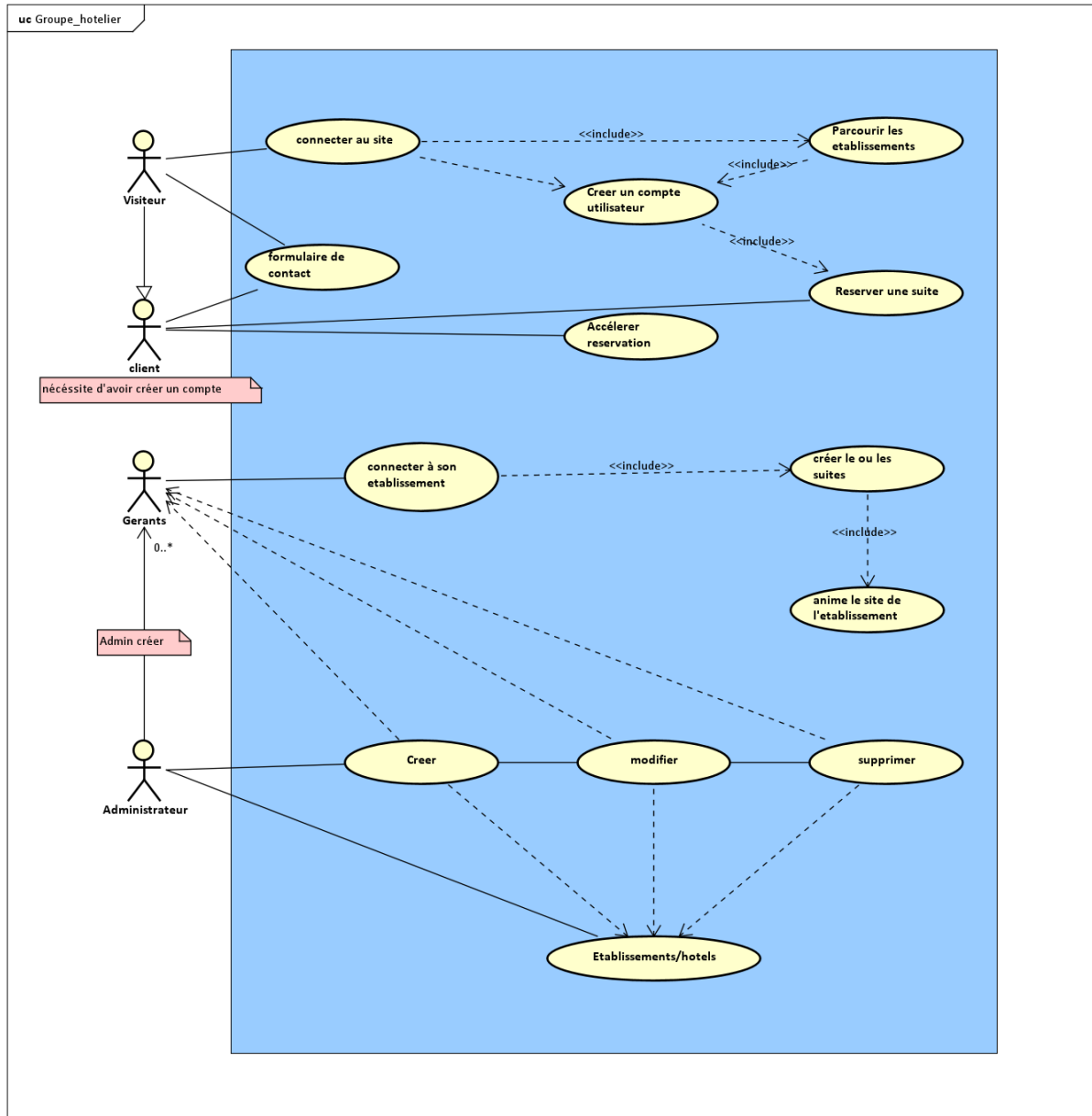
- **Framework** : Symfony et Symfony/cli
- Doctrine ORM
- MySQL
- faker/factory pour les fixtures.

Deploiement

- Heroku

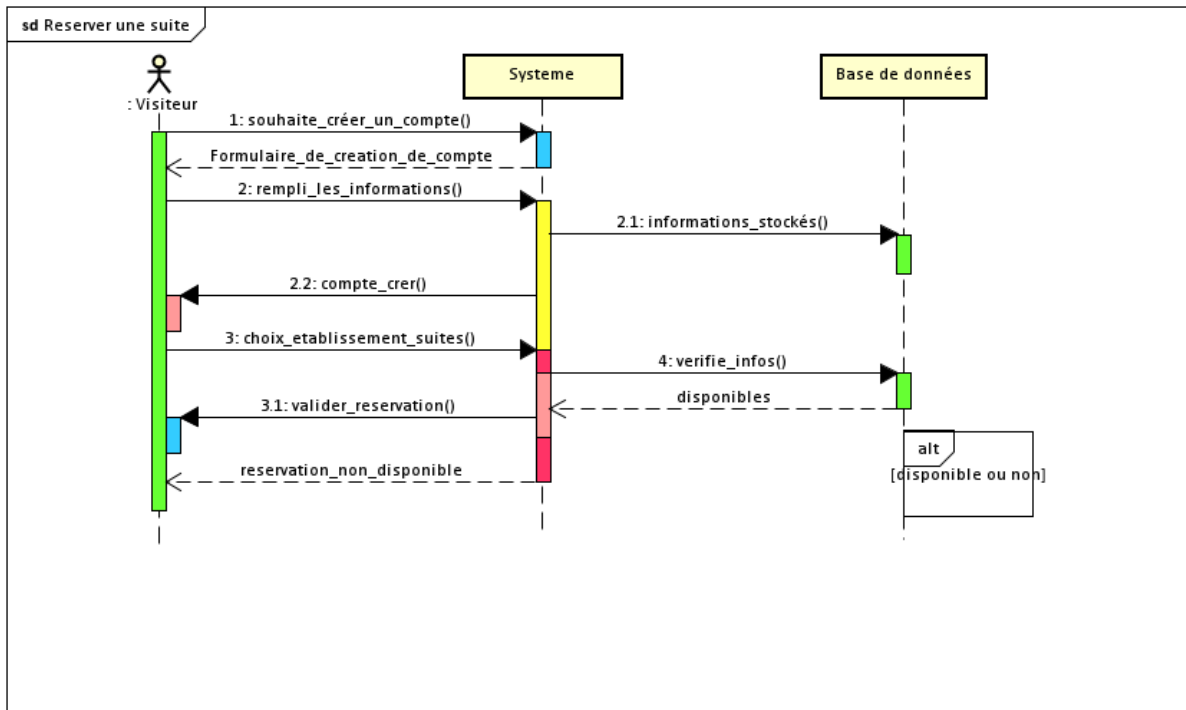
Documentation Technique

Diagramme de cas d'utilisation :

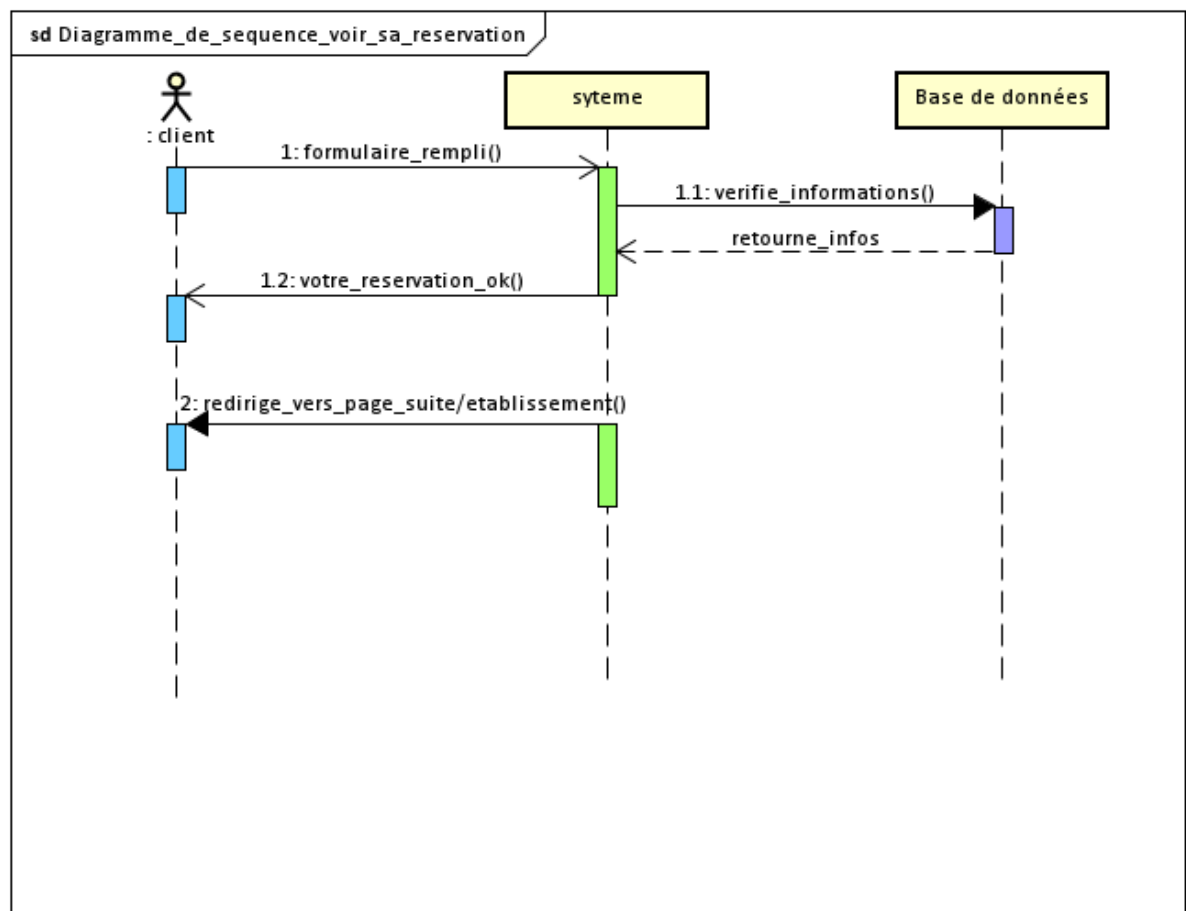


B DIAGRAMME DE SEQUENCE :

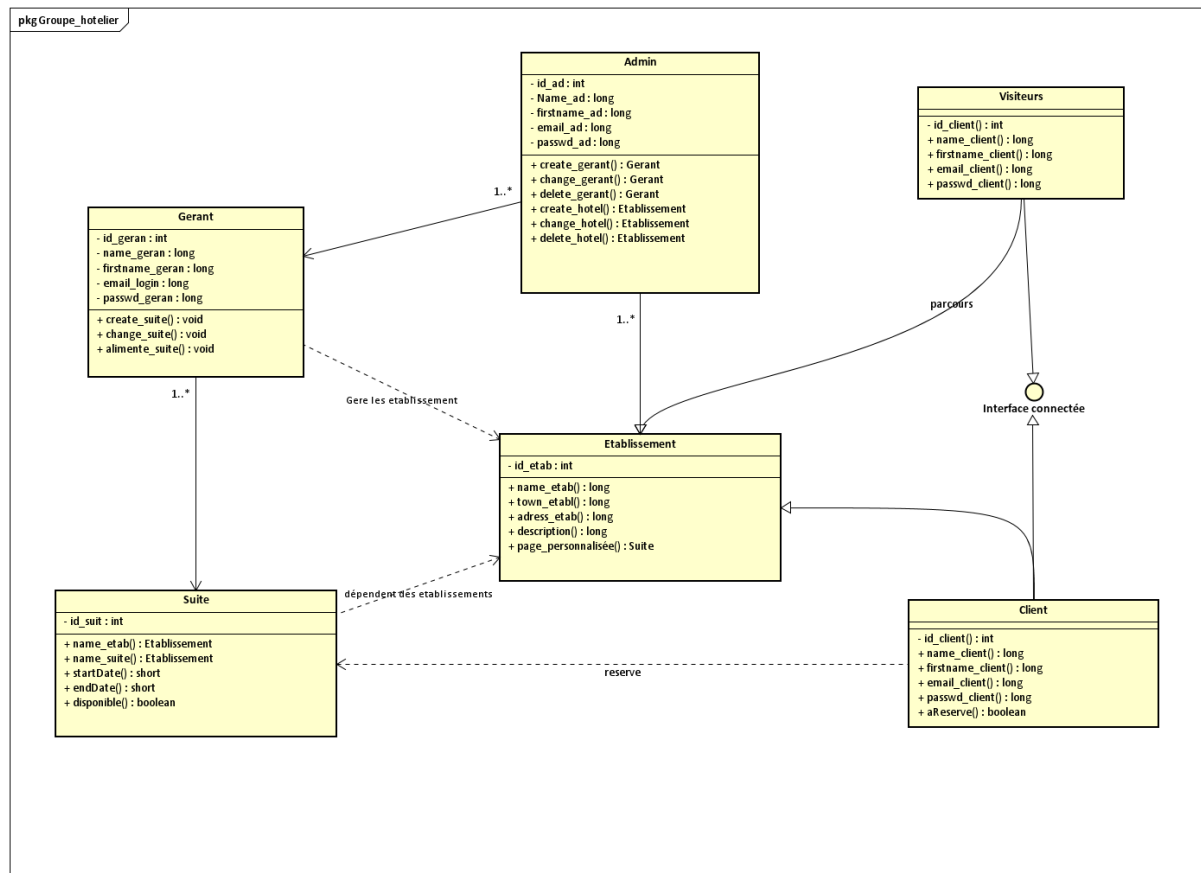
RESERVER UNE SUITE



VOIR SA RESERATION :

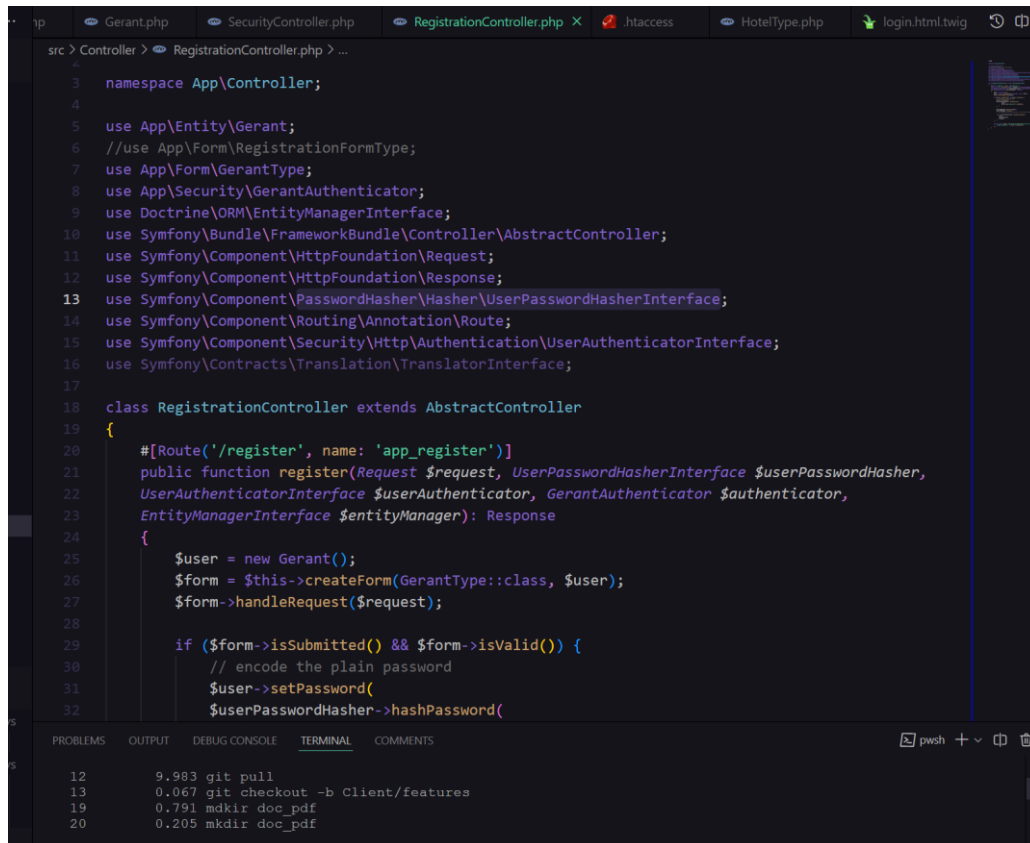


B Diagramme de Class :



Pratique de Sécurité mise en place :

1 – Pour la gestion de la sécurité j'ai utilisé j'utilise la méthode de hash du mot de passe. La fonction register utilise le PasswordHasher\Hasher. Cette option est fourni par dans la composante symfony lorsque nous créons l'auteur avec la commande : `make :auth`.



The image shows a code editor with a dark theme. The top tab bar shows several files: Gerant.php, SecurityController.php, RegistrationController.php (active), .htaccess, HotelType.php, and login.html.twig. The main editor area displays the code for RegistrationController.php. The code includes namespace declarations, use statements for various Symfony and Doctrine classes/interfaces, and a class definition for RegistrationController that extends AbstractController. The register method is public and takes several parameters, including Request, UserPasswordHasherInterface, UserAuthenticatorInterface, GerantAuthenticator, and EntityManagerInterface. It creates a Gerant object, creates a form, and handles the request. A conditional block checks if the form is submitted and valid, then encodes the password and sets it on the user object. The bottom panel shows a terminal window with a list of commands and their execution times.

```

src > Controller > RegistrationController.php > ...
3 namespace App\Controller;
4
5 use App\Entity\Gerant;
6 //use App\Form\RegistrationFormType;
7 use App\Form\GerantType;
8 use App\Security\GerantAuthenticator;
9 use Doctrine\ORM\EntityManagerInterface;
10 use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
11 use Symfony\Component\HttpFoundation\Request;
12 use Symfony\Component\HttpFoundation\Response;
13 use Symfony\Component\PasswordHasher\Hasher\UserPasswordHasherInterface;
14 use Symfony\Component\Routing\Annotation\Route;
15 use Symfony\Component\Security\Http\Authentication\UserAuthenticatorInterface;
16 use Symfony\Contracts\Translation\TranslatorInterface;
17
18 class RegistrationController extends AbstractController
19 {
20     #[Route('/register', name: 'app_register')]
21     public function register(Request $request, UserPasswordHasherInterface $userPasswordHasher,
22         UserAuthenticatorInterface $userAuthenticator, GerantAuthenticator $authenticator,
23         EntityManagerInterface $entityManager): Response
24     {
25         $user = new Gerant();
26         $form = $this->createForm(GerantType::class, $user);
27         $form->handleRequest($request);
28
29         if ($form->isSubmitted() && $form->isValid()) {
30             // encode the plain password
31             $user->setPassword(
32                 $userPasswordHasher->hashPassword(

```

| PROBLEMS | OUTPUT | DEBUG CONSOLE | TERMINAL | COMMENTS |
|----------|--------|---------------|---------------------------------|----------|
| | 12 | 9.983 | git pull | |
| | 13 | 0.067 | git checkout -b Client/features | |
| | 19 | 0.791 | mkdir doc_pdf | |
| | 20 | 0.205 | mkdir doc_pdf | |