

Hypnos



Dossier Projet

Développeur Web et Web Mobile

Hugo Dessauw

Studi
DIGITAL EDUCATION



Présentation du projet

- Résumé du projet
- Compétences du référentiel couvertes par le DP
- Environnement
- Mon Projet
 - Cahier des charges
 - Spécifications techniques
- Réalisation du projet
 - Description d'une situation de travail
- Recherche Anglophone
- Conclusion



Le projet Hypnos

- Issu d'un ancien ECF (Evaluation en Cours de Formation)
- Exercice proposé par Studi+
- Délai: Rush de 2 Semaines, ajout de fonctionnalités s'en est suivie

Objectif :

Concevoir une application d'Hôtellerie pour une entreprise fictive.



Compétences

Front-End

- ▀ Maquettage de l'application
- ▀ Interface utilisateur web Responsive
- ▀ Dynamisme des pages



Compétences

Back-End

- Création base de données
- Développer les composants d'accès au données
- Développer la partie Back-End d'une application



Environnement

- ▀ Humain
- ▀ Technique



Mon projet

- Cahier des charges
 - Général
 - Administrateur
 - Manager
 - Utilisateur
- Spécifications techniques



Mon projet

Cahier des charges

- Général
 - Libre d'accès
 - Formulaire de contact à disposition

Plus d'exemple p.32



Mon projet

Cahier des charges

- Administrateur
 - Hôtel : Créer, modifier, supprimer
 - Manager : Créer, modifier, supprimer
 - Recevoir et traiter formulaires de contact

Mon projet

Cahier des charges

- Manager
 - Suite : Créer, modifier, supprimer
 - Galerie d'image : Créer, modifier, supprimer
 - Un manager par hôtel

Plus d'exemple p.27



Mon projet

Cahier des charges


- Utilisateur
 - S'enregistrer
 - Se connecter
 - Réserver
 - Parcourir ses réservations
 - Voir celles annulées

Mon projet

Spécifications Techniques

- Environnement de travail :
 - Editeur de texte:
 - Système d'exploitation:
 - Dépôt distant :
 - Gestion des dépendances:

 PHPStorm

 Windows

 Github

 NPM

Mon projet

Spécifications Techniques

➤ Front-End:

- Moteur de template:
- Développement d'interfaces:
- Langages prédominants:
- Framework utilisé:

 TWIG

 WebpackEncore

 HTML, SCSS, JS

 Bootstrap

Mon projet

Spécifications Techniques

► Back-End:

► Environnement côté serveur:

 Node.js

► Serveur local:

 MariaDB


► Serveur distant:

 MySQL

► Langage prédominant:

 PHP

► Framework utilisé:

 Symfony

► Utilisations de multiples bundles Symfony

Réalisation du projet

- 1) Création de l'application
 - Conception BDD
 - Analyse chez la concurrence
 - Maquettage
 - Sélection Design System

Réalisation du projet

Création BDD

- Conception de la base de données
 - Analyse du sujet
 - Todo List
 - Cerner les besoins de l'application

Méthode Utilisée :

Agile

Réalisation du projet

Création BDD

- Conception de la base de données
 - Création de personas
 - Personne fictive
 - Représente groupe de personne ciblé pour les besoins de l'application
 - Y sont répertorié : Profil, aisance numérique, Utilisation de notre produit ...

Réalisation du projet

Création BDD



- Nom Robert
- Prénom: Lalarve
- Âge: 23 ans
- Situation sociale: Etudiant
- Adresse: Habite à Saint-Jean Plât de Cor
- Culture digitale:
Expérimenté

Personnalité:

Timide; Brave; Dévoué;

Utilisation du produit:

Désire s'enregistrer sur le site
Désire avoir accès aux hôtels disponibles.

Profil:

Bob, vivant encore chez ses parents, souhaite amener son amour de jeunesse vivre un séjour en amoureux. Sans trop de finances, car étudiant, mais malgré tout décidé de partir en vacances, il souhaite s'enregistrer sur le site afin de se renseigner sur les emplacements des hotels.

Problème/Préférence:

Ne pas avoir accès aux hotels disponibles
Ne pas pouvoir réserver.

Réalisation du projet

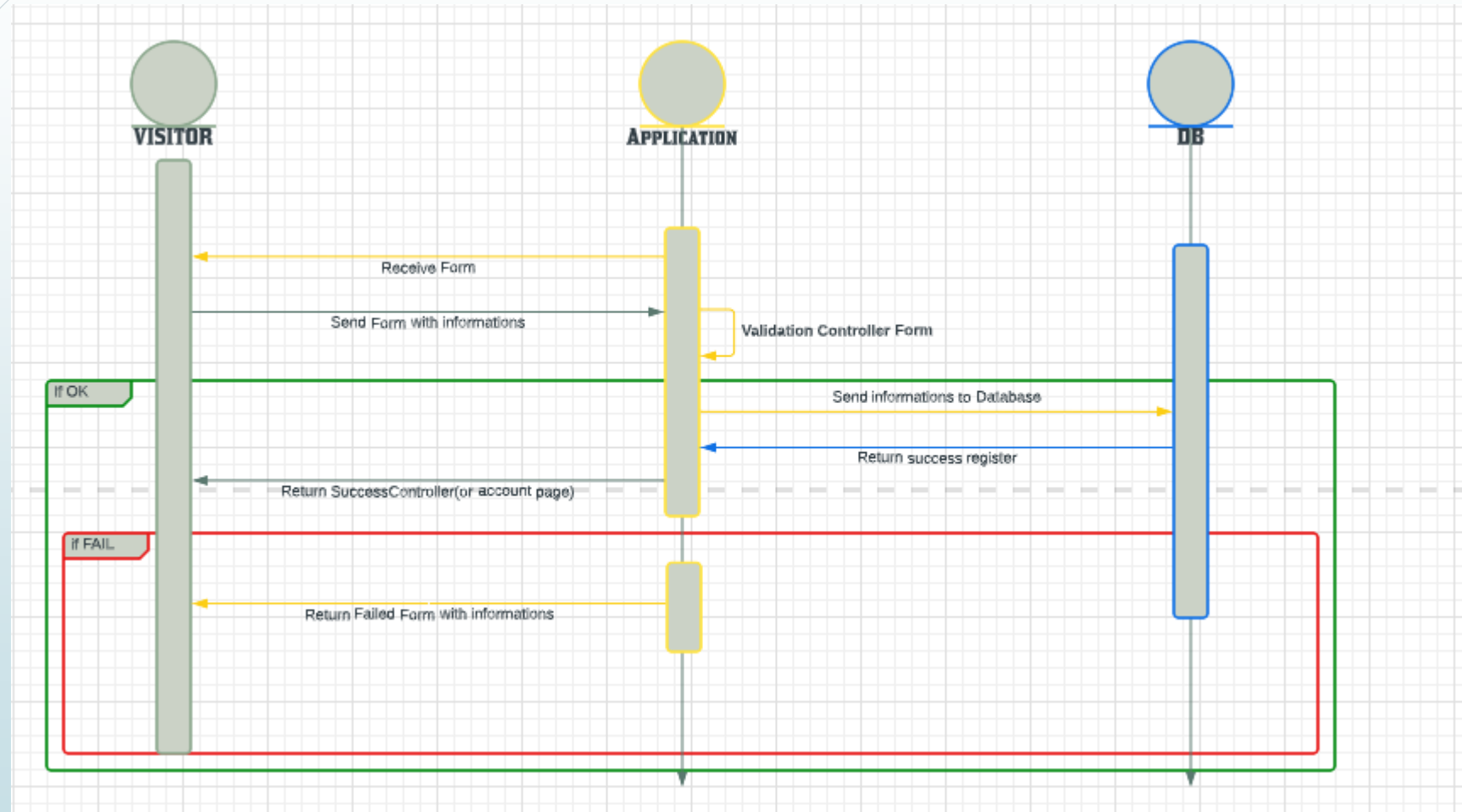
Création BDD

► User Stories :

- **En tant que** Robert,
- **Je souhaite** pouvoir m'enregistrer sur le site,
- **Afin** de pouvoir parcourir la liste des Hôtels

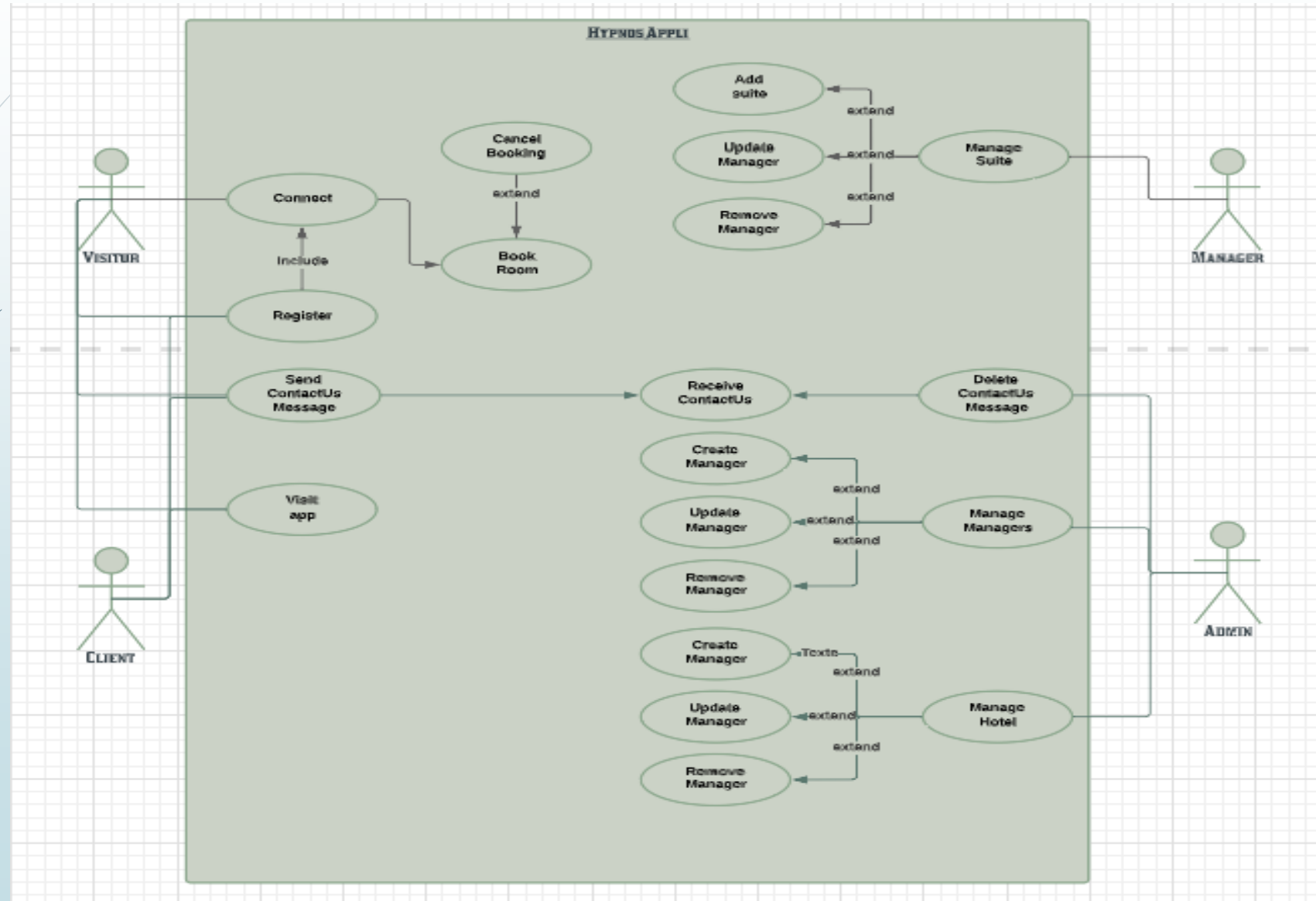
Réalisation du projet

Création BDD



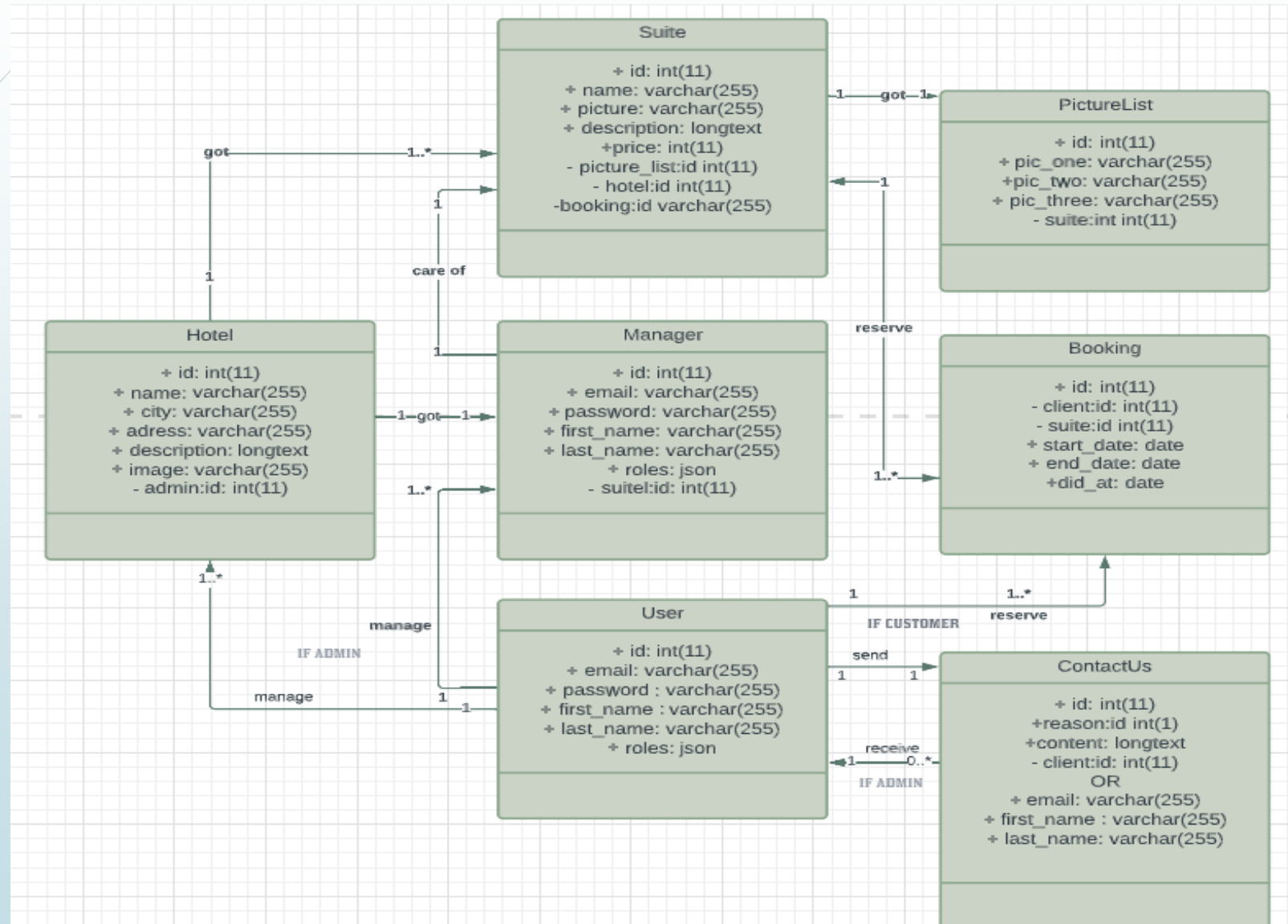
Réalisation du projet

Création BDD



Réalisation du projet

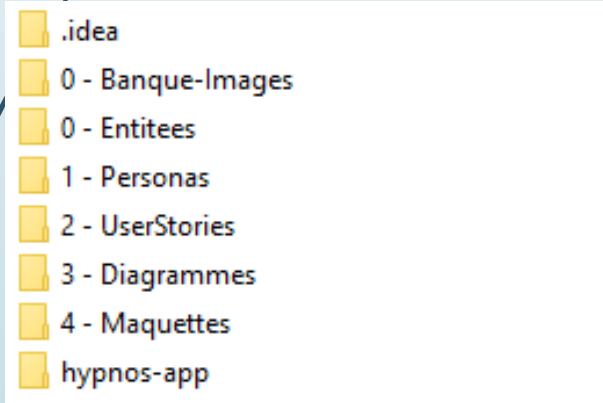
Création BDD



Réalisation du projet

Création BDD

➤ Création de mon application:



- .idea
- 0 - Banque-Images
- 0 - Entitees
- 1 - Personas
- 2 - UserStories
- 3 - Diagrammes
- 4 - Maquettes
- hypnos-app

</>

```
symfony new hypnos-app --version='6.1.*'
```

```
composer require symfony/webpack-encore-bundle
```

```
npm install sass-loader@^9.0.1 node-sass --save-dev
```

```
npm install bootstrap --save-dev
```

```
composer require symfony/orm-pack
```

</>

Première installation de bundles

Réalisation du projet

Création BDD

► Création de ma BDD:

```
'DATABASE_URL' => 'mysql://root:@127.0.0.1:3306/hypnos-app-db?serverVersion=mariadb-10.4.118&charset=utf8mb4'
```

```
php bin/console doctrine:database:create
```

```
symfony console make:entity entity_name
```


Réalisation du projet

Création BDD

➤ Exemple de création Entité

Création des Entités:

- Hotel
- Suite
- User
- Manager

```
CREATE TABLE hotel
(
  id int NOT NULL PRIMARY KEY AUTO_INCREMENT,
  name varchar(250) NOT NULL,
  ville varchar(250) NOT NULL,
  codePostal INT(5) NOT NULL
)
INSERT INTO hotel ( name, ville, codePostal)
VALUES ("Eurotel", "Perols", "34470")
```

```
PS C:\Travail\Sites\blog-app> symfony console make:entity Hotel

created: src/Entity/Hotel.php
created: src/Repository/HotelRepository.php

Entity generated! Now let's add some fields!
You can always add more fields later manually or by re-running this command.

New property name (press <return> to stop adding fields):
> name

Field type (enter ? to see all types) [string]:
> string

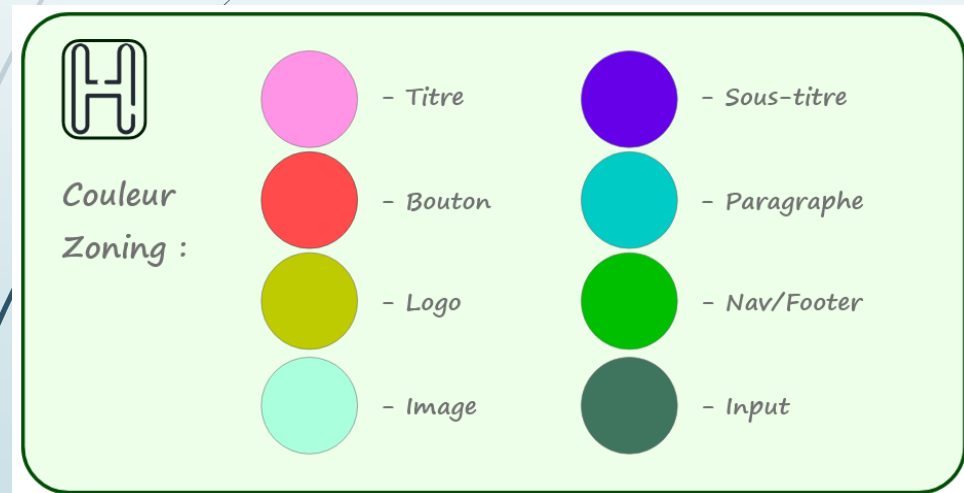
Field length [255]:
>

Can this field be null in the database (nullable) (yes/no) [no]:
>
```

Réalisation du projet

Maquettage

➡ Création du Zoning :

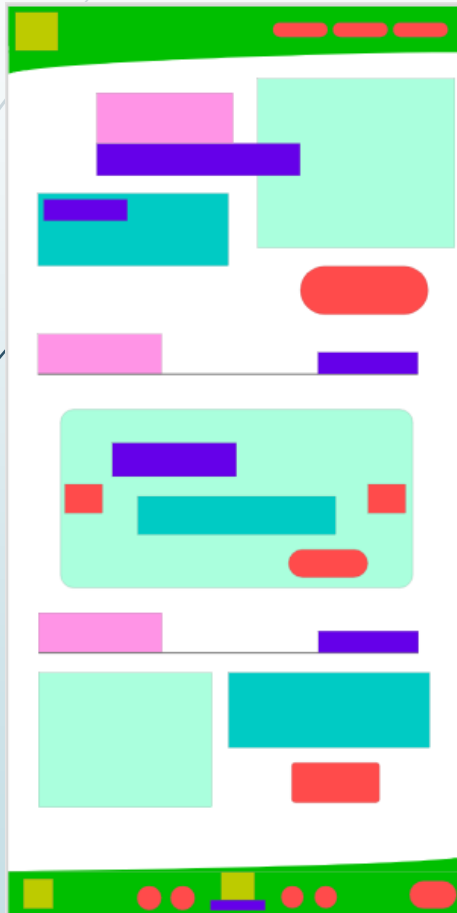


- Outil de maquettage : Adobe XD
- Format mobile, puis Desktop
- Pages principales

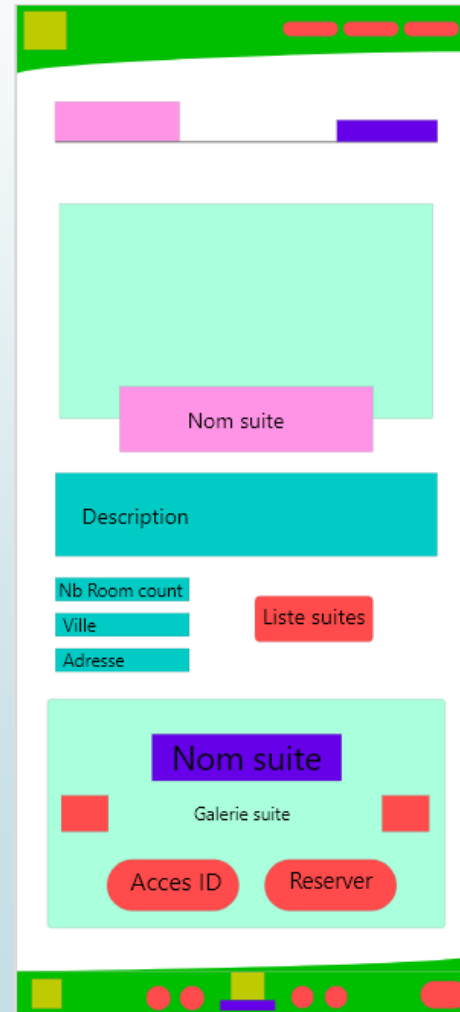
Réalisation du projet

Maquettage

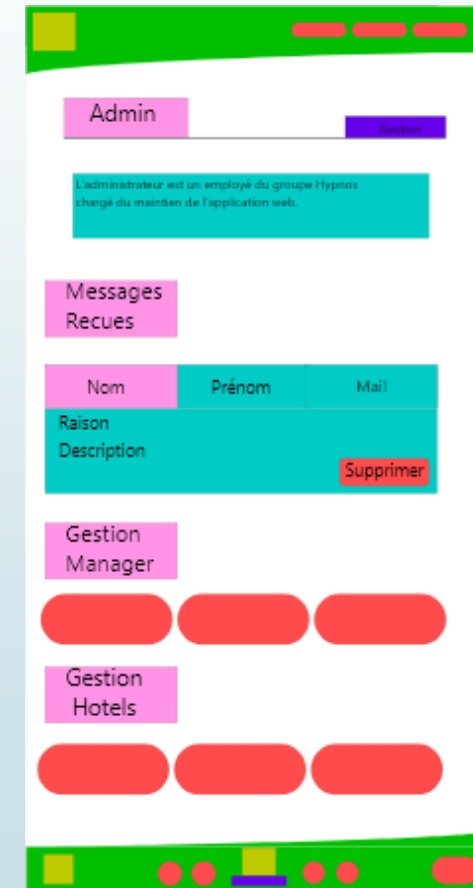
Accueil



Etablissement/id



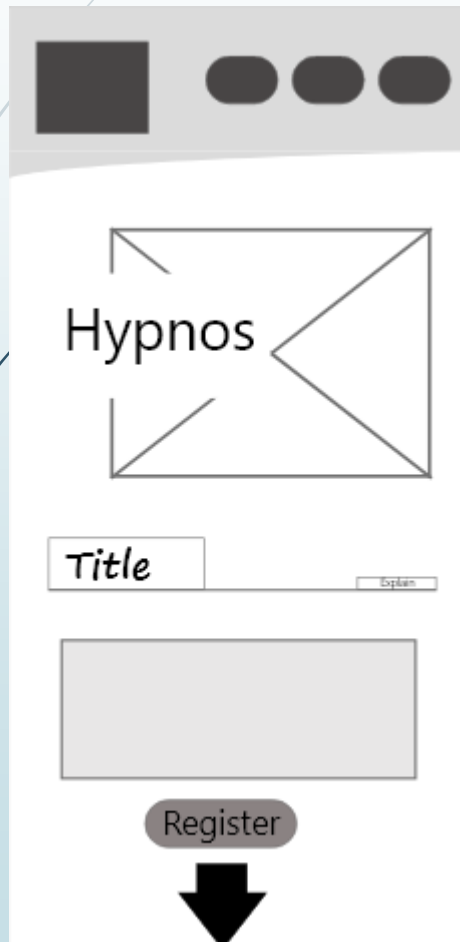
Gestion



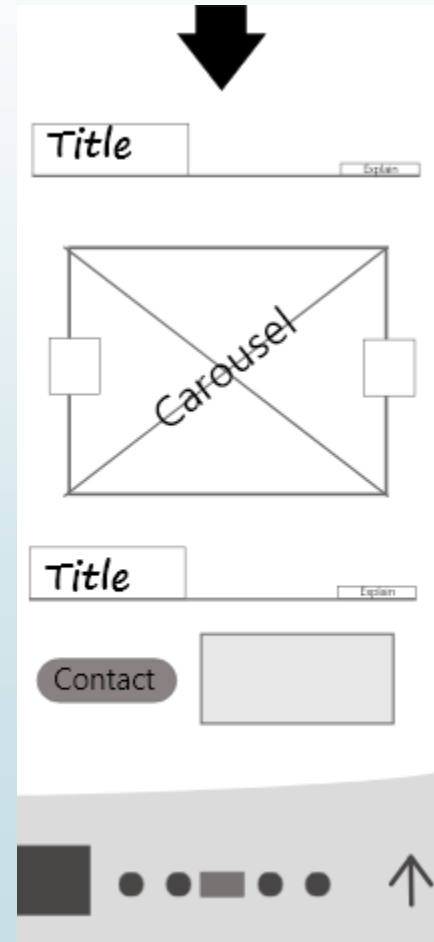
Réalisation du projet

Maquettage

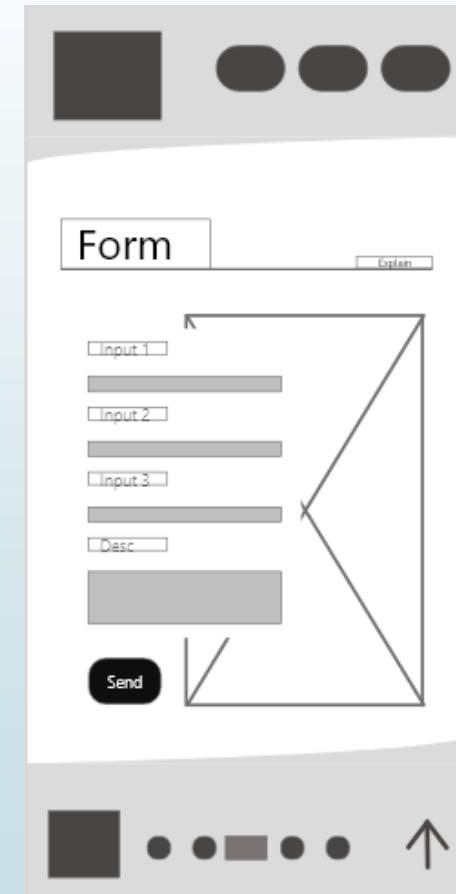
Accueil



Acceuil suite



Formulaire



Réalisation du projet

Design System

► Sélection du Design System :

Font-Family :

- Titre 'Bai Jamjuree', sf
- Sous
titre Consolas, serif
- Button Consolas, sans-serif
- Paragraphe Oswald, sans-serif;

Couleur :

DFFFFE

eee6d6

39EFEF

c8b48f

684034

157061

625f48

Réalisation du projet

Design System

- Implantation dans mon projet :
 - _design.scss

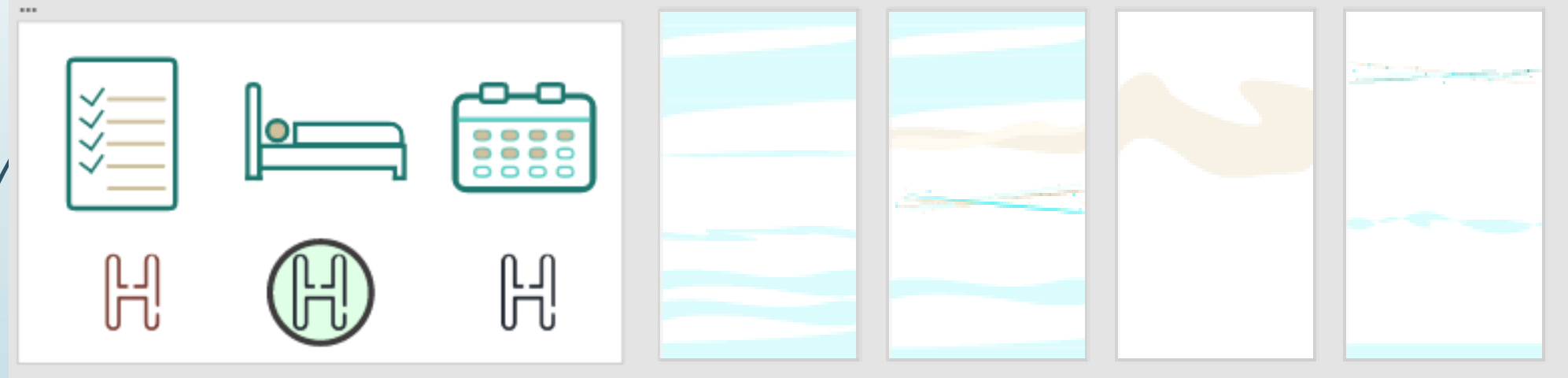
```
@import '../..//node_modules/bootstrap/scss/bootstrap';

//Palette couleur
// Marron
$marronLight: #eee6d6;
$marronMedium: #c8b48f;
$marronDark: #625f48;
$marronLogo: #684034;
// Bleu
$blueNav: #DFFFFE;
$blueButton: #39EFEF;
$blueBorder: #157061;
//Font-family:
//Title
$title: 'Bai Jamjuree', sans-serif;
//Subtitle
$subTitle: Consolas, serif;
//Buttons
$button: Consolas, sans-serif;
//Text1
$textOne: Oswald, sans-serif;
```

Réalisation du projet

Background/Logo

- Modification couleur logo, création Backgrounds:



Réalisation du projet

Faire un point

► Récapitulatif:

- Je possède toutes les fonctionnalités qui me sont demandées
- Parcours utilisateur fluide et intuitif
- Logique de mon application
- Mes entités



Réalisation du projet

- 2) Conception de l'application
 - Mise en place Router/Contrôleurs
 - Création Register/Login
 - Gestion de compte
 - Dynamisme des pages
 - Sécurité application
 - Déploiement du projet

Réalisation du projet

Conception

- Création des pages principales
 - Accueil
 - Liste des hotels et des suites
 - Page de gestion utilisateur
 - Page par ID Hotel et suite

</>

```
symfony console make:controller SuiteController
```

</>

Réalisation du projet

Routing

- Exemple du lien d'une Suite:

Lien TWIG

```
<h4><a href="{{ path('app_etablissement', { 'id' : hotels.id }) }}">
    •{{ hotels.name }}
</a>
</h4>
```

Suite en fonction de l'ID

```
namespace App\Controller\ById;

use ...

class SuiteController extends AbstractController
{
    #[Route('/suite/{id}', name: 'app_suite')]
    public function index(ManagerRegistry $doctrine, int $id): Response
    {
        $suite = $doctrine->getRepository( Suite::class)->find($id);
        $pictureList = $doctrine->getRepository( PictureList::class)->findAll();

        return $this->render( view: 'by_id/suite.html.twig', [
            'title' => 'Hypnos - Suite',
            'suite' => $suite,
            'pictureList' => $pictureList,
        ]);
    }
}
```

Réalisation du projet

Suite par ID TWIG

```
<!-- Suite ID -->
<div class="backHotelId">
  <div class="roomId">
    
    <div class="suiteIdDesc">
      <h4>{{ suite.hotel }}</h4>
      <div class="suiteIdDescListButton">
        <div class="suiteIdDescList">
          <h5>Nom de la suite: </h5>
          <p> {{ suite.name }}</p>
          <h5>Propriétaire:</h5>
          <p>{{ suite.manager }}</p>
          <h5>Prix à la nuit :</h5>
          <p>{{ suite.price }}€/Nuit</p>
          <h5>Description: </h5>
          <p>{{ suite.description }}</p>
        </div>
        <div class="suiteIdDescButton">
          {% if app.user %}
            {% if is_granted('ROLE_ADMIN') %}
              <button type="button" class="buttonAcceuil"><a href="{{ path('app_update_suite', { 'id' : suite.id}) }}">
            {% elseif is_granted('ROLE_MANAGER') %}
              <button type="button" class="buttonAcceuil"><a href="{{ path('app_update_suite', { 'id' : suite.id}) }}">
            {% elseif is_granted('ROLE_USER') %}
              <button type="button" class="buttonAcceuil"><a href="{{ path('app_booking') }}">Réserver</a></button>
            {% endif %}
          {% else %}
            <button type="button" class="buttonAcceuil"><a href="{{ path('app_redirect_booking') }}">Réserver</a></button>
          {% endif %}
          <button type="button" class="buttonAcceuil"><a href="{{ path('app_contact_us') }}">En savoir plus</a></button>
        </div>
      </div>
    </div>
  </div>
</div>
```

Réalisation du projet

Register

- Création du Formulaire Register :
 - Configuration du hachage de mot de passe dans security.yaml
 - Rôle par défaut : User
 - Vérification de l'email

</>

```
symfony console make:registration-form
```

</>

Bundle Form, Security, Verify Email

Réalisation du projet

Login

- Création du Formulaire Login :
 - Protection csrf
 - Configuration du path dans security.yaml

Formulaire de connexion

```
<form action="{{ path('app_login') }}" method="post" class="loginForm">
  <div class="form-group email">
    <label for="username">Adresse mail:</label>
    <input type="email" class="form-control" id="username" name="_username" value="{{ last_username }}" aria-describedby="emailHelp" placeholder="Entrez votre adresse e-mail" />
    <small id="emailHelp" class="form-text text-muted">Nous ne partagerons votre adresse e-mail avec personne.</small>
  </div>
  <div class="form-group password">
    <label for="password">Mot de passe: </label>
    <input type="password" class="form-control" id="password" name="_password" placeholder="Entrez votre mot de passe" />
    <input type="hidden" name="_csrf_token" value="{{ csrf_token('authenticate') }}" />
  </div>
  <button type="submit" class="buttonSendForm">Connexion</button>
  <br>
  {% if error %}
    <div class="errorLogin">{{ error.messageKey|trans(error.messageData, 'security') }}</div>
  {% endif %}
</form>
```

Réalisation du projet

Login

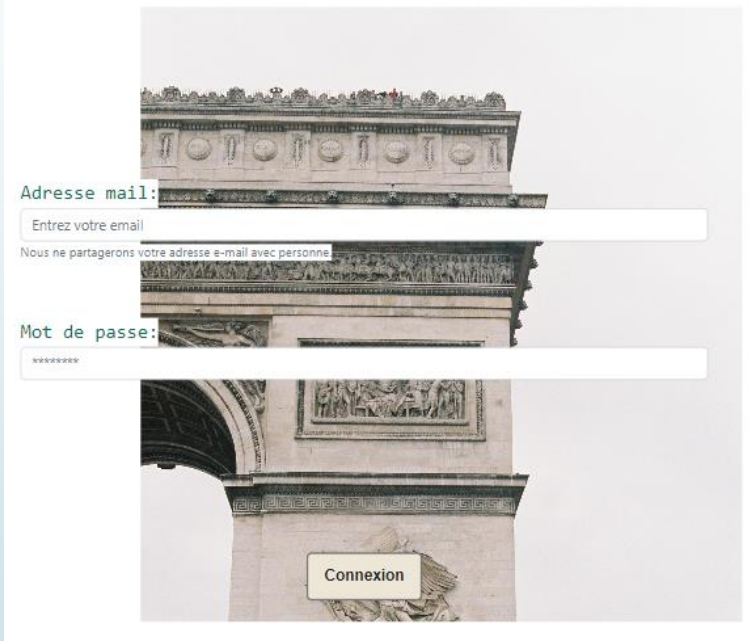
Login Controller

```
class LoginController extends AbstractController
{
    #[Route('/hypnos-connexion', name: 'app_login')]
    public function index(AuthenticationUtils $authenticationUtils): Response
    {
        $error = $authenticationUtils->getLastAuthenticationError();
        $lastUsername = $authenticationUtils->getLastUsername();

        return $this->render(view: 'form/login.html.twig', [
            'title' => 'Hypnos - Connexion',
            'last_username' => $lastUsername,
            'error' => $error,
        ]);
    }

    #[Route('/logout', name: 'app_logout', methods: ['GET'])]
    public function logout()
    {
        // Redirection dans security.yaml
    }
}
```

Connexion



</>

symfony console make:controller Login

</>

Réalisation du projet

Login

security.yaml

```
security:
  password_hashers:
    Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface: 'auto'
  providers:
    chain_provider:
      chain:
        providers: [ app_user_provider, app_manager_provider ]

    app_user_provider:
      entity:
        class: App\Entity\User
        property: email
    app_manager_provider:
      entity:
        class: App\Entity\Manager
        property: email
  firewalls:
    dev:
      pattern: ^/(_(profiler|wdt)|css|images|js)/
      security: false
    main:
      lazy: true
      provider: chain_provider
      form_login:
        login_path: app_login
        check_path: app_login
        enable_csrf: true
      logout:
        path: app_logout
        target: app_home
```


Réalisation du projet

Gestion

➡ Page de gestion Administrateur:

```
➡ class AdminGestionController extends AbstractController
{
    #[Route('/admin/{id}', name: 'app_admin_gestion')]
    public function index(ManagerRegistry $doctrine,
        int $id): Response
    {
        $admin = $doctrine->getRepository( persistentObject: User::class)->find($id);
        $hotellist = $doctrine->getRepository( persistentObject: Hotel::class)->findAll();
        $manager = $doctrine->getRepository( persistentObject: Manager::class)->findAll();
        $contact = $doctrine->getRepository( persistentObject: ContactUs::class)->getContactUsList();

        return $this->render( view: 'gestion/admin.html.twig', [
            'title' => 'Hypnos - Gestion Administrateur',
            'admin' => $admin,
            'hotellist' => $hotellist,
            'contact' => $contact,
            'manager' => $manager,
        ]);
    }
}
```

Réalisation du projet

Administrateur

Hotel/Manager

Organisation

Email	Prénom Manager	Hotel attribué
Manager1-Hypnos@gmail.com	Manager	Hotel Bon'Apparts
Manager2-Hypnos@gmail.com	Manager	Hotel Maritime
Manager3-Hypnos@gmail.com	Manager	Hotel Le Petit Moulin
Manager4-Hypnos@gmail.com	Manager	Hotel Rome Antique
Manager5-Hypnos@gmail.com	Manager	Hotel De France
Manager6-Hypnos@gmail.com	Manager	Hotel Atlas
Manager7-Hypnos@gmail.com	Manager	Hotel Tong Serviette

Gestion Managers

Il vous sera permis, dans cette section, de gérer les différents managers faisant parti de votre entreprise. Vous pourrez en assigner un au bâtiment de votre choix, pour qu'il puisse à son tour créer et administrer les différentes suites que compose l'hôtel.

Création Manager

Manager1-Hypnos@gmail.com	Mettre à jour	Supprimer
Manager2-Hypnos@gmail.com	Mettre à jour	Supprimer
Manager3-Hypnos@gmail.com	Mettre à jour	Supprimer
Manager4-Hypnos@gmail.com	Mettre à jour	Supprimer
Manager5-Hypnos@gmail.com	Mettre à jour	Supprimer
Manager6-Hypnos@gmail.com	Mettre à jour	Supprimer
Manager7-Hypnos@gmail.com	Mettre à jour	Supprimer

Formulaires reçus

Contact

De : Contact1@contact.com

Nom

Jean

Prénom

Dubouche

Raison

Je souhaite poser une réclamation

Description

Cotton candy marshmallow powder icing gingerbread powder chocolate bar cotton candy, jelly beans ice cream candy canes candy canes pudding cheesecake danish cake oatmeal wafer, Cookie toffee ice cream apple pie ice cream pie liquorice torte tart.

Supprimer

De : Contact2@contact.com

Nom

Charles

Prénom

DeGaulle

Raison

Je souhaite commander un service supplémentaire

Description

Chocolate bar jelly beans carrot cake gummiex brownie cotton candy muffin, Brownie sugar plum macarons patry shortbread, Pudding soufflé sesame wafer soufflé marzipan cake marshmallow halvah danish, Fruitcake lollipop jelly-o muffin apple pie.

Supprimer

Gestion Hotel

Il vous sera permis, dans cette section, de créer les différents hotels que compose votre entreprise. Vous pourrez modifier leurs informations. En cas de besoin, puis un manager (précédemment créé) pourra y insérer les différentes suites constituant l'hôtel en question.

Création Hotel

Hotel Bon'Apparts	Mettre à jour	Supprimer
Hotel Maritime	Mettre à jour	Supprimer
Hotel Le Petit Moulin	Mettre à jour	Supprimer
Hotel Rome Antique	Mettre à jour	Supprimer
Hotel De France	Mettre à jour	Supprimer
Hotel Atlas	Mettre à jour	Supprimer
Hotel Tong serviette	Mettre à jour	Supprimer

Réalisation du projet

Exemple : Créer un Manager

```
class ManagerController extends AbstractController
{
    #[Route('/add-manager', name: 'app_manager_create')]
    public function index(Request $request,
        EntityManagerInterface $entityManager,
        UserPasswordHasherInterface $managerPasswordHasher):
        Response
    {
        $manager = new Manager();

        $form = $this->createForm( type: ManagerType::class, $manager);
        $form->handleRequest($request);
        if ($form->isSubmitted() && $form->isValid()) {
            // Hashage mdp
            $manager->setPassword(
                $managerPasswordHasher->hashPassword(
                    $manager,
                    $form->get('password')->getData()
                )
            );
            $manager->setRoles(array('ROLE_MANAGER'));
            $manager = $form->getData();
            //Push
            $entityManager->persist($manager);
            $entityManager->flush();
            return $this->redirectToRoute( route: 'app_success_add_manager');
        }
        return $this->render( view: 'form/manager-creation.html.twig', [
            'title' => 'Hypnos - Creation Manager',
            'form' => $form->createView(),
        ]);
    }
}
```

```
$builder
->add( child: 'email', type: EmailType::class, [
    'label' => 'Email du Manager : ',
    'attr' => array(
        'placeholder' => 'ManagerX-Hypnos@gmail.com')
])
->add( child: 'password', type: PasswordType::class, [
    'label' => 'Mot de passe provisoire',
    'mapped' => false,
    'attr' => [
        'autocomplete' => 'new-password',
        'placeholder' => '*****'
    ],
    'constraints' => [
        new NotBlank([
            'message' => 'Vous devez entrer un mot de passe',
        ]),
        new Length([
            'min' => 6,
            'minMessage' => 'Votre mot de passe doit contenir au moins {{ lim',
            'max' => 4096,
        ]),
    ],
])
->add( child: 'first_name', type: TextType::class, [
    'label' => 'Votre nom : ',
    'attr' => array(
        'placeholder' => 'Dubois')
])
->add( child: 'last_name', type: TextType::class, [
    'label' => 'Votre prénom : ',
    'attr' => array(
        'placeholder' => 'Robert')
])
->add( child: 'submit', type: SubmitType::class, [
```

Réalisation du projet

Gestion

➡ Page de gestion Manager:

```
➡ class ManagerGestionController extends AbstractController
{
    #[Route('/manager/{id}', name: 'app_manager_gestion')]
    public function index(ManagerRegistry $doctrine,
        int $id): Response
    {
        $manager = $doctrine->getRepository( persistentObject: Manager::class)->find($id);
        $hotel = $doctrine->getRepository( persistentObject: Hotel::class)->findOneBy(['manager' => $manager]);

        $suite = $doctrine->getRepository( persistentObject: Suite::class)->findAll();
        $picturelist = $doctrine->getRepository( persistentObject: PictureList::class)->findAll();

        $suiteById = $doctrine->getRepository( persistentObject: Suite::class)->findOneBy(['manager' => $manager]);
        $picturelistById = $doctrine->getRepository( persistentObject: PictureList::class)->findBy(['suite' => $suiteById]);

        return $this->render( view: 'gestion/manager.html.twig', [
            'title' => 'Hypnos - Gestion Manager',
            'manager' => $manager,
            'hotel' => $hotel,
            'suite' => $suite,
            'picturelist' => $picturelist,
            'picturelistById' => $picturelistById,
        ]);
    }
}
```

Réalisation du projet

Manager

Ajouter des photos

Suite



Vous pourrez ci-dessous ajouter une galerie d'image accompagnant la Suite précédemment créée.

Creation Galerie d'image

Charlemagne II



Mettre à jour

Supprimer

Vos suite

Hotel



Vous trouverez ci-dessous la liste des suites qui sont assignées à Hotel Bon'Apparts. Vous pourrez en créer des nouvelles, ou en actualiser

Ajout d'une suite

Liste des suites



Mettre à jour

En savoir plus

Supprimer

Charlemagne II

Description :

Restaurant proposé par l'hotel, Chambre avec vue particulière possible, Spa / Hammam / Massage à

Prix :

345 €/Nuit

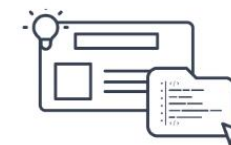
Propriétaire :

Manager Un

Votre hotel

Attribution

Vous avez ici une vue d'ensemble sur votre Hotel. Il ne vous sera pas permis de modifier ses informations, mais il sera par contre autorisé la création de suites le composant. Le panorama de gestion de suite est plus bas.



Hotel Bon'Apparts



Description:

Qui n'a jamais rêvé de loger dans un hotel présidentiel? Allez, avouez le ! Vous aussi, vous aimeriez découvrir la vie de château. Rassurez-vous, Napoléon n'a pas l'apanage de la noblesse. Pour un t...

Adresse :

123 Avenue du Putsch

Ville :

Paris

Nombre de suites :

3

Accès Hotel

Réalisation du projet

Exemple : Changement MDP

- Votre prénom : Manager
- Votre nom : Un
- Votre mail : Manager1-Hypnos@gmail.com

Il est responsable de la gestion de l'exploitation de l'établissement et des résultats. Pour développer le chiffre d'affaires de son établissement, il veille au taux de remplissage de son hôtel afin de dégager du chiffre d'affaires tout en satisfaisant la clientèle.

Changement Mot de passe

Lors de votre première connexion, il vous est conseillé de renouveler le mot de passe vous ayant été fourni par l'administrateur.

Changez votre Mot de passe !

Changement mot de passe

Manager

Nouveau mot de passe

Envoyer

Retour au compte

```
class ManagerChangePasswordController extends AbstractController
{
    #[Route('/manager/{id}/change-password', name: 'app_manager_change_password')]
    public function index(ManagerRegistry $doctrine,
        Request $request,
        UserPasswordHasherInterface $userPasswordHasher,
        EntityManagerInterface $entityManager,
        int $id): Response
    {
        $manager = $doctrine->getRepository(Manager::class)->find($id);
        $form = $this->createForm(type: ChangePasswordType::class, $manager);
        $form->handleRequest($request);

        if ($form->isSubmitted() && $form->isValid()) {
            $manager->setPassword(
                $userPasswordHasher->hashPassword(
                    $manager,
                    $form->get('plainPassword')->getData()
                )
            );
            $entityManager->persist($manager);
            $entityManager->flush();
            return $this->redirectToRoute(route: 'app_success_change_password');
        }

        return $this->render(view: 'form/manager-change-password.html.twig', [
            'title' => 'Hypnos - Changement de mot de passe',
            'manager' => $manager,
            'form' => $form->createView(),
        ]);
    }
}
```


Réalisation du projet

Gestion

➡ Page de gestion Utilisateur:



```
class UserGestionController extends AbstractController
{
    #[Route('/user/{id}', name: 'app_user_gestion')]
    public function index(ManagerRegistry $doctrine,
        int $id): Response
    {
        $suite = $doctrine->getRepository( persistentObject: Suite::class)->findAll();
        $booking = $doctrine->getRepository( persistentObject: Booking::class)->findAll();
        $utilisateur = $doctrine->getRepository( persistentObject: User::class)->find($id);

        return $this->render( view: 'gestion/user.html.twig', [
            'title' => 'Hypnos - Gestion Utilisateur',
            'utilisateur' => $utilisateur,
            'booking' => $booking,
            'suite' => $suite,
        ]);
    }
}
```

Réalisation du projet

Utilisateur

Utilisateur

Gestion


- Votre prénom : User
- Votre nom : Un
- Votre mail : User1-Hypos@gmail.com

En tant qu'Utilisateur, il vous est permis de parcourir les listes des hotels et suites mis à votre disposition. Vous pourrez, à partir de ces pages, réserver une ou plusieurs suites, à des dates précises.

Réervations futures

Suite

Vous pouvez Annuler une réservation au maximum 72 Heures avant la date d'échéance. Au delà, il faudra contacter le service Administratif afin de trouver une solution!



Suite	Fait le	Date début	Date fin	Annuler
Charlemagne II	18-09-2022	15-11-2022	18-12-2022	<div>Modifier</div>

Réervations annulées

Suite



Ci-dessous, vous pouvez voir toutes les réservations qui ont annulées.

Suite	Fait le	Date début	Date fin	Supprimer
La Luxuriante	17-09-2022	01-08-2022	05-09-2024	<div>Supprimer</div>

Réalisation du projet

Exemple : Booking

```
class BookingController extends AbstractController
{
    #[Route('/booking', name: 'app_booking')]
    public function index(Request $request,
        EntityManagerInterface $entityManager,
        ManagerRegistry $doctrine): Response
    {
        $user = $this->getUser();
        if (!$this->getUser()) {
            return $this->redirectToRoute( route: 'app_login');
        }
        $booking = new Booking();
//Form
        $form = $this->createForm( type: BookingFormType::class, $booking);
        $form->handleRequest($request);
        if ($form->isSubmitted() && $form->isValid()) {
            $datetime = new \DateTimeImmutable();
            $booking->setDidAt($datetime);
            $booking->setClient($user);
            $booking->setIsActive( isActive: true);
//Verification dispo
            $start_date = $request->request->get( key: 'start_date');
            $end_date = $request->request->get( key: 'start_end');
            $isPossibleBooking = $entityManager->getRepository(Booking::class)->isPossible($start_date, $end_date);
//Push
            if($isPossibleBooking) {
                $booking = $form->getData();
                $entityManager->persist($booking);
                $entityManager->flush();
                return $this->redirectToRoute( route: 'app_success_reservation');
            } else {
                $this->addFlash( type: 'error', message: 'La suite choisie n\'est pas disponible à ces dates. ');
                unset($newBooking);
            }
        }
        return $this->render( view: 'form/booking.html.twig', [
```

Réalisation du projet

Dynamisme

- Dynamisme dans mes pages:
 - JavaScript
 - Live Component

Réalisation du projet

JavaScript

➡ Différentes Fonctions :

```
require('bootstrap-datepicker/js/bootstrap-datepicker')
require('bootstrap-datepicker/js/locales/bootstrap-datepicker.fr')
require('bootstrap-datepicker/dist/css/bootstrap-datepicker.min.css')

$(document).ready(function() {
  $('.js-datepicker').datepicker({
    format: 'yyyy-mm-dd'
  });
});
```

```
function showConfirm() {
  var confirmation = confirm("Voulez vous faire cette action?");
  if (confirmation === true) {
    alert("vous avez cliqué OK");
    return true;
  } else {
    alert("Opération annulée");
    return false;
  }
}
```

Réalisation du projet

Live Component

- Live Component :
 - Jeux d'essai !
 - Recherche asynchrone désirée
 - Problème lors de la création de ma SearchBar
 - Erreur lors du retour mapping
 - Composant expérimental

</>

```
composer require symfony/ux-live-component
```

```
composer require symfony/ux-twig-component
```

</>

Réalisation du projet

Jeux d'essai

- Création d'un component Suite
- Lier un objet Suite à un modèle réutilisable
- Composant flexible

```
{{ component('suite', {'id': suite.id}) }}
```

Réalisation du projet

Jeux d'essai

- Création du Live Component
- Récupération d'informations grâce à une QueryBuilder

```
public function findByQuery(string $query): array
{
    if (empty($query)) {
        return [];
    }

    return $this->createQueryBuilder( alias: 'b')
        ->andWhere('b.name LIKE :query')
        ->setParameter( key: 'query', value: '%'.$query.'%')
        ->orderBy( sort: 'b.id', order: 'ASC')
        ->getQuery()
        ->getResult()
        ;
}
```

```
#[AsLiveComponent('suite_search')]
class SuiteSearchComponent
{
    use DefaultActionTrait;

    #[LiveProp(writable: true)]
    public string $query = '';

    public function __construct(
        private SuiteRepository $suiteRepository
    ) {
    }

    public function getSuites(): array
    {
        return $this->suiteRepository->findByQuery($this->query);
    }

    public function getAllSuites(): array
    {
        return $this->suiteRepository->findAll();
    }
}
```

Réalisation du projet

Jeux d'essai

- Création de l'input
- Gestion du rendu grâce à une condition Twig

```
{% extends 'layout.html.twig' %}
{% block title %}{{title}}{% endblock %}
{% block body %}
    {{ component('suite_search') }}
{% endblock %}
```

```
<div{{ attributes }}>
    <input
        class="input-search-suite"
        type="text"
        data-model="query"
        value="{{ query }}"
        placeholder="Rechercher une suite..."
    >
    {% if query is empty %}
        {% for suite in this.AllSuites %}
            {{ component('suite', {
                id: suite.id
            }) }}
        {% endfor %}
    {% else %}
        {% for suite in this.suites %}
            {{ component('suite', {
                'id': suite.id
            }) }}
        {% endfor %}
    {% endif %}
</div>
```

Réalisation du projet

Jeux d'essai

The screenshot shows a web browser window displaying a website for 'Hypnos'. The website has a light blue header with a logo 'H' and navigation buttons 'Accueil' and 'Connexion'. The main content area features a large image of two people sitting on a rocky outcrop overlooking a valley, with the text 'Hypnos' and 'Voyagez, ensemble.' overlaid. Below this is a section titled 'Qui sommes-nous?' with a 'Présentation' button. At the bottom, there is a 'Nos Hotels' button and a paragraph of text about the hotel group.

The Chrome DevTools console on the right shows several error messages:

- Expression non disponible
- Échec du chargement de la carte source par les Outils de développement : Impossible de charger le contenu de chrome-extension://cfdpdpk1hmkibokdaibdcdd/iiifddp/browser-polyfill.js.map : Erreur système: net::ERR_FILE_NOT_FOUND
- application #starting stimulus.js:1723
- application #start stimulus.js:1723
- Échec du chargement de la carte source par les Outils de développement : Impossible de charger le contenu de chrome-extension://mmhniccooindimnjhamobqdhaoime/iib/tp/browser-polyfill.min.js.map : Erreur système: net::ERR_BLOCKED_BY_CLIENT

Réalisation du projet

Sécurité

- Sécurité :
 - Déjà expliqué :
 - Hachage du mot de passe
 - Validation des contraintes formulaires
 - Moteur de template TWIG
 - Injection SQL
 - Jeton csrf

```
<input type="hidden" name="_csrf_token" value="{{ csrf_token('authenticate') }}">
```

Réalisation du projet

Sécurité

- security.yaml:
 - Création de la hiérarchie des rôles
 - En fonction de l'URL

```
role_hierarchy:  
  ROLE_MANAGER: ROLE_USER  
  ROLE_ADMIN: ROLE_MANAGER  
  ROLE_SUPER_ADMIN: [ROLE_ADMIN, ROLE_ALLOWED_TO_SWITCH]
```

```
access_control:  
  - { path: ^/booking, roles: ROLE_USER }  
  - { path: ^/user, roles: ROLE_USER }  
  - { path: ^/add-picture-list, roles: ROLE_MANAGER }  
  - { path: ^/success-change-password, roles: ROLE_MANAGER }  
  - { path: ^/manager, roles: ROLE_MANAGER }  
  - { path: ^/manager, roles: ROLE_MANAGER }  
  - { path: ^/remove, roles: ROLE_MANAGER }  
  - { path: ^/add-suite, roles: ROLE_ADMIN }  
  - { path: ^/admin, roles: ROLE_ADMIN }  
  - { path: ^/add-manager, roles: ROLE_ADMIN }  
  - { path: ^/add-hotel, roles: ROLE_ADMIN }
```

Réalisation du projet

Sécurité

- Contraintes de validation sur les Entités :

```
#[ORM\Column(length: 255)]
#[Assert\Image(
    minWidth: 100,
    maxWidth: 2600,
    maxHeight: 2600,
    minHeight: 100,
)]
private ?string $image = null;
```

```
#[Assert\GreaterThan("today",
    message: "La date d'arrivée doit être ultérieur à la date d'aujourd'hui")]
#[ORM\Column(type: Types::DATE_MUTABLE)]
private ?\DateTimeInterface $start_date = null;

#[ORM\Column(type: Types::DATE_MUTABLE)]
#[Assert\GreaterThan(propertyPath: "start_date",
    message: "La date de départ doit être plus éloignée que la date d'arrivée !")]
private ?\DateTimeInterface $end_date = null;
```

Réalisation du projet

Sécurité

- Côté utilisateur :
 - Politique de confidentialité
 - Mentions légales

Protection des données

Information

Mentions
légales

Pol.
Confidentialité

Réalisation du projet

Déploiement

- Mise en production de l'application :
 - Creation d'un nouveau répertoire sur le site Heroku
 - Creation du fichier .env.local
 - Compilation des assets

```
</>
```

```
npm run build
```

```
composer dump-env prod
```

```
</>
```

Réalisation du projet

Déploiement

- Création du répertoire Heroku :
 - Connexion à la plateforme Heroku
 - Sélection du répertoire créé précédemment
 - Envoi de la branche « main » de notre repository git

```
</>
```

```
heroku login
```

```
heroku git:remote hypnos-app
```







```
Git push heroku main
```

```
</>
```

Réalisation du projet

Déploiement

- Mise en place de la BDD sur serveur distant :
 - Installation d'extension (JawsDB)

 JawsDB MySQL  Kitefin Shared jawsdb-transparent-97022	 Librato  Development librato-fitted-19394
 Papertrail  Choklad papertrail-polished-30430	

- Configuration des informations serveurs dans mon dossier .env
- Utilisation de MysqlDump pour l'envoi de données

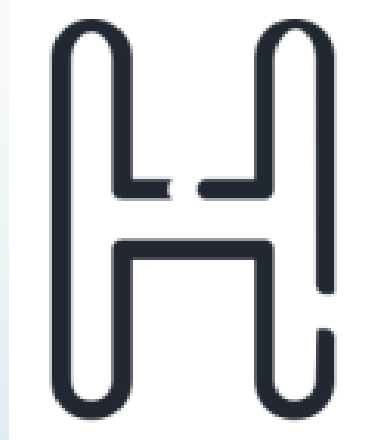
```
Mysqldump -u nomUser -pmdp nomDatabase < hypnos.sql
```

</>

</>

Réalisation du projet

Finalité



L'application est maintenant mise en ligne,
et en état de marche !



Recherche Anglophone

- Mise en place du Système d'Authentification
- Tiré de Symfony

Recherche Anglophone

Anglais

The firewalls section of `config/packages/security.yaml` is the most important section.

A "firewall" is your authentication system: the firewall defines which parts of your application are secured and how your users will be able to authenticate (e.g. login form, API token, etc).

Only one firewall is active on each request: Symfony uses the pattern key to find the first match (you can also match by host or other things).

The dev firewall is really a fake firewall: it makes sure that you don't accidentally block Symfony's dev tools - which live under URLs like `/_profiler` and `/_wdt`.

All real URLs are handled by the main firewall (no pattern key means it matches all URLs). A firewall can have many modes of authentication, in other words, it enables many ways to ask the question "Who are you?".

Often, the user is unknown (i.e. not logged in) when they first visit your website.

Recherche Anglophone

Anglais

La section *pare-feu* de `config/packages/security.yaml` est la partie la plus importante.

Un “firewall” est votre système d’authentification : le *pare-feu* définit quelle partie de votre application est sécurisée et comment vos utilisateurs pourront s’authentifier (Ex : formulaire d’inscription, jeton API, etc.).

Seul un seul *pare-feu* est actif par requête : Symfony utilise la clef du patron pour trouver la première correspondance (vous pouvez aussi correspondre par hôte ou par autre chose (?)).



Merci de votre écoute !

Ma Conclusion.



Merci de votre écoute !

Des questions ?