

# Compte rendu application Web GSB

## Sommaire

|                                                      |  |
|------------------------------------------------------|--|
| 1. Introduction.....                                 |  |
| 2. Préparation de l'environnement de travail.....    |  |
| 3. Mission 1 : Gérer les utilisateurs.....           |  |
| 4. Mission 2 : Page Ressources Humaines.....         |  |
| 5. Mission 3 : Validation sur les mots de passe..... |  |
| 6. Déploiement.....                                  |  |
| 7. Conclusion.....                                   |  |
| 8. Compétences validées.....                         |  |

## Introduction

Badénia Tech, jeune ESN, vient de remporter le marché de développement d'une application web pour l'entreprise pharmaceutique BSB (Galaxy Swiss Bourdin). L'application servira à améliorer et optimiser le travail de ses salariés. En tant que technicien développeur junior pour l'ESN, ma mission est de participer, avec 5 autres développeurs junior, au développement de l'application répondant au cahier des charges rédigé sous l'autorité de GSB. Ce Projet a été réalisé dans les locaux de Badenia Tech.

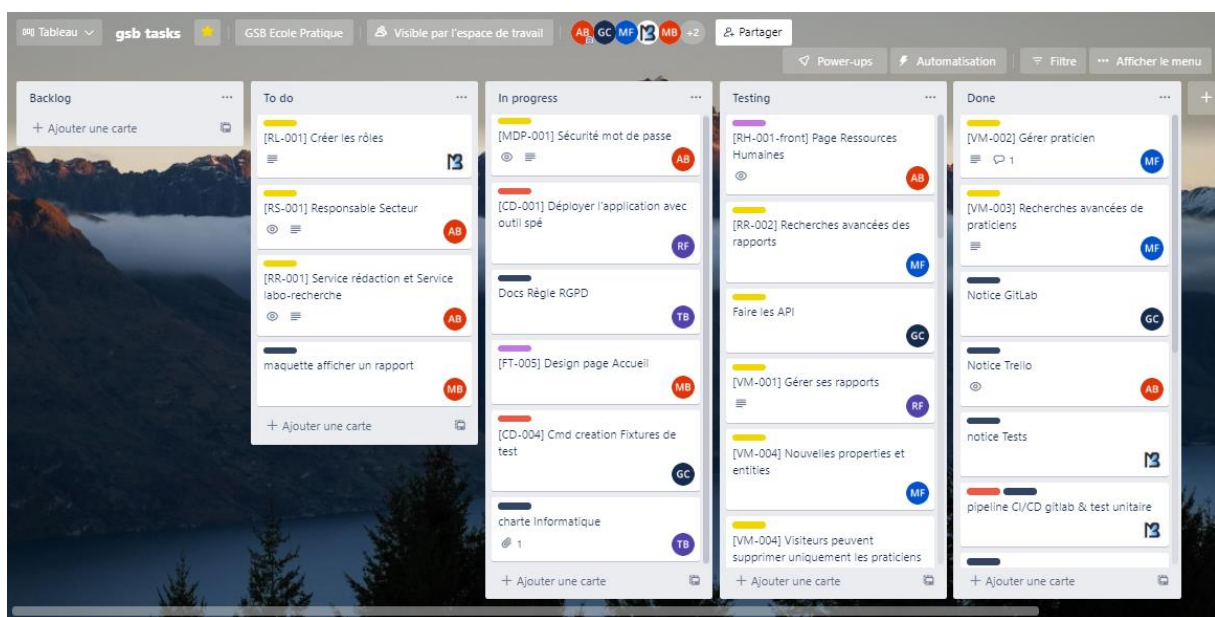
# Préparation de l'environnement de travail

## Gestion de projet

Avant de commencer la réalisation des besoins. Il est important de mettre en place la gestion de ces derniers. Celle-ci se fera grâce à l'outil Trello (voir la notice pour plus de détails).

Trello permettra de connaître clairement les besoins et de les suivre tout au long des réalisations. Toutes les tâches à effectuer sont listées dans le Backlog et ensuite attribuées.

Voici un aperçu du tableau :



## Communication

L'outil de communication que nous avons choisi est Discord. Vous pouvez retrouver tous les détails sur la mise en place du serveur dans la notice annexe.

## Gestion des versions

Tout le code de l'application est présent sur GitLab. Il n'est pas possible pour le public d'accéder au code. Tous les détails sur l'utilisation de Git et GitLab et sur la mise en œuvre de la gestion des versions sont disponibles dans la notice annexe.

## Création du projet

Pour réaliser cette application, nous avons choisi les outils suivants : La suite LAMP, le framework Symfony, Node.js, Yarn et Composer.

Voici la notice (README.md) disponible aux développeurs du projet pour qu'ils puissent facilement installer le projet.

### Getting Started

---

First, you need to deploy the environment on your local machine :

- Install WAMP
- Install Symfony
- Install Node.js
- Install Yarn
- Install Composer
- Run WAMP

Then :

1. Create a database in mysql with phpmyadmin
2. Create a `.env` using `.template.env` and update the `DATABASE_URL` with the mysql id and password, and the name of the database you created (e.g `symfony_pharma`)
3. To install the project dependencies : run `composer install`
4. Create tables in your database using doctrine with `php bin/console doctrine:schema:update --force`
5. Create fixtures for dev with `php bin/console doctrine:fixtures:load --group=dev`
6. Create fixtures for prod with `php bin/console doctrine:fixtures:load --group=prod`
7. Run `npm install`
8. Open a new terminal and run `yarn dev --watch`
9. Open a new terminal and run `symfony server:start`

That's it ! Now you are ready to use the GSB App on <http://localhost:8000>

L'application est désormais prête pour le développement des premières fonctionnalités. Vous trouverez, à la suite de ce compte rendu, les missions que j'ai réalisées.

## Mission 1 : Gestion des utilisateurs

Le compte administrateur doit pouvoir créer, modifier, et supprimer les utilisateurs suivants : Utilisateur, Visiteur médical et Responsable des Ressources Humaines.

Voici la tache sur Trello :



Premièrement, il est nécessaire de créer une entité Utilisateur. C'est une classe contenant toutes les propriétés et méthodes de l'Objet Utilisateur. Cette entité comme les autres seront ensuite persistées en base de données grâce à l'ORM Doctrine.

Doctrine s'installe simplement avec la commande :

- `composer require symfony/orm-pack`

Grâce à Symfony, il est possible de créer et persister des entités facilement avec les commandes suivantes :

- `php bin/console make:entity` (Pour créer l'entité)
- `php bin/console doctrine:schema:update --force` (Pour persister l'entité)
- `php bin/console make:crud` (Pour créer le contrôleur relié à l'entité)

Voice un aperçu du code une fois l'entité créée :

```
/**
 * @ORM\Entity(repositoryClass=UserRepository::class)
 * @UniqueEntity(fields={"email"}, message="There is already an account with this email")
 * @method string getUserIdentifier()
 * @Serializer\ExclusionPolicy("all")
 */
class User implements UserInterface
{
    /**
     * @ORM\Id
     * @ORM\GeneratedValue
     * @ORM\Column(type="integer")
     * @Serializer\Expose()
     */
    private $id;

    /**
     * @ORM\Column(type="string", length=180, unique=true)
     * @Serializer\Expose()
     */
    private $email;

    /**
     * @ORM\Column(type="json")
     * @Serializer\Expose()
     */
    private $roles = [];
```

Ici les propriétés avec des entêtes pour Doctrine. Elles serviront pour établir la persistance en bdd selon les paramètres choisis via le CLI. Il est aussi possible de relier les entités comme on peut le voir ci-dessous avec l'entité Praticiens reliée en OneToMany :

```

/**
 * @ORM\OneToMany(targetEntity=Praticien::class, mappedBy="creator")
 */
private $praticiens;

public function __construct()
{
    $this->clientMeetings = new ArrayCollection();
    $this->invitations = new ArrayCollection();
    $this->rapports = new ArrayCollection();
    $this->praticiens = new ArrayCollection();
}

public function __toString()
{
    return $this->email;
}

public function getId(): ?int
{
    return $this->id;
}

public function getEmail(): ?string
{
    return $this->email;
}

public function setEmail(string $email): self

```

Une fois l'entité Utilisateur créée, il faut créer le contrôleur pour effectuer les opérations Create Read Update Delete (CRUD) dessus.

Voici un aperçu du contrôleur :

```

/**
 * @Route("/new", name="rh_user_new", methods={"GET", "POST"})
 */
public function new(Request $request, UserPasswordEncoderInterface $passwordEncoder): Response
{
    $user = new User();
    $form = $this->createForm(UserType::class, $user);
    $form->handleRequest($request);

    if ($form->isSubmitted() && $form->isValid()) {
        $user
            ->setRoles([$request->request->get("roles")])
            ->setPassword($passwordEncoder->encodePassword(
                $user,
                $form->get('password')->getData()));

        $entityManager = $this->getDoctrine()->getManager();
        $entityManager->persist($user);
        $entityManager->flush();

        return $this->redirectToRoute('secured_home');
    }

    return $this->render('rh/user/new.html.twig', [
        'user' => $user,
        'form' => $form->createView(),
    ]);
}

```

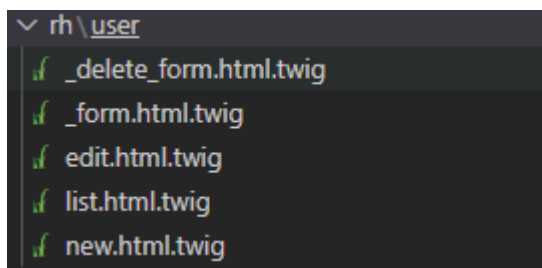


Le contrôleur va déterminer une route (en en-tête) qu'on va appeler par la suite pour créer un utilisateur, par exemple. Il va ensuite créer un objet Utilisateur avec les propriétés passées dans le formulaire du front, attribuer un rôle selon, et sécuriser le mot de passe grâce au Symfony Security Bundle (plus de détails dans la notice annexe). Enfin, il va persister l'entité en bdd et lui associer une vue.

La partie « back » est donc mise en place, il faut maintenant développer le « front ».

## Mission 2 : Page Ressources Humaines

Cette partie est constituée de pages html, créées avec le moteur de templates Twig. Twig permet de créer facilement des pages avec PHP. Par exemple pour les opérations effectuées par le Responsable des ressources humaines, il y a une page par opération, liées au contrôleur créé précédemment :



Ici les pages pour supprimer, éditer, lister, et créer un utilisateur, ainsi qu'un module pour le formulaire de création et d'édition.

Par exemple, pour créer un utilisateur, la page new.html.twig va inclure le module \_form.html.twig :

```
1  {% extends 'base.html.twig' %}
2
3  {% block title %}GSB | Nouvel Utilisateur{% endblock %}
4
5  {% block body %}
6      <div class="container">
7          <div class="card bg-white mt-5 card-container">
8              <h1 class="m-5 d-flex justify-content-center">Création d'un nouvel utilisateur</h1>
9              <div class="card-body">
10                 {{ include('rh/user/_form.html.twig') }}
11             </div>
12         </div>
13     </div>
14 {% endblock %}
15
```


```

1 <div class="container">
2   <form method="post">
3     {{ form_errors(form) }}
4     <div class="row">
5       <div class="col-8"><label class="mt-2" for="inputFirstname">Prénom</label>
6       <input type="text" name="user[firstname]" id="inputFirstname" class="form-control" placeholder="Prénom" required
7         autofocus>
8
9       <label class="mt-2" for="inputName">Nom</label>
10      <input type="text" name="user[name]" id="inputName" class="form-control" placeholder="Nom" required
11        autofocus>
12
13      <label class="mt-2" for="inputPhone">Numéro de téléphone</label>
14      <input type="text" name="user[phone]" id="inputPhone" class="form-control" placeholder="Numéro de téléphone"
15        required
16        autofocus>
17
18      <label class="mt-2" for="inputEmail">Email</label>
19      <input type="email" name="user[email]" id="inputEmail" class="form-control" placeholder="Email" required
20        autofocus>
21
22      <label class="mt-2" for="inputPassword">Mot de passe (8 caracteres)</label>
23      <input type="password" name="user[password]" id="inputPassword" class="form-control" placeholder="Mot de passe"
24        required>
25    </div>
26    <div class="col-4">
27      <label class="mt-2" for="roles">Poste</label>
28      <select name="roles" class="form-select" aria-label="role choice">
29        <option selected>Utilisateur</option>
30        <option value="ROLE_VISITEUR">Visiteur</option>
31        <option value="ROLE_RH">Ressources humaines</option>
32      </select>
33    </div>
34
35  </div>
36  {{ form_row(form._token) }}
37
38  <div class="d-flex justify-content-center mt-5">
39    <button type="submit" class="btn btn-primary m-2 ">Enregistrer</button>
40  </div>
41 </form>
42 </div>
43

```

Ce formulaire a été développé en suivant la maquette créée par un des membres de l'équipe (Mathieu BIER), que vous pouvez trouver en annexe.

On a donc un résultat qui est le suivant :


Gestion des utilisateurs ▾ Gestion des praticiens ▾ Gestion des rapports ▾
Déconnexion ↗

### Création d'un nouvel utilisateur

Prénom

Poste

Utilisateur ▾

Nom

Numéro de téléphone

Email

Mot de passe (8 caracteres)


Pour gérer le formulaire, du code est nécessaire, il va servir de pont entre la vue et le contrôleur. Voici un aperçu :

```
class UserType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $options)
    {
        $builder
            ->add('email', null, [
                'label' => 'E-mail'
            ])
            ->add('password', PasswordType::class, [
                'label' => 'Mot de passe',
                'mapped' => false,
                'constraints' => [
                    new NotBlank([
                        'message' => 'Please enter a password',
                    ]),
                    new Length([
                        'min' => 6,
                        'minMessage' => 'Your password should be at least {{ limit }} characters',
                        // max length allowed by Symfony for security reasons
                        'max' => 4096,
                    ]),
                ]
            ])
            ->add('name', null, [
                'label' => 'Nom de famille'
            ])
            ->add('firstname', null, [
                'label' => 'Prénom'
            ])
            ->add('phone', null, [
                'label' => 'Numéro de téléphone'
            ])
    ];
}
```















Dans le contrôleur :

```
/**
 * @Route("/new", name="rh_user_new", methods={"GET","POST"})
 */
public function new(Request $request, UserPasswordEncoderInterface $passwordEncoder): Response
{
    $user = new User();
    $form = $this->createForm(UserType::class, $user);
    $form->handleRequest($request);
}
```

Désormais il est possible de créer, éditer, supprimer et lire des utilisateurs dans l'application.

 [Gestion des utilisateurs](#) [Gestion des praticiens](#) [Gestion des rapports](#) [Déconnexion](#)

### Liste des utilisateurs

| Prénom    | Nom de famille | Email                   | Numéro de téléphone | Rôle                     | Action                                                                                                                                                                  |
|-----------|----------------|-------------------------|---------------------|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Roger     | Jaures         | roger.jaures@gsb.fr     | 0606060606          | USER                     |   |
| Charlotte | Becker         | charlotte.becker@gsb.fr | 0606060606          | RESSOURCES HUMAINES USER |   |
| Kevin     | Jean           | kevin.jean@gsb.fr       | 0707070707          | USER                     |   |
| Anna      | Julietta       | anna.julietta@gsb.fr    | 0101010101          | USER                     |   |
| Charles   | Collins        | charles.collins@gsb.fr  | 0101010101          | USER                     |   |
| Alice     | Paul           | alice.paul@app.fr       | 0007000700          | USER                     |   |
| Jack      | Dupont         | Jack.dupont@gmail.com   | 0791795212          | USER                     |   |

Afin d'assurer la sécurité des mots de passe et respecter le RGPD, il faut mettre en place la validation des entrées.

### Mission 3 : Validation sur les mots de passe

Voici les critères à respecter, selon la CNIL :

|                                            |                                                   |   |                                                                                                                                                                             |                                                                                                                                                                                                                                             |
|--------------------------------------------|---------------------------------------------------|---|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Avec restriction d'accès (le plus répandu) | SITES DE E-COMMERCE, COMPTE D'ENTREPRISE, WEBMAIL | 8 | <p>Au moins 3 des 4 types suivants :</p> <ul style="list-style-type: none"><li>• majuscules</li><li>• minuscules</li><li>• chiffres</li><li>• caractères spéciaux</li></ul> | <p>Blocage des tentatives multiples :</p> <p>(exemples)</p> <ul style="list-style-type: none"><li>• Temporisation d'accès au compte après plusieurs échecs</li><li>• « Captcha »</li><li>• Verrouillage du compte après 10 échecs</li></ul> |
|--------------------------------------------|---------------------------------------------------|---|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

La validation se fera directement dans le formulaire html :

```
<label class="mt-2" for="inputPassword">Mot de passe (8 caracteres minimum)</label>
<input type="password" name="user[password]" id="inputPassword" class="form-control" placeholder="Mot de passe"
pattern="(?!.*)(?!.*(\d))(?!.*[a-z])(?!.*[A-Z]).{8,}" title="Must contain at least one number and one uppercase and lowercase letter and one special character, and at least 8 or more charact
```

Le code CSS :

```
8  #message {
9    display:none;
10   background: #f1f1f1;
11   color: #000;
12   position: relative;
13   padding: 20px;
14   margin-top: 10px;
15 }
16
17 #message p {
18   padding: 10px 35px;
19   font-size: 18px;
20 }
21
22 .valid {
23   color: green;
24 }
25
26 .valid:before {
27   position: relative;
28   left: -35px;
29   content: "&#10004;";
30 }
31
32 /* Add a red text color and an "x" icon when the requirements are wrong */
33 .invalid {
34   color: red;
35 }
36
37 .invalid:before {
38   position: relative;
39   left: -35px;
40   content: "&#10006;";
41 }
```

Le code Javascript :

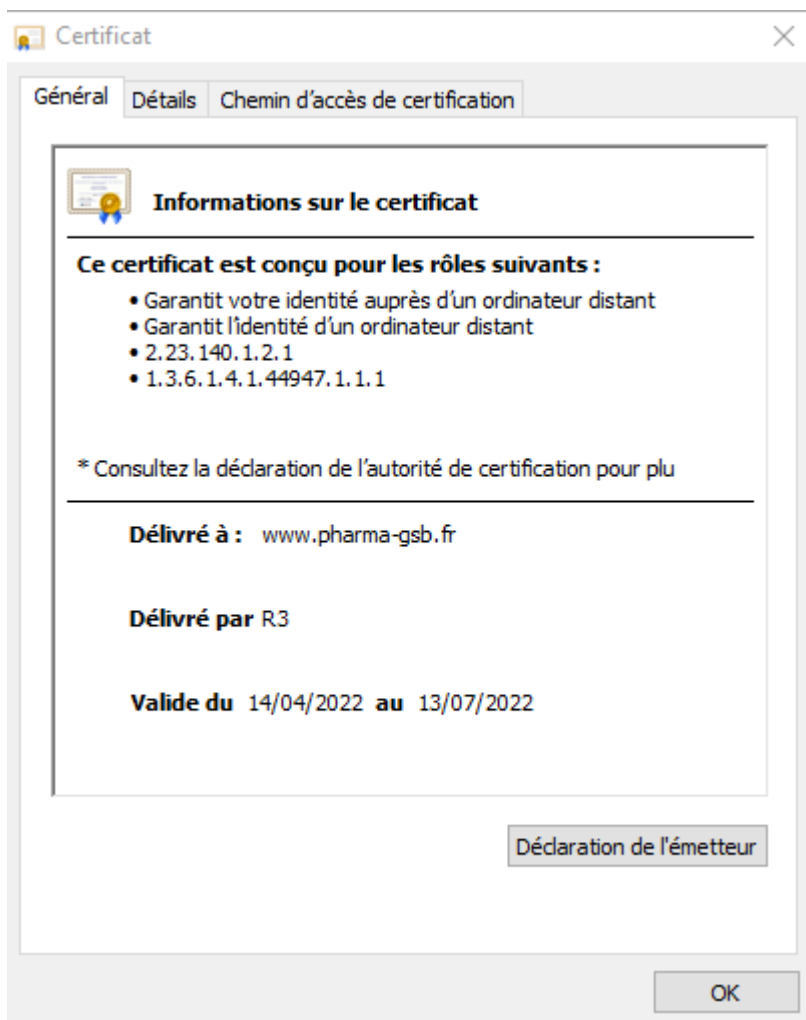
```
3  var myInput = document.getElementById("psw");
4  var letter = document.getElementById("letter");
5  var capital = document.getElementById("capital");
6  var number = document.getElementById("number");
7  var special = document.getElementById("special");
8  var length = document.getElementById("length");
9
10 // When the user clicks on the password field, show the message box
11 myInput.onfocus = function() {
12   document.getElementById("message").style.display = "block";
13 }
14
15 // When the user clicks outside of the password field, hide the message box
16 myInput.onblur = function() {
17   document.getElementById("message").style.display = "none";
18 }
19
20 // When the user starts to type something inside the password field
21 myInput.onkeyup = function() {
22   // Validate lowercase letters
23   var lowerCaseLetters = /[a-z]/g;
24   if(myInput.value.match(lowerCaseLetters)) {
25     letter.classList.remove("invalid");
26     letter.classList.add("valid");
27   } else {
28     letter.classList.remove("valid");
29     letter.classList.add("invalid");
30   }
31
32   // Validate capital letters
33   var upperCaseLetters = /[A-Z]/g;
34   if(myInput.value.match(upperCaseLetters)) {
35     capital.classList.remove("invalid");
36     capital.classList.add("valid");
37   } else {
38     capital.classList.remove("valid");
39     capital.classList.add("invalid");
40   }
41 }
```

On assure ainsi que le formulaire ne peut être validé sans respecter les critères de sécurité.

## Déploiement

L'application est disponible à l'adresse : <https://www.pharma-gsb.fr/>

L'application est sécurisée avec le protocole HTTPS, voici le certificat généré :



Tous les détails concernant le déploiement sont disponibles en annexe.

## Conclusion

La réalisation de ce projet a été mon premier sur une application web. Il m'a permis d'apprendre le travail en équipe, le développement web avec les frameworks Symfony et Bootstrap, et la programmation orienté objet. Lors de la première année, j'ai eu du mal à comprendre comment fonctionnait l'application, mais en apprenant la programmation orienté objet tout est plus clair. Le projet m'a aussi appris l'utilisation d'outils indispensables comme Git et GitLab, pour versionner son code et travailler en équipe, et plus encore... Même si je n'ai pas énormément travaillé avec Symfony, mais plus avec des framework Javascript et Typescript, je pense désormais être capable de développer ce genre d'application, moi-même ou avec une équipe.

## Compétences validées

| Répondre aux incidents et aux demandes d'assistance et d'évolution                                                                                                                                                                                | Développer la présence en ligne de l'organisation                                                                                                                                                                                                                                                                                                                                           | Travailler en mode projet                                                                                                                                                                                                                | Mettre à disposition des utilisateurs un service informatique                                                                                                                                                                |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> <li>▸ Collecter, suivre et orienter des demandes</li> <li>▸ Traiter des demandes concernant les services réseau et système, applicatifs</li> <li>▸ Traiter des demandes concernant les applications</li> </ul> | <ul style="list-style-type: none"> <li>▸ Participer à la valorisation de l'image de l'organisation sur les médias numériques en tenant compte du cadre juridique et des enjeux économiques</li> <li>▸ Référencer les services en ligne de l'organisation et mesurer leur visibilité.</li> <li>▸ Participer à l'évolution d'un site Web exploitant les données de l'organisation.</li> </ul> | <ul style="list-style-type: none"> <li>▸ Analyser les objectifs et les modalités d'organisation d'un projet</li> <li>▸ Planifier les activités</li> <li>▸ Évaluer les indicateurs de suivi d'un projet et analyser les écarts</li> </ul> | <ul style="list-style-type: none"> <li>▸ Réaliser les tests d'intégration et d'acceptation d'un service</li> <li>▸ Déployer un service</li> <li>▸ Accompagner les utilisateurs dans la mise en place d'un service</li> </ul> |
| X                                                                                                                                                                                                                                                 | X                                                                                                                                                                                                                                                                                                                                                                                           | X                                                                                                                                                                                                                                        | X                                                                                                                                                                                                                            |