MOSAIC(logo)

Dossier de synthèse

Titre professionnel développeur Web et Web Mobile

Projet : « Mosaic » - 2023-2024 – Élan Formation

Molin Laëtitia

**SOMMAIRE**

1. Introduction
   1. Qui suis-je / présentation personnelle
   2. Reconversion pro
   3. Contexte
   4. Elan formation / présentation Elan formation
   5. Intro / présentation brève du projet
2. Projet / présentation du projet / processus / démarche / contexte
   1. Origine / genèse
   2. Description
   3. Enjeux
   4. Public cible (cahier des charges ?)
   5. Expressions des besoins
   6. Contraintes
3. Cahier des charges
   1. Fonctionnalités principales / fonctionnalités du site
      1. Inscription et connexion
      2. Profil utilisateur / compte utilisateur
      3. Système d’event / réservation
      4. Panneau d’amin / espace administrateur / dashboard
      5. Panneau supervisor
      6. Panneau artiste : ajout image
      7. Paiement
      8. Administration
   2. Fonctionnalités secondaires
      1. Commentaires
4. Gestion de projet / méthodologie
   1. Trello / MOSCOW
   2. Méthodologie
   3. Le MVP
5. Compétences du référentiel couvertes / compétences couvertes par le projet
   1. Compétences front-end
   2. Compétences back-end
   3. Compétences transversales
   4. Compétences personnelles acquises / travaillées
6. Technologies / environnement technique / architecture technique
   1. Front-end
   2. Back-end
   3. Logiciels
   4. Techno utilisées
   5. Langages utilisés
   6. Outils supplémentaires
   7. API, libriaries et Frameworks
7. Framework Symfony
   1. Le principe
   2. MVC
   3. Au sein du projet
   4. Principe de fonctionne de Symfony
   5. Définition d’un design pattern de Symfony
   6. Définitions de Routing dans Symfony et exemple
8. RGPD
   1. Le principe de finalité
   2. Le principe de proportionnalité et de pertinence
   3. Le principe de sécurité et de confidentialité
   4. Le principe d’une durée de conversation limitée
   5. Le droit à l’effacement ou « droit à l’oubli »
9. Modélisation
   1. Méthode Merise
      1. Mcd
      2. Mld
      3. Mpd
   2. Base de données
10. Maquettage
    1. UX  / UI
    2. Responsive design
    3. Ergonomie
    4. Navigation
    5. Fil d’ariane
11. Identité visuelle
    1. Couleurs
    2. Ergonomie
    3. Creation de component que je réutilise (sass ?)
    4. Police typographie
    5. Color scheme
    6. Logo
12. Back-end
13. Sécurité
    1. Failles (symfony)
    2. Faille upload
    3. Gestion des droits d’accès
    4. Sécurité dans l’application
    5. Veille sur les autres failles de sécurité
    6. Connexion / inscription
    7. Vérification de l’email
    8. Echappement automatique des variables
    9. Sécurisation des formulaires
    10. rôle (admin etc)
    11. Attaque par force brute, dictionnaire, arc en ciel
14. SEO
    1. Balisage
    2. Slug
    3. Accessibilité
    4. Performances
    5. Responsive Design
15. API / API platform
    1. Mise en place
    2. Exemple d’utilisation
16. Spécificités techniques
    1. API
    2. Paiements
    3. Command
    4. Ajax
    5. Rôle d’utilisateurs et Json
    6. Services
17. Traduction de texte / situation de travail ayant nécessité une recherche
    1. Contexte de la traduction / situation
    2. Texte original
    3. Traduction
18. Axes d’amélioration / difficultés rencontrées
    1. Améliorations court terme
    2. Améliorations long terme
19. Conclusion / perspectives d’évolution
20. Remerciements
21. Annexes
    1. Schémas
    2. Maquettage
    3. Capture d’écran

Table des matières

[I. PARTIE 12](#_Toc160405847)

[01. Catégorie 12](#_Toc160405848)

[a. Sous catégorie 12](#_Toc160405849)

[b. Sous catégorie 12](#_Toc160405850)

[02. Catégorie 12](#_Toc160405851)

[a. Sous catégorie 12](#_Toc160405852)

[b. Sous catégorie 12](#_Toc160405853)

[I. Introduction 13](#_Toc160405854)

[a. Présentation personnelle 13](#_Toc160405855)

[b. Présentation Elan formation 13](#_Toc160405856)

[II. Projet 14](#_Toc160405857)

[a. Origine 14](#_Toc160405858)

[b. Description 14](#_Toc160405859)

[c. Objectif et enjeux 14](#_Toc160405860)

[III. Compétences couvertes du référentiel 16](#_Toc160405861)

[01. Front-end 16](#_Toc160405862)

[02. Back-end 16](#_Toc160405863)

[03. Compétences transversales 16](#_Toc160405864)

[IV. Cahier des charges 17](#_Toc160405865)

[01. Analyse 17](#_Toc160405866)

[a. Public cible 17](#_Toc160405867)

[b. Expressions des besoins 17](#_Toc160405868)

[c. Contraintes 17](#_Toc160405869)

[02. Fonctionnalités principales 18](#_Toc160405870)

[03. Fonctionnalités secondaires 18](#_Toc160405871)

[V. Gestion de projet / Méthodologie 19](#_Toc160405872)

[01. Méthodologie 19](#_Toc160405873)

[03. Le MVP 20](#_Toc160405874)

[VI. Modélisation des données 24](#_Toc160405875)

[01.Modélisation 24](#_Toc160405876)

[a. La méthode Merise 24](#_Toc160405877)

[b. MCD – Analyse – modèle conceptuel 24](#_Toc160405878)

[c. MLD – Conception – modèle logique 26](#_Toc160405879)

[d. MPD – Réalisation – modèle physique 27](#_Toc160405880)

[02. Base de données 28](#_Toc160405881)

[a. Création dans Symfony (doctrine ?) 28](#_Toc160405882)

[b. Structure générale / liste des entités 28](#_Toc160405883)

[c. Détail entité / table 28](#_Toc160405884)

[VII. Maquettage 30](#_Toc160405885)

[01. Conception de l’interface visuelle 30](#_Toc160405886)

[a. Wireframe 30](#_Toc160405887)

[b. Mockup 31](#_Toc160405888)

[c. Prototype 31](#_Toc160405889)

[02. Considérations Responsives 32](#_Toc160405890)

[a. Approche adoptée 32](#_Toc160405891)

[b. Optimisation de l’Expérience Utilisateur (UX) 33](#_Toc160405892)

[03. Templating 33](#_Toc160405893)

[a. -- 33](#_Toc160405894)

[VIII. Identité visuelle / direction artistique 35](#_Toc160405895)

[01. Catégorie 35](#_Toc160405896)

[a. Sous catégorie 35](#_Toc160405897)

[b. Sous catégorie 35](#_Toc160405898)

[02. Catégorie 35](#_Toc160405899)

[a. Sous catégorie 35](#_Toc160405900)

[b. Sous catégorie 35](#_Toc160405901)

[IX. RGPD 36](#_Toc160405902)

[01. RGDP & CNIL 36](#_Toc160405903)

[02. Politique de Confidentialité et Consentement 37](#_Toc160405904)

[a. Politique de Confidentialité 37](#_Toc160405905)

[b. Consentement de L’Utilisateur 37](#_Toc160405906)

[02. Droits des Utilisateurs 38](#_Toc160405907)

[a. Droit à l’Information 38](#_Toc160405908)

[b. Droit d’Accès et de Rectification 39](#_Toc160405909)

[c. Droit à l’oubli et droit à l’effacement 39](#_Toc160405910)

[02. Collecte et Traitement des Données 39](#_Toc160405911)

[a. Principe de minimisation 39](#_Toc160405912)

[a. Finalités de Collecte / principe de finalité 39](#_Toc160405913)

[a. Le principe de proportionnalité et de pertinence 39](#_Toc160405914)

[a. Le principe de sécurité et de confidentialité 39](#_Toc160405915)

[b. Durée de Conversation / Le principe d’une durée de conservation limitée 40](#_Toc160405916)

[c. Traitement des Données Sensibles 40](#_Toc160405917)

[X. Technologies / structure technique 40](#_Toc160405918)

[01. Front-End 41](#_Toc160405919)

[a. HTML5 et Css3 41](#_Toc160405920)

[b. Sass 41](#_Toc160405921)

[c. JavaScript, Jquery et Ajax 41](#_Toc160405922)

[02. Back-end 42](#_Toc160405923)

[a. Php & mySql 42](#_Toc160405924)

[03. Librairie 42](#_Toc160405925)

[a. Jquery 42](#_Toc160405926)

[a. React 42](#_Toc160405927)

[04. API 42](#_Toc160405928)

[a. Stripe 42](#_Toc160405929)

[b. Leatlet & Denominati 42](#_Toc160405930)

[05. Autre 42](#_Toc160405931)

[a. Gestion de projet et développement 42](#_Toc160405932)

[c. Graphisme 42](#_Toc160405933)

[XI. Framework : Symfony 44](#_Toc160405934)

[01. Fonctionnement 44](#_Toc160405935)

[a. Doctrine 46](#_Toc160405936)

[a. Moteur de Template Twig 46](#_Toc160405937)

[a. Bundles 47](#_Toc160405938)

[a. Composer 47](#_Toc160405939)

[a. Service 47](#_Toc160405940)

[02. Design pattern 48](#_Toc160405941)

[a. 49](#_Toc160405942)

[a. 49](#_Toc160405943)

[03. Routing 49](#_Toc160405944)

[a. 49](#_Toc160405945)

[a. 49](#_Toc160405946)

[04. Au sein du projet 49](#_Toc160405947)

[a. fonctionnalité principale 49](#_Toc160405948)

[a. services ? bundle? Command ? securité login / identification / isverified mail qqchose / rôle / token – sécurité , form 49](#_Toc160405949)

[a. Présentation 50](#_Toc160405950)

[c. Symfony eu sein du projet 51](#_Toc160405951)

[a.Démonstration d’une fonctionnalité – mvc 51](#_Toc160405952)

[b.Exemple d’une requête personnalisée 51](#_Toc160405953)

[XII. SEO – Search Engine Optimisation 52](#_Toc160405954)

[01. Introduction au SEO et Référencement 52](#_Toc160405955)

[a. Définition du SEO 52](#_Toc160405956)

[b. Objectifs du référencement 52](#_Toc160405957)

[02. Aspects Front-End du SEO 53](#_Toc160405958)

[a. Structure HTML Sémantique 53](#_Toc160405959)

[b. Optimisation des Images 53](#_Toc160405960)

[b. URLs Conviviales 54](#_Toc160405961)

[d. Balisage des Données Structurées 54](#_Toc160405962)

[e. Responsive design 54](#_Toc160405963)

[03. Aspects Back-End du SEO 54](#_Toc160405964)

[a. Gestion des Redirects 54](#_Toc160405965)

[b. Gestion des Erreurs HTPP 54](#_Toc160405966)

[c. Optimisation de la Vitesse du Chargement 54](#_Toc160405967)

[d. Propreté des Liens 55](#_Toc160405968)

[e. Web Crawling 55](#_Toc160405969)

[d. Sitemap XML et Robots.txt 55](#_Toc160405970)

[04. Contenu de Qualité et Stratégie de Mots-clefs / Stratégies de Contenu de SEO 55](#_Toc160405971)

[a. Recherche de Mots-clefs 55](#_Toc160405972)

[b. Eléments On-Page 56](#_Toc160405973)

[b. Liens Externes et Internes 56](#_Toc160405974)

[b. Contenu unique et Engageant 56](#_Toc160405975)

[05. Suivi et Analyse 56](#_Toc160405976)

[a. Intégration d’Outils d’Analyse 56](#_Toc160405977)

[b. Analyse des Données 56](#_Toc160405978)

[Conclusion 56](#_Toc160405979)

[XIII. Sécurité 62](#_Toc160405980)

[01. Gestion des failles et vulnérabilités 62](#_Toc160405981)

[a. Prévention des Attaques par Injection SQL 62](#_Toc160405982)

[b. Protections contre les failles XSS (Cross-Site Scripting) 65](#_Toc160405983)

[c. Faille CSRF (Cross-Site Request Forgery) / 66](#_Toc160405984)

[c. Sécurité lors de l’Upload de fichiers 67](#_Toc160405985)

[d. Détection d’actions suspectes avec le Honey Pot 70](#_Toc160405986)

[02. Gestion des mots de passe 70](#_Toc160405987)

[a. Hachage du mot de passe 70](#_Toc160405988)

[b. Politique du mot de passe 73](#_Toc160405989)

[b. Limitation des tentatives de connexion 74](#_Toc160405990)

[a. Captcha ? 76](#_Toc160405991)

[02. Authentification & Autorisation / Gestion des Accès et Authentification 76](#_Toc160405992)

[a. Connexion / / vérification par email / session !! 76](#_Toc160405993)

[b. Authentification (forte) 77](#_Toc160405994)

[c. Autorisation 77](#_Toc160405995)

[b. Contrôle des accès 77](#_Toc160405996)

[d. Gestion des droits d’accès:/ d’utilisateurs – Autorisation et gestion des droits d’accès 77](#_Toc160405997)

[user non co, user co, artist, supervisor, admin 77](#_Toc160405998)

[e. Les différents rôles 77](#_Toc160405999)

[04. Sécurité dans l’application 78](#_Toc160406000)

[a. La gestion de la sécurité avec Symfony 78](#_Toc160406001)

[b. La sécurisation des formulaires 78](#_Toc160406002)

[a. Sécurisation des URLs / gestion error 404 78](#_Toc160406003)

[a. Sécurité système de paiement 78](#_Toc160406004)

[XIV. Traduction de texte 79](#_Toc160406005)

[01. Contexte de la traduction 79](#_Toc160406006)

[02. Texte original 79](#_Toc160406007)

[03. Traduction 79](#_Toc160406008)

[XV. Axes d’améliorations 79](#_Toc160406009)

[01. Améliorations court terme 79](#_Toc160406010)

[02. Amélioration long terme 79](#_Toc160406011)

[XVI. Conclusion et Remerciements 79](#_Toc160406012)

[01. Conclusion / défis ? 79](#_Toc160406013)

[02. Remerciements 79](#_Toc160406014)

[XVII. Annexes 80](#_Toc160406015)

[01. Schémas 80](#_Toc160406016)

[02. Maquettage 80](#_Toc160406017)

[02. Capture d’écran 80](#_Toc160406018)

[Notes : 81](#_Toc160406019)

# I. PARTIE

## 01. Catégorie

### a. Sous catégorie

### b. Sous catégorie

## 02. Catégorie

### a. Sous catégorie

### b. Sous catégorie

# I. Introduction

### a. Présentation personnelle

qui suis je / reconversion pro

### b. Présentation Elan formation

# II. Projet

### a. Origine

L'idée de ce site web trouve son origine dans ma passion personnelle pour l'art, qui a toujours occupé une place centrale dans ma vie. Depuis toujours, j'expérimente dans divers domaines artistiques, que ce soit à travers des études effectuées dans ce domaine ou par mes hobbies personnels. Cette immersion continue qui m'a permis de comprendre les défis auxquels les artistes sont confrontés, notamment en termes de partage d'espaces, d'équipements et de visibilité.

De ces constatations est née l'ambition de créer un espace communautaire virtuel dédié aux artistes, permettant de pallier ces manques. Ce site fictif serait à la fois une plateforme de partage et un catalyseur d'opportunités, envisagé comme un lieu où la création, l'échange et la collaboration artistique pourraient s'épanouir librement.

### b. Description

Le site que je propose vise à être un carrefour pour la communauté artistique. Il permettra non seulement de partager des locaux et des équipements mais aussi d'organiser des événements tels que des expositions, des workshops et d'autres manifestations artistiques. Le but est de créer un espace en ligne où les artistes peuvent se mettre en avant, découvrir et être découverts.

Le site comprendra plusieurs fonctionnalités clés : une plateforme pour mettre en avant les profils des artistes, une section dédiée à la réservation et au partage de ressources, ainsi qu'un espace dashboard administratif pour la gestion et la création d'activités artistiques.

Cet espace dynamique et interactif est conçu pour répondre aux besoins de la communauté artistique dans toute sa diversité. Cette plateforme serait conçue pour être intuitive, engageante, et facile d'utilisation, s'adressant aussi bien aux étudiants en art, professionnels, qu'aux amateurs passionnés.

Le nom du projet a pour but de refléter ce concept, ainsi "Mosaic", évoque la vision d'une communauté artistique dynamique, où chaque individu, tel un fragment de mosaïque, contribue à former une toile artistique plus vaste et variée. Comme une mosaïque qui rassemble des pièces diverses pour créer une œuvre cohérente, notre plateforme aspire à être un carrefour où les artistes peuvent se connecter, collaborer et partager leurs talents, formant ainsi un ensemble harmonieux et stimulant.

### c. Objectif et enjeux

"Mosaic" vise à être bien plus qu'un simple site web ; il s'agit d'un lieu d'échange et d'expression où les artistes peuvent non seulement mettre en avant leur travail, mais aussi découvrir de nouveaux horizons artistiques, participer à des événements stimulants et bénéficier d'un soutien mutuel au sein d'une communauté soudée, créer un pont entre les mondes artistiques et de susciter une curiosité culturelle plus large

En alignement avec cette vision, nos objectifs sont multiples :

1. **Favoriser l'accès à l'art et à la création artistique** pour tous, en levant les barrières matérielles et en offrant des opportunités de collaboration et d'exposition et un accès à des espaces de travail et équipements.
2. **Créer une communauté artistique soudée et dynamique**, où le partage de connaissances et d'expériences enrichit chacun de ses membres, permettant de tisser des liens entre les artistes, favorisant ainsi une dynamique de collaboration et de soutien mutuel et faciliter la mise en relation entre artistes de divers horizons
3. **Stimuler la créativité et l'innovation** en proposant un espace où les nouvelles idées et les projets collaboratifs peuvent prendre vie.
4. **Valoriser et démocratiser l’art** en organisant des événements et des expositions, le site contribue à la valorisation de l'art et des artistes, mais également permettre la découverte de divers domaines artistiques à un public élargi : À travers l'organisation de workshops, d'événements et d'expositions, le site vise à éveiller la curiosité et à enrichir les connaissances artistiques du grand public, y compris ceux qui sont moins familiers avec le monde de l'art.
5. **Offrir une visibilité accrue aux artistes**, en leur fournissant une plateforme pour exposer leur travail et interagir avec le public ainsi qu'avec d'autres créateurs.
6. **Développer une ressource durable pour la gestion et l'organisation d'activités artistiques**, simplifiant ainsi la création d'événements culturels de toutes sortes.

Ces objectifs convergent vers la création d'un espace communautaire actif, polarisant et stimulant, qui serve de catalyseur pour le partage et la création artistique. Nous envisageons un lieu où la diversité des expressions artistiques est non seulement célébrée mais aussi activement encouragée, créant ainsi un écosystème dynamique où chaque interaction enrichit le tissu culturel global.

Les objectifs à long terme incluent l'expansion de la plateforme pour intégrer des fonctionnalités innovantes qui enrichiront l'expérience utilisateur et renforceront la communauté artistique.

# III. Compétences couvertes du référentiel

Les principaux objectifs de ce projet consistent à acquérir les compétences nécessaires pour obtenir le titre professionnel de **«** **Développeur Web et Web Mobile »**. Dans le cadre de mon application, j'ai également exploré l'approche **orientée objet avec PHP** et le **Framework Symfony**, élargissant ainsi mes connaissances et permettant le déploiement d'une application plus efficace et performante.

SCREEN : Screen langages (github)

**Le référentiel d'emploi, d'activités et de compétences** (REAC) du titre professionnel de Développeur Web et Web Mobile définit l'ensemble des compétences à valider pour obtenir ce titre. Selon le REAC, ces compétences englobent à la fois le **front-end** et le **back-end**.

## 01. Front-end

Développer la partie **front-end** d'une application web ou web mobile en intégrant les recommandations de sécurité

* Maquetter une application.
* Réaliser une interface utilisateur web statique et adaptable.
* Développer une interface utilisateur web dynamique.
* Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce.

## 02. Back-end

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

* Créer une base de données.
* Développer les composants d'accès aux données.
* Développer la partie back-end d'une application web ou web mobile.
* Elaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce.

## 03. Compétences transversales

* Utiliser l’anglais dans son activité professionnelle en développement web et web mobile.
* Actualiser et partager ses compétences en développement web et web mobile.

# IV. Cahier des charges

Dans cette section, nous définirons les exigences et les spécifications du projet "Mosaic". Le cahier des charges servira de guide pour le développement et la mise en œuvre de la plateforme, en tenant compte des besoins de notre public cible, des contraintes techniques et des fonctionnalités attendues.

## 01. Analyse

### a. Public cible

Le public cible est diversifié et comprend :

* Les artistes de tous niveaux : des étudiants en art aux professionnels établis, cherchant un espace pour partager leur travail, collaborer avec d'autres artistes et accéder à des ressources et des opportunités.
* Les passionnés d'art : des amateurs et des curieux désireux d'explorer différents domaines artistiques, de participer à des événements culturels et d'interagir avec la communauté artistique.
* Les administrateurs et gestionnaires d'événements artistiques : des personnes chargées de coordonner et de gérer des expositions, des workshops et d'autres activités artistiques.

### b. Expressions des besoins

Les principaux besoins exprimés par notre public cible sont les suivants :

* **Facilité d'utilisation :** Une interface conviviale et intuitive pour naviguer facilement à travers la plateforme et accéder aux fonctionnalités principales.
* **Connectivité et partage :** La possibilité de se connecter avec d'autres artistes, d'organiser des événements et de collaborer sur des projets artistiques.
* **Visibilité :** Une vitrine pour exposer leur travail, accéder à des opportunités d'exposition et de promotion, et interagir avec un public plus large.
* **Gestion efficace des activités :** Des outils administratifs pour gérer et organiser des événements, des réservations d'espaces, ainsi que pour suivre les interactions et les engagements des membres.

### c. Contraintes

Les contraintes à prendre en compte dans le développement de la plateforme incluent :

* La nécessité de concevoir une plateforme web robuste, sécurisée et adaptable à différents appareils et navigateurs.
* De par la diversité de la cible, la nécessité de concevoir une plateforme intuitive et facilement appréhensible par tous
* La nécessité de concevoir une plateforme visuellement attrayante pour refléter son thème artistique et susciter l'engouement des utilisateurs. Cela nécessite une attention particulière à l'ergonomie, au design et à l'esthétique de l'interface utilisateur.

## 02. Fonctionnalités principales

Les fonctionnalités principales de "Mosaic" comprennent :

* **Enregistrement et connexion :** Un processus simple et sécurisé d'inscription permettant aux utilisateurs de créer un compte personnel sur la plateforme
* **Gestion des événements :** Un système de réservation et de promotion pour les expositions, les workshops et autres événements artistiques.
* **Dashboard administratif :** Un espace dédié aux administrateurs pour gérer les activités, les membres et les contenus de la plateforme.
* **Profil d'artiste :** Un espace personnalisé pour les artistes afin de présenter leur travail, leur portfolio et leur biographie.
* **Gestion des rôles et des autorisations :** La création de différents rôles utilisateur avec des autorisations spécifiques, tels que simple utilisateur, artiste, administrateur et superviseur.

## 03. Fonctionnalités secondaires

En plus des fonctionnalités principales, "Mosaic" inclura également des fonctionnalités secondaires pour améliorer l'expérience utilisateur, telles que :

* **Calendrier d'événements :** Un calendrier interactif pour consulter et s'inscrire aux événements à venir.
* **Système d'abonnement** : Ce système permet aux utilisateurs de s'abonner et de participer aux workshops organisés par la plateforme.
* **Intégration d’une carte interactive :** Cette fonctionnalité permet aux artistes de partager leurs boutiques physiques ou leurs lieux d'exposition.

En résumé, le cahier des charges de "Mosaic" établit les fondations pour la création d'une plateforme dynamique et inclusive, répondant aux besoins diversifiés de la communauté artistique qui stimulera la créativité et la collaboration au sein de notre communauté.

# V. Gestion de projet / Méthodologie

Dans le parcours de développement de toute application, la méthodologie de gestion de projet joue un rôle clé. Elle guide les étapes, optimise les ressources et assure une progression efficace vers la réalisation des objectifs. La gestion de projet est l'ensemble des activités visant **à organiser, planifier, et superviser les ressources et les tâches nécessaires pour atteindre les objectifs d'un projet**.

Afin d’organiser le travail à fournir je me suis basée sur le méthode MoSCoW et sur le MVP, méthodologie repose notamment sur le concept de MVP (Minimal Viable Product), une approche stratégique visant à livrer un produit fonctionnel et essentiel dès les premières phases du développement. Cela nous a permis de concentrer nos efforts sur les fonctionnalités clés et d'obtenir rapidement un retour d'information précieux des utilisateurs, orientant ainsi le développement futur.

## 01. Méthodologie

MoSCoW est une **technique de priorisation des exigences dans la gestion de projet.** Les éléments sont classés en quatre catégories : Must-haves (doit avoir), Should-haves (devrait avoir), Could-haves (pourrait avoir), et Won't-haves (ne doit pas avoir). Cela aide à hiérarchiser les exigences du projet en fonction de leur importance et permet une meilleure **simplicité, clarté dans la définition des priorités** permettant également d’orienter **les efforts sur les fonctionnalités essentielles** du projet.

Elle a été intégrée dans notre gestion de projet pour déterminer les exigences essentielles, importantes, facultatives et non nécessaires.

Trello, un outil de gestion de projet basé sur des tableaux, a été notre choix pour organiser, attribuer et suivre les tâches au quotidien. Son interface intuitive a facilité la collaboration au sein de l'équipe en offrant une vue claire des activités. De la planification à l'exécution, Trello a été essentiel pour maintenir une dynamique de travail efficace.

Must-Have : (doit avoir)

Should have (devrait avoir)

Could-Have (pourrait avoir)

Won’t-Have (N’aura pas)

**Gestion des rôles et des autorisations :** La création de différents rôles utilisateur avec des autorisations spécifiques, tels que simple utilisateur, artiste, administrateur et superviseur. Chaque rôle aura des privilèges définis pour accéder et gérer certaines fonctionnalités de la plateforme, garantissant ainsi une gestion efficace et sécurisée des activités.

## 03. Le MVP

La méthodologie MVP (Minimum Viable Product) consiste à développer une version **minimale d'un produit avec les fonctionnalités essentielles pour le rendre utilisable.** Cela permet de lancer rapidement un produit sur le marché, **d'obtenir des retours des utilisateurs**, et **d'itérer pour améliorer le produit en fonction de retours**. Les avantages **sont la réduction des coûts de développement et la validation précoce de l’idée du produit.**

L'adoption du modèle MVC a structuré notre code de manière modulaire, favorisant la séparation des préoccupations. Cette architecture a permis une maintenance plus aisée, tout en rendant notre application artistique extensible et adaptable aux évolutions futures.

Dans mon cas, j’ai considéré les objectifs des parties « Must Have » ainsi « Should Have » comme objectifs à atteindre pour obtenir un produit minimum viable.

Github propose de nombreux outils d’analyse pour les projets suivis, permettant d’avoir une vision globale sur le déroulement du développement :

Git ?

**Introduction à Git et GitHub :** Expliquez brièvement ce qu'est Git, un système de contrôle de version décentralisé, et GitHub, une plateforme de gestion de code source basée sur Git. Soulignez l'importance de ces outils dans le suivi des modifications du code source, la gestion des branches, et la collaboration au sein de l'équipe de développement.

**Utilisation de Git dans le Projet :** Détaillez comment Git a été utilisé pour gérer les versions de votre application. Par exemple, mentionnez la création et la fusion de branches pour développer de nouvelles fonctionnalités tout en maintenant une version stable du code principal (branche principale ou "master"). Expliquez comment les commits ont été utilisés pour documenter les changements apportés au code.

**GitHub comme Plateforme de Collaboration :** Si votre équipe a utilisé GitHub, parlez de la manière dont cette plateforme a facilité la collaboration. Mentionnez des fonctionnalités spécifiques telles que les demandes d'extraction (pull requests) pour discuter des modifications, les problèmes (issues) pour suivre les tâches, et les actions GitHub pour automatiser les workflows.

**Outils d'Analyse de Code :** Si vous avez utilisé des outils d'analyse de code intégrés à GitHub, parlez de leur rôle dans l'amélioration de la qualité du code. Par exemple, des analyses statiques de code peuvent identifier des problèmes potentiels, des vulnérabilités de sécurité, ou des opportunités d'optimisation.

**Impact sur la Gestion de Projet :** Discutez de l'impact positif de l'utilisation de Git et GitHub sur la gestion de projet. Cela peut inclure une meilleure traçabilité des modifications, une collaboration efficace, et une gestion plus transparente des tâches et des contributions au sein de l'équipe.

En résumé, l'intégration de Git et GitHub dans la partie gestion de projet peut apporter une perspective importante sur la manière dont la collaboration et la gestion du code source ont été gérées tout au long du développement de votre application artistique.

**Définir le contrôle de version ? Qu’est-ce que Git ?**

Le contrôle de version est un système qui permet de gérer et de suivre les modifications apportées aux fichiers d'un projet au fil du temps. Il assure le suivi de l'historique des modifications, permet de revenir en arrière à n'importe quel moment (via des versions antérieures), de fusionner les modifications et facilite la collaboration entre plusieurs personnes travaillant sur un projet.

**Git :** Git est l'un des systèmes de contrôle de version les plus utilisés. Il se distingue par son caractère décentralisé, ce qui signifie que chaque copie du dépôt contient l'intégralité de l'historique des modifications.

**Branches :** sont des lignes de développement séparées dans un dépôt Git. Elles permettent de travailler sur des fonctionnalités, des correctifs ou des expériences de manière isolée par rapport à la branche principale du projet. Cela facilite le développement parallèle et l'organisation du travail d'équipe.

**Commits :** Un commit est une opération dans Git qui enregistre une modification ou un ensemble de modifications dans le dépôt accompagné d'un message décrivant la nature de la modification apportée. Les commits forment l'historique du projet.

**Dépôts (Repositories) :** est un espace où toutes les données et l'historique d'un projet Git sont stockés. Il peut être local ou distant (sur un serveur). Les dépôts permettent de collaborer avec d'autres développeurs en partageant et en synchronisant le code source.

# VI. Modélisation des données

Une fois le cahier des charges établi, la seconde étape consistait à réaliser le modèle conceptuel de données (MCD) de l'application, première étape d'abstraction qui fournit les bases nécessaires pour la création ultérieure de la base de données définitive sur laquelle s'organisera mon application.

## 01.Modélisation

### a. La méthode Merise

La modélisation des données est une approche visant à représenter la structure et les relations des données d'un système. Elle est guidée par la méthodologie Merise, qui repose sur trois étapes claires. Développée en France dans les années 1970 et 1980, cette méthode vise à structurer le processus de conception des systèmes d'information en étapes claires et logiques. Elle favorise une approche par abstraction, permettant de modéliser les concepts de plus haut niveau avant de se concentrer sur les détails de mise en œuvre.

Les avantages de la méthode Merise résident dans sa structuration méthodique et son approche évolutive, offrant une meilleure compréhension des besoins du système et de ses composants, tout en permettant d'ajuster les modèles en fonction des évolutions des besoins.

### b. MCD – Analyse – modèle conceptuel

**Le modèle conceptuel de données** (MCD) en Merise **est une représentation abstraite des entités, de leurs attributs et des relations entre elles dans un système.** Il offre une vision globale et simplifiée de la structure des données **sans tenir compte des détails de mise en oeuvre technique.** Le MCD permet de définir les concepts clés du domaine d'application et est utilisé pour développer ls modèles logique et physique de données dans le processus de modélisation Merise.

Cette étape consiste à comprendre les besoins et les objectifs du système, à identifier les entités, leurs attributs et les interactions entre elles afin de créer un modèle conceptuel de données (MCD). Le MCD est une représentation abstraite, sous forme graphique, des données et de leurs relations, permettant de définir conceptuellement :

**Les Entités :** Dans cette phase, on repère les objets ou concepts importants du système, appelés entités, qui seront ensuite modélisés dans la base de données.

représente **un objet du monde réel**, **identifiable et ayant des caractéristiques propres.** Généralement représenté par une table dans une base de données relationnelle. Où chaque ligne représente une **occurrence unique** de cette entité.

**Les Attributs :** Chaque entité a des caractéristiques propres appelées attributs, définissant les propriétés ou les détails spécifiques liés à cette entité

**Les Relations :** On spécifie comment les entités interagissent entre elles. Les relations représentent les liens logiques entre les entités.

**association entre deux ou plusieurs entités** dans un modèle de donnée. Elle représente une **connexion sémantique ou une interaction entre les entités**. Les relations sont représentées par des **liens ou des jointures** entre les tables d’une base de données.

Il existe trois types de relations :

* **One-to-One :** Chaque enregistrement dans la première table est lié à un seul enregistrement dans la deuxième table, et vice versa.
* **One-To-Many :** Chaque enregistrement dans la première table peut être lié à plusieurs enregistrements dans la deuxième table.
* **Many-to-Many :** Les enregistrements dans les deux tables peuvent être liés à plusieurs enregistrements dans l'autre table.

Ces relations vont de pair avec la notion de cardinalité. Les cardinalités définissent le nombre d'occurrences d'une entité qui peuvent être liées à une occurrence d'une autre entité dans une relation.

cela décrit les relations entre les entités dans une relation. Elle permet de définir combien d’entités d’un côté de la relation peuvent être associés à combien d’entités de l’autre côté. Cela indique **le nombre d’occurrences d’une entité qui peuvent être associées à une occurrence d’une autre entité via une relation**, indiquant ainsi les contraintes de quantité entre les entités liées.

Il existe 4 types de cardinalités :

* **0,1** : Zéro ou une occurrence.
* **1,1** : Une et seulement une occurrence.
* **0,n** : Zéro ou plusieurs occurrences.
* **1,n** : Une ou plusieurs occurrences.

SCREEN : Screen MCD

Détailler en prenant l’exemple de l’entité user

En conclusion, le modèle conceptuel de données (MCD) constitue la première étape essentielle dans le processus de modélisation des données. Il permet de visualiser de manière abstraite les entités, attributs, relations, et cardinalités, offrant ainsi une compréhension claire des besoins du système et de ses composants, en favorisant une conception de base de données plus centrée sur les besoins métiers, indépendamment du système de gestion de base de données (SGBD). C’est une étape essentielle pour avoir une base solide pour ensuite aborder la seconde phase ; la modélisation logique de données (MLD), où le MCD abstrait sera transformé en un modèle plus détaillé et orienté vers l'implémentation de la base de données.

### c. MLD – Conception – modèle logique

Le Modèle Logique de Données (MLD) représente la seconde phase cruciale dans le processus de conception d'une base de données. Cette étape de conception transforme le Modèle Conceptuel de Données (MCD) en un MLD, offrant une représentation structurelle détaillée intégrant les éléments nécessaires à la mise en œuvre effective de la base de données. Il inclut des spécifications telles que les clés primaires, les clés étrangères, les index, et les types de données attribués à chaque entité. Ainsi, le MLD est une étape intermédiaire entre la compréhension métier abstraite du MCD et la réalisation d’une base de données dans un SGBD.

SCREEN : Screen MLD (avec type + clefs)

Pendant cette phase, les types de données sont définis, ce qui permet d'établir des directives claires pour chaque attribut. Chaque entité est dotée d'une clé primaire, également connue sous le nom d'index ou ID, un attribut garantissant l'unicité de chaque enregistrement dans la table, généralement auto-incrémenté. Cette approche assure l'unicité des données et simplifie les opérations de recherche et de liaison entre les tables.

Les clés primaires identifient de manière unique chaque enregistrement, tandis que les clés étrangères établissent des liens entre les tables en reliant les clés primaires d'une table aux clés étrangères d'une autre. Cette utilisation stratégique des clés primaires et étrangères permet de garantir l'intégrité référentielle, empêchant ainsi toute opération qui pourrait compromettre la cohérence des relations entre les tables.

**Une clé primaire** est un attribut ou un ensemble d’attributs qui permet **d’identifier de manière unique chaque entrée (occurrence) dans une table.** Elle garantit l’**unicité** des enregistrements et sert de référence pour établir des relations avec d’autres tables.

**Une clé étrangère** est un attribut ou un ensemble d’attributs dans une table qui fait référence à une clé primaire d’une autre table, permettant d’établir une **relation entre les tables** et permet d’assurer **l’intégrité référentielle** : les données dans une table faisant référence respectent les données dans la table référencée.

L'intégrité référentielle agit comme une contrainte cruciale, assurant la cohérence des données en empêchant toute action susceptible d'induire des incohérences dans les relations entre les tables. Il s'agit d'un mécanisme essentiel pour maintenir la fiabilité des données au sein de la base de données.

**12) A quoi sert une vue dans une base de données ?**

Dans le contexte des bases de données, une vue **est une représentation virtuelle d'un ensemble spécifique de données dans une ou plusieurs tables.** Elle ne stocke pas physiquement les données elle-même, mais fournit une vue structurée et filtrée des données existantes dans la base de données, permettant de garantir la cohérence des relations. Elle prévient les références brisées et maintient l’intégrité des données dans la base de données.

SCREEN : Screen zoom sur table user + table associative

Démo : clefs primaires / étrangères / relation / table associative

**6) Que devient une relation de type « Many To Many » dans le modèle logique de données ?**

Dans le modèle logique de données, la relation « Many to Many » est généralement représentée en utilisant **une table intermédiaire, appelée également table de jonction ou table associative**. Cette table associative **contient les clés primaires des entités liées**, permettant de représenter les multiples associations entre les entités.

### d. MPD – Réalisation – modèle physique

Une base de données est **un ensemble organisé et structuré de données**, généralement stockées électroniquement dans un **système de gestion de base de données (SGBD).** Elle sert à stocker, gérer, et récupérer des informations. Les bases de données sont utilisées pour stocker une grande variété de données, telles que des informations clients, des transactions financières etc. Elles sont gérées et interrogées pour répondre aux besoins d’une application ou d’un système.

Après la modélisation logique, nous concrétisons le modèle en schémas de base de données, prêts pour l'implémentation dans un système de gestion de base de données (SGBD). Notre choix pour cette application se porte sur MySQL, un SGBD relationnel qui utilise le langage SQL (Structured Query Language) pour la gestion des données, offrant fiabilité, performance, et flexibilité. SQL est spécifiquement conçu pour interagir avec les SGBD relationnels.

MySQL, en tant que SGBDR, permet la création, la maintenance, et l'interaction avec des bases de données. Les bases de données relationnelles, organisées en tables avec des relations prédéfinies, sont adaptées aux données structurées et aux systèmes où les relations entre les données sont cruciales.

**a. SQL (Structured Query Language) :** C'est **un langage conçu pour communiquer avec les systèmes de gestion de bases de données relationnelles (SGBDR).** Il est utilisé pour créer, récupérer, mettre à jour et gérer des données dans les bases de données.

**b. MySQL** est un **système de gestion de base de données** relationnelle open source très populaire. Il utilise le langage SQL pour gérer les données, offrant une grande fiabilité, performance et flexibilité.

**c. SGBD (Système de Gestion de Base de Données)** est un **logiciel ou un système qui permet de créer, de maintenir et d'interagir avec des bases de données.** Des exemples de SGBD sont MySQL, PostgreSQL, Oracle Database. Ces systèmes offrent des fonctionnalités avancées pour gérer les données de manière sécurisée et efficace.

SCREEN : Screen mySQL - + clef unique?

Parler des clefs uniques ? Vocabulaire de la base de données ?

Dans une base de données, les données sont stockées dans des **tables.** Celles-ci sont constituées de lignes appelées **enregistrements (ou entrées)** et de colonnes appelées **champs (ou attributs).**

**12) A quoi sert une vue dans une base de données ?**

**Dans le contexte des bases de données, une vue est une représentation virtuelle d'un ensemble spécifique de données dans une ou plusieurs tables. Elle ne stocke pas physiquement les données elle-même, mais fournit une vue structurée et filtrée des données existantes dans la base de données, permettant de garantir la cohérence des relations. Elle prévient les références brisées et maintient l’intégrité des données dans la base de données.**

## 02. Base de données

Nous allons détailler le processus de création de la base de données, en utilisant Symfony et Laragon comme environnement de développement local.

### a. Création dans Symfony (doctrine ?)

### b. Structure générale / liste des entités

Explication de la structure de ma base de données et des entités et comment elles sont reliées entres elle

### c. Détail entité / table

+ focus sur une entité (user ?)

**15) Qu’est ce qu’un CRUD dans le contexte d’une base de données ?**

**CRUD est un acronyme pour Create, Read, Update et Delete.** Ces quatre opérations représentent les actions de base que l'on peut effectuer sur les données d'une table ou d'une base de données :

**Create :** Créer de nouveaux enregistrements ou ajouter de nouvelles données à la base de données.

**Read :** Lire et récupérer des données existantes de la base de données.

**Update :** Mettre à jour les données existantes dans la base de données.

**Delete :** Supprimer des enregistrements ou des données de la base de données.

# VII. Maquettage

1. UX  / UI
2. Ergonomie
3. Navigation
4. Fil d’ariane

Le maquettage représente une étape cruciale dans le parcours de développement d'une application, offrant une vision préliminaire et tangible de son interface utilisateur. C'est à travers ce processus que la structure, la disposition, et l'interaction des éléments prennent forme permettant d’établir la direction artistique.

Dans le contexte de notre application dédiée au milieu artistique, le maquettage transcende le simple aspect technique pour devenir un outil central dans la création et la consolidation d'une identité graphique distinctive, déterminant l'esthétique et l'impact visuel de notre plateforme. Le maquettage devient un moyen privilégié pour donner vie à des concepts, explorer des esthétiques, et définir une interface utilisateur à la fois impactante et travaillée.

Nous avons opté pour Figma en tant qu'outil de maquettage. Figma, une plateforme collaborative de conception d'interfaces, se distingue par sa flexibilité, sa facilité d'utilisation et ses fonctionnalités interactives.

## 01. Conception de l’interface visuelle

### a. Wireframe

Un wireframe, dans le processus de conception, est une représentation schématique et simplifiée de la structure fondamentale d'une page web ou d'une application. Il se construit à l'aide d'éléments de base tels que des boîtes, des lignes et du texte, fournissant ainsi une vision simplifiée de la disposition des éléments majeurs. L'accent est mis sur la structure et l'agencement sans se préoccuper des détails visuels comme les couleurs ou les typographies.

Essentiellement, le wireframe agit comme une ébauche préliminaire de l'architecture globale de l'application. Il se distingue par son manque de détails visuels superflus et prépare le terrain pour la phase de maquettage, qui vise à donner une représentation plus réaliste de l'interface utilisateur. Grâce au wireframe, nous définissons la disposition des éléments clés tels que les boutons, les menus et les champs de formulaire, établissant ainsi les bases structurelles nécessaires avant d'entrer dans les aspects plus visuels du processus de conception.

SCREEN : Screen figma du wireframe (desktop + mobile) d’une ou deux pages principales

### b. Mockup

Le **mockup**, dans le processus de conception, occupe une place centrale en introduisant les détails visuels cruciaux tels que les couleurs, les polices et les images, apportant ainsi une représentation plus réaliste de l'application. Contrairement au wireframe qui se concentre sur la structure et la disposition des éléments, le mockup va au-delà, jouant un rôle essentiel dans l'affinement de l'esthétique et de l'expérience utilisateur.

Au cours de cette étape, j'ai débuté ma réflexion sur l'identité visuelle de mon application et son esthétique. J'ai opté pour une approche minimaliste et épurée, caractérisée par une palette de couleurs restreinte mais impactante, des choix graphiques qui seront détaillés plus amplement dans la partie dédiée à l'identité visuelle.

L'utilisation d'**une grille de mise en page** sur l'ensemble de mes pages (desktop et mobile) a été une décision stratégique pour structurer mon contenu visuel. Une grille de mise en page est une structure visuelle composée de lignes horizontales et verticales qui définissent la disposition des éléments sur une page web ou dans une application. En exploitant cette grille pour mon application m’a permis de faciliter l'organisation et l'alignement des contenus, contribuant ainsi à une présentation visuelle équilibrée, harmonieuse et structurée. Cette approche a grandement contribué à renforcer l'impact visuel et l'ergonomie de l'interface.

SCREEN : Screen figma du mockup des pages principales (desktpo + mobile)

+ screen grille de mise en page

### c. Prototype

Le prototype, dans le cadre du processus de conception, se présente comme une maquette interactive offrant une simulation du flux de l'application. Il va au-delà de la statique des wireframes et des mockups en fournissant une représentation fonctionnelle et interactive du produit ou de l'interface utilisateur. Cette approche permet une évaluation approfondie des fonctionnalités et de l'ergonomie du projet avant sa mise en œuvre complète.

La création du prototype a été une étape clé pour tester mon application auprès d'utilisateurs potentiels. J'ai sollicité des retours sur leur expérience en utilisant le prototype sur Figma, évaluant la fluidité de la navigation et recueillant des impressions générales sur l'interface. Ces retours ont été cruciaux, me permettant d'apporter les ajustements nécessaires dès la phase de prototypage. Cette approche proactive a optimisé le processus global, évitant des modifications majeures une fois l'application réellement développée. En résumé, le prototype a joué un rôle déterminant dans l'itération du design, garantissant une expérience utilisateur optimale lors de la version finale de l'application, tout en économisant du temps et des efforts.

SCREEN : Screen figma du prototype pour qqpages

## 02. Considérations Responsives

Une part significative des utilisateurs navigue sur le web via des appareils mobiles et des tablettes, l'importance du responsive design est incontestable. Selon les statistiques récentes, près de [insérer un pourcentage pertinent] de la consommation en ligne provient de ces dispositifs. Ignorer cette réalité reviendrait à compromettre l'accessibilité et l'expérience utilisateur pour une portion considérable de notre audience potentielle. Ainsi, intégrer une approche responsive dans la conception de notre application devient impératif pour garantir une présence efficace sur tous les écrans, répondant aux attentes variées de notre public, notamment dans le milieu artistique que nous ciblons.

De nos jours, la diversité des appareils connectés est omniprésente, le responsive design se distingue comme un impératif essentiel dans la conception d'applications et de sites web en garantissant une adaptation efficace des interfaces aux divers dispositifs connectés, en offrant une expérience utilisateur harmonieuse et optimale. Cette approche technique vise à créer des sites web et des applications capables de s'ajuster de manière fluide aux différentes tailles d'écrans et orientations, allant des smartphones aux ordinateurs. L'objectif fondamental est d'assurer une expérience utilisateur optimale, avec des mises en page flexibles, des images redimensionnées et d'autres techniques spécifiques.

### a. Approche adoptée

L'approche adoptée en matière de responsive design a été intégrée dès les premières phases du maquettage. La conception des maquettes a été pensée de manière à s'adapter de manière fluide et efficace aux différents dispositifs, incluant desktop, tablette et mobile. Cette démarche a été initiée dès la réalisation des wireframes et des mockups sur Figma, où chaque page a été envisagée dans une perspective responsive, anticipant ainsi les ajustements nécessaires pour garantir une expérience utilisateur optimale et également un gain de temps considérable lors de la réalisation de l’application.

SCREEN : Screen figma vues desktop / mobile / tablette

Plusieurs techniques ont été pensées concernant le responsive design :

* Display grid et display flex

La mise en œuvre de cette approche repose largement sur l'utilisation judicieuse de propriétés CSS telles que "**display flex"** et "**display grid**". Ces techniques jouent un rôle central dans la structuration dynamique des pages, permettant une adaptation rapide de la disposition en fonction du type d'appareil, de son orientation et de son format.

"display flex" et "display grid" sont des propriétés CSS qui facilitent la création de mises en page flexibles et basées sur une grille, respectivement. "Display flex" est idéal pour les modèles de mise en page unidimensionnels, tandis que "display grid" offre une approche bidimensionnelle, permettant de définir des rangées et des colonnes pour une disposition plus complexe.

SCREEN : Screen pages avec display grid et display flex + code + changement sur mobile

* Utilisation mesure relative

-utilisation de mesure relative (rem) à la place des px, permet de s’adapter plus facile

SCREEN : Screen code avec rem

* Break point / medias queries

-partie sur les breaks point / point de rupture

SCREEN : Screen code avec breakpoint

Un "media query" est une fonctionnalité en CSS qui permet **de définir des points de rupture pour appliquer des styles spécifiques en fonction de certaines conditions.** Ces conditions sont généralement basées sur des caractéristiques de l'appareil, telles que la largeur de l'écran, l'orientation de l'écran, la résolution, etc. Les media queries sont essentielles pour la mise en oeuvre du **responsive design**, car elles permettent d'adapter la mise en page en fonction des dispositifs utilisés.

* Gestion police adaptive ?
* Menu burger

### b. Optimisation de l’Expérience Utilisateur (UX)

## 03. Templating

### a. --

Templating (twig) – repetition de structure, bouton etc économiser du code, factoriser, meilleur lecture, meilleur maintenalibité évolutivité, cohérence graphique,

**Templating avec Twig (Symfony)**

Mentionnez comment Twig, le moteur de template de Symfony, est utilisé pour organiser la logique de présentation dans votre application, en séparant le code PHP et le code HTML.

Le templating est une technique de développement web qui vise **à séparer la structure de présentation d'une page web**. Il offre la possibilité de réutiliser des modèles prédéfinis pour générer dynamiquement des pages avec du contenu provenant de différentes sources de données. Le développement et la maintenance s’en voient plus flexibles et efficaces, permettant alors **une meilleure séparation des préoccupations et une réutilisation du code.**

UX : UI ?

**1) Quelle est la différence entre UX Design et UI Design ?**

**UX : User experience Design :** se concentre sur **l'expérience globale de l'utilisateur** lorsqu'il interagit avec une interface, en cherchant à la rendre intuitive, fluide et agréable. Il vise à comprendre les besoins et les attentes de l'utilisateur et à les traduire en une expérience positive.

**UI : User interface Design :** concerne principalement **la conception visuelle et fonctionnelle de l'interface.** Il s'agit de la conception graphique, de l'agencement des éléments visuels, de la typographie, des couleurs, et de l'interactivité avec ces éléments. L'UI Design travaille sur la manière dont l'utilisateur interagit visuellement avec l'interface.

La hiérarchie visuelle en UI Design concerne la manière dont les éléments **visuels sont organisés pour guider l'attention de l'utilisateur.** Elle implique l'utilisation de différentes tailles, couleurs, typographies et positions **pour mettre en avant certains éléments** et rendre l'interface plus compréhensible et intuitive.

L'affordance en UX Design est la **capacité d'un élément visuel ou/et interactif à suggérer son utilisation ou sa fonction à l'utilisateur.** Par exemple, un bouton avec une style en

relief suggère qu'il peut être cliqué offrant de ce fait une **affordance pour l’action de cliquer**. Le but étant de créer des indices ou **signaux visuels** qui vont inviter **intuitivement**

Accessibilité ??

8. Intégration avec l'Identité Graphique

Transition vers la partie suivante (Identité Graphique) en expliquant comment les maquettes s'alignent avec les choix visuels et l'identité de l'application.

9. Conclusion du Maquettage

Récapitulation rapide des principaux points abordés dans la section.

Souligner l'impact positif du maquettage sur le développement ultérieur de l'application.

Premier blabla sur l’identité visuelle, enchaîner avec la partie identité visuelle

# VIII. Identité visuelle / direction artistique

## 01. Catégorie

### a. Sous catégorie

### b. Sous catégorie

## 02. Catégorie

### a. Sous catégorie

### b. Sous catégorie

# IX. RGPD

La protection des données personnelles est devenue une préoccupation majeure, non seulement pour les utilisateurs mais aussi pour les entreprises et les développeurs web. Intégrer les principes du **Règlement Général sur la Protection des Données (RGPD)** dans mon projet revêt une importance cruciale. Non seulement cela s'aligne sur les normes éthiques et légales mises en place au sein de l’Union Européenne, mais c'est aussi essentiel pour bâtir une relation de confiance avec les utilisateurs. Le RGPD offre un cadre solide pour garantir que les données personnelles sont traitées de manière transparente, sécurisée, et conforme aux attentes des utilisateurs en matière de vie privée.

## RGDP & CNIL

Le Règlement Général sur la Protection des Données (RGPD) est **une législation de l'Union Européenne**, entrée en application le 25 mai 2018, visant à renforcer la protection des données personnelles des individus. Est considéré comme données personnelles toutes données contenant des informations permettant d’identifier directement (comme le nom) ou indirectement (comme le numéro de téléphone) une personne physique.

Il impose des obligations strictes aux organisations qui collectent, traitent, et stockent des données personnelles, et confère des droits étendus aux utilisateurs sur leurs propres données**.**

**La Commission Nationale de l'Informatique et des Libertés (CNIL)** en France est l'organisme chargé de veiller à la conformité des organisations avec le RGPD et de garantir le respect des droits des individus. Elle met également à disposition une documentation fournie et complète accessible directement sur son site Internet afin de mieux comprendre et appliquer ce règlement[[1]](#footnote-1).

En respectant les normes du RGPD et en collaborant avec la CNIL, notre projet garantit une expérience utilisateur responsable, respectueuse de la vie privée, et en conformité avec les dernières avancées en matière de protection des données.

## 02. Politique de Confidentialité et Consentement

### a. Politique de Confidentialité

- Fournir une politique de confidentialité complète et facilement accessible. - Clarifier les types de données personnelles collectées et les finalités de leur traitement.

La politique de confidentialité constitue le fondement de la transparence dans le traitement des données personnelles. Notre engagement envers les utilisateurs repose sur les principes suivants :

* **Accessibilité et Exhaustivité :**
  + Fournir une politique de confidentialité complète et facilement accessible, disponible sur toutes les pages du site.
  + Clarifier de manière exhaustive les types de données personnelles collectées, les finalités du traitement, et la durée de conservation de ces données.
* **Navigation Intuitive :**
  + Faciliter la navigation en proposant des liens directs vers la politique de confidentialité sur la page d'inscription, à côté des cases à cocher pour le consentement.
  + Inclure des liens dans le footer pour permettre un accès rapide depuis n'importe quelle page du site.

### b. Consentement de L’Utilisateur

Le consentement des utilisateurs est un élément clé de notre approche respectueuse de la vie privée. Nous nous engageons à garantir un processus de consentement transparent et conforme aux exigences du RGPD :

* **Mécanismes Clairs :**
  + Expliquer de manière claire et compréhensible comment les utilisateurs peuvent donner leur consentement.
  + Mettre en place des mécanismes visuels, tels que des cases à cocher spécifiques, pour obtenir le consentement de manière explicite.
* **Accessibilité et Information :**
  + Offrir des liens directs vers la politique de confidentialité et d'autres documents pertinents à côté des cases à cocher pour que les utilisateurs puissent accéder aux informations détaillées.
  + Assurer que ces liens sont également disponibles dans le footer du site.
* **Consentement Éclairé :**
  + Respecter le principe du consentement éclairé, en fournissant des informations suffisantes pour que l'utilisateur comprenne clairement les implications de son consentement.
  + Permettre le retrait du consentement à tout moment, conformément aux exigences du RGPD.

En adoptant ces pratiques, notre projet assure une expérience utilisateur transparente, respectueuse de la vie privée, et pleinement en accord avec les normes du RGPD.

- Expliquer clairement comment les utilisateurs peuvent donner leur consentement. - Mettre en place des mécanismes clairs pour obtenir le consentement, par exemple, des cases à cocher spécifiques.

Case à coché, non coché par défaut car illégal

Liens directement sur la page d’inscription à côté de la checkbox pour accéder aux différents textes et aussi dispo dans le footer (politique de confidentialité, CGU, CGV, autres ?)

Consentement éclairé ? Le consentement valide selon le RGPD doit être donné de manière claire, explicite et libre par l'individu concerné. Il doit être informé et révocable à tout moment.

**2. Consentement Lors de l'inscription,** un consentement explicite est recueilli où l'utilisateur est pleinement informé de ce à quoi il consent et où il conserve son libre arbitre. Le consentement est requis de manière explicite, c'est-à-dire que la case n'est pas pré-cochée par défaut. De plus, un lien direct cliquable redirige vers les politiques de confidentialité et les conditions d'utilisation pour une transparence totale. Ces documents sont également accessibles dans le footer du forum.

## 02. Droits des Utilisateurs

### a. Droit à l’Information

- Informer les utilisateurs de manière transparente sur leurs droits en matière de protection des données.

### b. Droit d’Accès et de Rectification

- Fournir aux utilisateurs un moyen d'accéder à leurs données personnelles et de les corriger si nécessaire.

### c. Droit à l’oubli et droit à l’effacement

- Expliquer comment les utilisateurs peuvent demander la suppression de leurs données.

Anonymiser différent de pseudoniser

## 02. Collecte et Traitement des Données

### a. Principe de minimisation

### a. Finalités de Collecte / principe de finalité

- Indiquer clairement les raisons pour lesquelles les données personnelles sont collectées.

Le responsable d’un fichier ne peut enregistrer et utiliser des informations personnelles que dans un but précis, légal et légitime. Ainsi, il est essentiel de préciser clairement les finalités pour lesquelles les données personnelles sont collectées.

Pour répondre à ce principe, je mets à disposition une **Politique de confidentialité** facilement accessible sur le site et qui est nécessaire d’accepter lors de l’inscription.

### Le principe de proportionnalité et de pertinence

Les informations enregistrées doivent être pertinentes et strictement nécessaires au regard de la finalité du fichier. Lors de la collecte de données, je ne demande que les informations strictement nécessaires pour atteindre la finalité prévue. Par exemple, pour s'inscrire sur mon application, j’ai besoin de l’adresse mail de l’utilisateur de façon à vérifier son compte. L’utilisateur aura aussi besoin d’un pseudonyme qui permettra de l’identifier sur l'application. Par ailleurs, je n’ai pas besoin de sa date de naissance ni de **63**

son nom car ces informations ne sont pas nécessaires au bon fonctionnement de l’application.

### a. Le principe de sécurité et de confidentialité

Le responsable du fichier doit garantir la sécurité des informations qu'il détient. Il doit en particulier veiller à ce que seules les personnes autorisées aient accès à ces informations.

Pour répondre à ce principe, j'ai mis en place des mesures de sécurité pour protéger les données personnelles collectées.

Cela inclut le chiffrement du mot de passe, mais aussi une restriction stricte des accès aux données, comme nous avons pu le détailler dans le chapitre dédié à la sécurité.

### b. Durée de Conversation / Le principe d’une durée de conservation limitée

- Spécifier la période pendant laquelle les données seront conservées et la logique sous-jacente.

l n'est pas possible de conserver des informations sur des personnes physiques dans un fichier pour une durée indéfinie. Une durée de conservation précise doit être fixée, en fonction du type d'information enregistrée et de la finalité du fichier. Ainsi, dans la politique de confidentialité disponible sur mon site, je précise la durée pendant laquelle je conserverai les données collectées.

De plus, j’ai mis en place un suivi de la dernière activité de l’utilisateur. Ainsi, l’entité User possède un attribut lastActivityAt qui est mis à jour à 2 moments :

### c. Traitement des Données Sensibles

- Expliquer comment les données sensibles seront traitées et protégées.

Transition :

1. Données supplémentaires : pour passer un achat
2. Pour avoir une page artiste : aka entreprise
3. Anonymisation
4. Cookie
5. Tarte au citron

# X. Technologies / structure technique

Dans le cadre ce projet, j'ai sélectionné un ensemble de technologies front-end robustes pour offrir une expérience utilisateur riche, dynamique et intuitive. Ces technologies, soigneusement intégrées, contribuent à l'esthétique, à la convivialité et à la performance globale de mon application.

## Front-End

### HTML5 et Css3

**HTML5 (HyperText Markup Language)** élément fondamental de la structure de toutes les pages web. HTML5 fournit une syntaxe plus propre et plus sémantique, permettant une organisation logique des contenus.

**CSS3 (Cascading Style Sheets 3)** responsable de la présentation et de la mise en page, CSS3 offre une panoplie de fonctionnalités avancées. Les sélecteurs avancés, les transitions, les animations, et les effets visuels permettent une personnalisation approfondie de l'apparence de l'application. La compatibilité avec les designs responsifs est également optimisée, garantissant une expérience utilisateur homogène sur divers appareils.

### Sass

Sass (Syntactically Awesome Stylesheets) est un préprocesseur CSS puissant et étendu pour une Gestion Avancée des Styles. En tant que tel, Sass va au-delà des fonctionnalités traditionnelles de CSS en offrant une syntaxe étendue et des fonctionnalités de programmation qui améliorent significativement la gestion des styles dans le développement front-end.

Une extension de CSS, Sass offre des fonctionnalités supplémentaires telles que les variables, les boucles, et les fonctions, facilitant la gestion et la maintenance du code CSS. La modularité de Sass permet une organisation claire et une réutilisation efficace des styles, ce qui se traduit par une feuille de style plus maintenable et évolutive.

### c. JavaScript, Jquery et Ajax

## 02. Back-end

### a. Php & mySql

## 03. Librairie

### a. Jquery

### a. React

## 04. API

### a. Stripe

### b. Leatlet & Denominati

## 05. Autre

### a. Gestion de projet et développement

Trello, looping, vscode, laragon, gihub

### Graphisme

figma et photoshop

1. Front-End
2. **JavaScript :**
   * Langage de programmation incontournable pour le développement front-end, JavaScript confère une interactivité dynamique à mon application. Grâce à ses fonctionnalités asynchrones, il permet des mises à jour en temps réel des contenus sans nécessiter le rechargement de la page. Les bibliothèques et frameworks JavaScript modernes simplifient le développement, tandis que les APIs facilitent l'intégration de fonctionnalités avancées.
3. **jQuery :**
   * Une bibliothèque JavaScript légère mais puissante, jQuery simplifie la manipulation du DOM (Document Object Model) et l'interaction avec les événements. Son approche simplifiée du code facilite le développement rapide, améliorant ainsi l'efficacité du projet. Bien que d'autres technologies aient gagné en popularité, jQuery reste une option viable pour des projets spécifiques.
4. **AJAX (Asynchronous JavaScript and XML) :**
   * La technologie AJAX permet des échanges de données asynchrones entre le navigateur et le serveur, améliorant la réactivité de l'application en évitant les rechargements complets de la page. Cela se traduit par une expérience utilisateur plus fluide, avec des mises à jour en temps réel et des chargements de contenu sans interruption.
5. **Sass :**

Préprocesseur CSS :

Sass fonctionne comme un préprocesseur CSS, ce qui signifie qu'il introduit une couche d'abstraction entre le code que j'écris et le fichier CSS final généré. Cette approche présente plusieurs avantages, notamment la possibilité d'utiliser des fonctionnalités qui ne sont pas présentes nativement dans CSS, comme les variables, les boucles, les conditions, et les fonctions.

Variables :

L'utilisation de variables en Sass permet de définir des valeurs réutilisables dans tout le code. Par exemple, je peux déclarer une couleur primaire sous forme de variable et l'utiliser dans tout le projet. Cela facilite la maintenance du code, car si la couleur doit être modifiée, je n'ai qu'à le faire à un seul endroit.

SCREEN

Nesting :

Sass prend en charge le nesting, ce qui signifie que je peux imbriquer des sélecteurs pour refléter la hiérarchie HTML. Cela améliore la lisibilité du code en rendant la structure HTML et la structure CSS plus alignées.

SCREEN

En conclusion, l'intégration de Sass dans mon projet améliore considérablement la maintenabilité, la lisibilité et la réutilisabilité de mon code CSS. offrant une gestion des styles plus efficiente et une flexibilité accrue pour répondre aux exigences spécifiques de mon application web et mobile.

En somme, l'utilisation concertée de HTML5, CSS3, JavaScript, jQuery, AJAX et Sass dans mon projet vise à créer une application web et mobile performante, esthétiquement plaisante, et hautement interactive. Ces technologies sont non seulement alignées sur les normes modernes du développement web, mais elles apportent également une valeur ajoutée significative à l'expérience utilisateur.

# XI. Framework : Symfony

Un **framework** est une infrastructure logicielle qui fournit une base structurée pour la création d'applications. Il offre un ensemble d'outils, de bibliothèques et de conventions préétablies pour simplifier le processus de développement, favoriser la cohérence du code, et accélérer la réalisation de projets complexes.

Afin de garantir l'efficacité et la robustesse de l'architecture globale de mon projet, j'ai fait le choix de recourir au framework Symfony. Celui-ci s'est imposé comme une solution incontournable dans le monde professionnel du développement web.

Symfony a été sélectionné pour optimiser le processus de développement de l'application, offrant une approche structurée et des outils puissants qui permettent de gagner du temps tout en garantissant la qualité du code. De plus, son adoption est motivée par la volonté de se conformer aux normes et exigences professionnelles du secteur du développement web. En exploitant les fonctionnalités étendues de Symfony, j'ai pu bénéficier d'un large éventail de ressources et de bundles, enrichissant ainsi mon projet à divers niveaux. Ces choix stratégiques visent à assurer une base solide et évolutive pour le développement de l'application artistique, tout en facilitant la maintenance future et l'intégration de nouvelles fonctionnalités.

## Fonctionnement

Symfony est une Framework open source français basé sur **le langage de programmation PHP** et repose sur le design pattern **MVC** (Modèle-Vue-Contrôleur).

>> blabla concernant le fonctionnement global

SCREEN : Schéma global Symfony

Symfony propose également de nombreux avantages permettant d’optimiser le processus de développement :

* **Architecture MVC (Modèle-Vue-Contrôleur) :** Symfony adopte l'architecture Modèle-Vue-Contrôleur, favorisant une organisation claire du code, facilitant la maintenance et l'évolutivité.
* **Sécurité :** Intégration de composants de sécurité robustes, protégeant contre les attaques, gérant les sessions et les autorisations pour des applications sécurisées.
* **Doctrine ORM :** Utilisation de Doctrine, un ORM puissant simplifiant la gestion des bases de données via des objets PHP.
* **Conventions de codage et de standards :** Encourage l'adoption de conventions, assurant la cohérence du code au sein des équipes de développement.
* **Symfony Components :** Vaste bibliothèque de composants réutilisables pour une modularité et une réutilisation du code.
* **Console Symfony :** Interface en ligne de commande (CLI) offrant des outils puissants pour diverses tâches de développement.
* **Documentation exhaustive :** Dispose d'une documentation complète et bien entretenue et d’une communauté active, fournissant un support continu.
* **Flexibilité et extensibilité :** Permet une personnalisation approfondie avec une vaste bibliothèque de bundles de la communauté.
* **Performance :** Constantes optimisations pour des performances élevées même dans des applications complexes.

Symfony : stabilité, travail en équipe, robustesse, adéquation du monde du travail, optimisation, rapidité, versatilité etc.

Symfony propose plusieurs technologie majeures telles que :

Lien avec le projet : permet d’avoir une structure claire et lisible, permet évolution facilement.

Security

**10) Quel est le composant qui garantit l’authentification et l’autorisation des utilisateurs ?**

Le composant qui garantit l'authentification et l'autorisation des utilisateurs dans Symfony est **le composant de sécurité (Security Component).** Ce composant fournit des fonctionnalités pour la gestion de l'authentification (par exemple, formulaire de connexion, authentification basée sur un jeton, authentification à l'aide de services tiers, etc.), la gestion des rôles et permissions, et d'autres aspects liés à la sécurité dans une application web Symfony.

Définition du routing dans symfony

Grandement extensible, il est utilisé pour développer des applications web et des API, il offre une large gamme de fonctionnalités prêtes à l’emploi (comme la gestion des routes, la gestion des formulaires, la sécurité, l’accès aux bases de données etc) facilitant ainsi significativement le processus de développement.

### a. Doctrine

L'utilisation de la programmation orientée objet avec une base de données relationnelle implique la conversion des données relationnelles en objets et réciproquement. Symfony utilise la bibliothèque Doctrine comme ORM (Object Relational Mapping) pour simplifier cette interaction.

L'ORM agit comme une interface entre le programme applicatif et la base de données relationnelle, simulant une base de données orientée objet. Il permet de faire le lien entre les objets de la programmation orientée objet et les tables d’une base de données relationnelle. Il facilite le mapping, la persistance et la récupération des données, créant ainsi une couche d'abstraction entre le monde objet et le monde relationnel.

Doctrine, l'ORM de Symfony, offre une interface intuitive permettant d'interagir avec la base de données en utilisant des entités PHP à la place de requêtes SQL directes. Cette approche permet de manipuler les objets Symfony via des requêtes DQL (Doctrine Query Language), rendant les opérations de lecture, d'écriture et de modification de données plus naturelles et orientées objet. Doctrine automatise la création des requêtes SQL, facilite la gestion des schémas de base de données, et propose des fonctionnalités avancées telles que la gestion des relations entre entités et la validation des données. Ainsi, il simplifie le travail sur les données côté base de données en évitant l'utilisation directe de requêtes SQL.

**11. Qu’est-ce que le « mapping des objets » ?**

Dans le contexte du développement web et des bases de données relationnelles, le "mapping des objets" fait référence à la manière dont les objets d'une application orientée objet sont associés aux enregistrements dans une base de données relationnelle. Cela implique de définir la correspondance entre les propriétés et les méthodes d'un objet et les colonnes et lignes d'une table dans la base de données.

L'objectif principal du mapping des objets est de permettre une interaction transparente entre le code de l'application (qui utilise des objets) et la base de données (qui stocke les données de manière relationnelle). Un ORM (Object-Relational Mapping) est souvent utilisé pour faciliter cette correspondance en automatisant la conversion entre les objets de l'application et les enregistrements de la base de données.

**9) Quel est le langage de requêtage exploité au sein d’un projet Symfony ?**

Au sein d'un projet Symfony, le langage de requêtage exploité est **le DQL (Doctrine Query Language) qui est une variante orientée objet du langage SQL.** Il est utilisé pour interagir avec la base de données via Doctrine. Il offre une syntaxe plus orientée objet et abstraite par rapport au SQL pur, **facilitant de ce fait le mapping des objets** avec les tables de la base de données.

### a. Moteur de Template Twig

Symfony utilise Twig comme moteur de template, un outil moderne, puissant et flexible qui simplifie la gestion des vues dans les applications Symfony. Twig permet de séparer la logique de présentation de la logique métier, offrant une syntaxe intuitive, simplifiée et concise par rapport au PHP natif. Cette syntaxe facilite l'intégration de boucles, variables, conditions, et d'autres fonctionnalités dans les templates, comme pour les formulaires.

Comparé au PHP natif :

PHP natif : <?php echo $variable ?>

Twig : {{ variable }}

Twig offre des fonctionnalités avancées qui améliorent l'efficacité, l'intuitivité et la sécurité du processus de création et de rendu des vues. Cela permet de créer des vues dynamiques tout en rendant les templates plus lisibles et modulaires, facilitant ainsi la maintenance du code.

SCREEN : screen code page.html.twig

Ex : condition, boucle, variable, formulaire, |DateTime (dire qu’il y a qqchose de natif dans twig mais comme c’est qqchose que j’ai bcp utilisé j’ai crée DateTime personnalisé)

### a. Bundles

Symfony offre un vaste éventail de bundles, ce qui constitue un avantage majeur permettant d'intégrer facilement des fonctionnalités supplémentaires. Un bundle dans Symfony est un ensemble de fichiers ou composants, agissant comme un plugin ou module, offrant une ou plusieurs fonctionnalités de manière totalement réutilisable et autonome.

Parmi les bundles majeurs, le bundle Maker se démarque. Cet outil simplifie la création d'éléments fréquemment utilisés tels que des contrôleurs, des entités, des formulaires, etc. Il génère rapidement du code boilerplate (code de base), accélérant ainsi le processus de développement de manière efficace et optimisée.

Commands ?

### a. Composer

Composer, largement adopté dans le développement Symfony, est l'outil utilisé pour la gestion des dépendances des projets PHP. L'injection de dépendances est un principe de conception consistant à fournir à une classe toutes les dépendances dont elle a besoin au lieu de les créer à l'intérieur de la classe.

Composer permet de spécifier les bibliothèques tierces (packages) requises, assurant automatiquement leur téléchargement et leur installation. Composer simplifie la gestion des dépendances, garantissant la présence et la mise à jour des bibliothèques nécessaires.

Les dépendances du projet sont répertoriées dans le fichier "composer.json", qui permet leur installation, mise à jour et gestion des versions.

### a. Service

## 02. Design pattern

dans son cas, à la différence qu’il adopte une variante de ce design pattern à savoir : il adopte également une variante le MVP (modèle vue présentation).

Symfony est un Framework PHP basé sur le design pattern « MVP » (pour « Model Vue Presenter ») qui ressemble au design Pattern « MVC » à la seule exception que les Controllers sont remplacés par des Presenters et que les vues ne communiquent pas directement avec les modèles.

MVC -avantages + explication

* Séparation claire des responsabilités
* Facilité de maintenance et l’évolutivité
* Réutilisation et optimisation d code
* Flexbilité dans le développement
* Compréhension & collaboration facilitées

Modèle vue controller (schéma)

Ce sont des guides de conception qui aident à structurer et organiser le code selon des pattern de conception.

Modèle vue contrôleur. Elle sépare l’application en 3 parties principales : modèle, vue, contrôleur chacun ayant son rôle spécifique à jouer.

**Modèle (Model) :** Le modèle représente les données et la logique métier de votre application. Il gère la récupération et la manipulation des données, ainsi que les règles de l'application. Par exemple, pour une fonctionnalité majeure, le modèle pourrait gérer la collecte des messages dans un forum.

**Vue (View) :** La vue gère l'interface utilisateur. Elle est responsable de l'affichage des données du modèle et de la présentation visuelle. Dans le contexte d'une fonctionnalité majeure, la vue pourrait afficher les messages du forum de manière conviviale pour l'utilisateur.

**Contrôleur (Controller) :** Le contrôleur agit comme un intermédiaire entre le modèle et la vue. Il gère les interactions de l'utilisateur, traite les demandes et met à jour le modèle ou la vue en conséquence. Par exemple, pour une fonctionnalité de forum, le contrôleur gère la création, la modification et la suppression de messages.

Un design pattern, ou motif / patron de conception, est une **solution réutilisable à un problème couramment rencontré pour la conception de logiciel.** Il offre des directives et des bonnes pratiques éprouvée qui aident **à structurer et organiser le code de manière efficace et maintenable.**

**Modèle vue contrôleur** – est un **pattern de conception utilisé pour organiser la structure d’une application logicielle.** Elle divise l’application en trois composants distincts : Model, View et Controller, chacun ayant son rôle spécifique à jouer. Cela permet de maintenir la séparation entre la logique métier et l’interface utilisateur.

**Modèle :** représente les données de la **logique métier** de l’application. Il gère la **manipulation, la récupération et la mise à jour des données.**

**Vue :** représente **l’interface utilisateur**. Elle **affiche les données au format approprié** et permet à l’utilisateur d’interagir avec l’application.

**Contrôleur :** représente un **intermédiaire entre modèle et vue**, il gère **la logique de traitement des requêtes de l’utilisateur.** Il reçoit les actions de l’utilisateur depuis la Vue, interagit avec le Model pour récupérer les données souhaitées, et met à jour la Vue en conséquence.

**5) Quels sont les avantages de l’architecture MVC ?**

Les avantages sont les suivants : **séparation claires des tâches, réutilisation du code, testabilité, maintenabilité et flexibilité, évolutivité et collaboration efficace.**

**6) Existe-t-il des variantes à l’architecture MVC ?**

**MVP – Modèle-Vue-Présentation :** Le Présentateur (ou la présentation Model) prend en charge le rôle

### a.

### a.

## 03. Routing

### a.

### a.

## 04. Au sein du projet

### a. fonctionnalité principale

### a. services ? bundle? Command ? securité login / identification / isverified mail qqchose / rôle / token – sécurité , form

Symfony intègre des composants de sécurité avancés, tels que le filtrage des entrées, la protection CSRF (Cross-Site Request Forgery), et la gestion des sessions, la gestion des rôles et des autorisations, ainsi que des outils pour prévenir les vulnérabilités courantes contribuant à la création d'applications robustes et sécurisées.

* **Console Symfony :** L'interface en ligne de commande (CLI) de Symfony fournit des outils puissants pour la gestion des tâches courantes, telles que la génération de code ou encore les migrations de base de données
* Symfony est optimisé pour les performances, avec des composants tels que l'injection de dépendances, le cache, et l'optimisation des requêtes SQL, ce qui contribue à des applications rapides et réactives.

Il facilite notamment la mise en place de certaines fonctions de base dont j’avais besoin telles que l’authentification et la connexion.

**Gestion avancée des formulaires :** Symfony propose un composant de gestion des formulaires puissant, facilitant la création, la validation et la manipulation des données des formulaires.

**10) Quel est le composant qui garantit l’authentification et l’autorisation des utilisateurs ?**

Le composant qui garantit l'authentification et l'autorisation des utilisateurs dans Symfony est **le composant de sécurité (Security Component).** Ce composant fournit des fonctionnalités pour la gestion de l'authentification (par exemple, formulaire de connexion, authentification basée sur un jeton, authentification à l'aide de services tiers, etc.), la gestion des rôles et permissions, et d'autres aspects liés à la sécurité dans une application web Symfony.

design pattern (mvc)

doctrine

dql (exemple requête personnalisée)

exemple de parcours mvp

POO ?

routing ?

Transition :.

### Présentation

. Lorsque vous voyez sur les schémas de Symfony la mention "Noyau de Symfony" ou "Symfony Kernel" au centre, cela fait référence au Kernel de Symfony. C'est le composant central qui orchestre le processus de gestion des requêtes HTTP, la configuration de l'application, la gestion des bundles, et d'autres aspects cruciaux du cycle de vie de l'application Symfony.

Dans le contexte de Symfony, le terme "Kernel" se réfère au **Symfony Kernel**, qui est le cœur de l'application Symfony. Le Kernel est responsable de la gestion du cycle de vie de l'application et de la gestion des événements qui se produisent tout au long du processus de traitement des requêtes.

Voici quelques points clés concernant le Kernel dans Symfony :

1. **Entrée Unique :** Le Kernel fournit un point d'entrée unique pour toutes les requêtes HTTP entrantes. Il initialise l'application, gère les requêtes et renvoie les réponses.
2. **Gestion des Bundles :** Symfony utilise le concept de "bundles" pour organiser et étendre les fonctionnalités de l'application. Le Kernel charge ces bundles et les configure en fonction de la configuration de l'application.
3. **Événements :** Le Kernel gère un système d'événements qui permet aux développeurs de réagir à différentes étapes du cycle de vie de l'application. Cela offre une flexibilité considérable pour personnaliser le comportement de l'application.
4. **Environnements :** Le Kernel prend en charge différents environnements tels que "dev" (développement), "prod" (production), etc. Cela permet de configurer l'application de manière différente en fonction du contexte, par exemple, pour activer le débogage en environnement de développement.
5. **Gestion des Configurations :** Le Kernel est responsable de la gestion de la configuration de l'application, notamment la configuration des services, des routes, des paramètres, etc.
6. **Traitement des Requêtes :** Lorsqu'une requête HTTP entre dans l'application Symfony, le Kernel orchestre le processus de traitement de la requête, appelant les différentes parties de l'application nécessaires pour générer la réponse.

En résumé, le Symfony Kernel est un élément central dans l'architecture Symfony, fournissant une structure et un mécanisme de gestion pour le développement d'applications web robustes et extensibles.

### c. Symfony eu sein du projet

#### a.Démonstration d’une fonctionnalité – mvc

Afficher les events

#### b.Exemple d’une requête personnalisée

# XII. SEO – Search Engine Optimisation

Dans le développement web, **l'optimisation pour les moteurs de recherche (SEO)** et donc le **référencement naturel** est bien plus qu'une simple pratique technique : c'est une stratégie essentielle qui façonne la visibilité en ligne d'un site, dictée par la prédominance des moteurs de recherche, en particulier Google, dans le paysage numérique. Le SEO englobe **un ensemble de techniques, de bonnes pratiques et d'ajustements[[2]](#footnote-2)** tant au niveau du front-end que du back-end, visant **à améliorer le classement d'un site web** dans les résultats des moteurs de recherche et à **accroître sa visibilité** auprès du public cible.

Dans le monde numérique concurrentiel d'aujourd'hui, la position d'un site dans les résultats de recherche peut faire toute la différence en termes de succès et d'impact. En introduisant des pratiques de SEO dans notre projet, nous visons à garantir que notre site web soit non seulement techniquement performant, mais qu'il soit également positionné de manière optimale pour attirer et retenir un public plus large. En comprenant les principes du SEO, nous cherchons **à maximiser la visibilité de notre contenu, à offrir une expérience utilisateur optimale et à établir une présence en ligne pérenne.**

## 01. Introduction au SEO et Référencement

### a. Définition du SEO

Le SEO, ou Search Engine Optimization, représente une composante essentielle de tout projet de développement web axé sur la création d'une présence en ligne robuste et facilement accessible. Il s'agit d'un ensemble de techniques et de bonnes pratiques visant à optimiser un site web afin d'obtenir un meilleur classement dans les résultats des moteurs de recherche. Dans le cadre de ce projet, l'intégration d'une stratégie SEO est une démarche stratégique visant à accroître la visibilité du site sur Internet.

### b. Objectifs du référencement

Les objectifs du référencement sont multiples et convergent vers l'amélioration de la visibilité et de la pertinence du site web. Tout d'abord, le référencement vise à améliorer le classement global du site dans les moteurs de recherche, permettant ainsi de figurer parmi les premiers résultats lorsqu'un utilisateur effectue une recherche pertinente. Cette position privilégiée accroît considérablement la probabilité que les utilisateurs cliquent sur le lien, générant ainsi un trafic organique.

De plus, le référencement cherche à augmenter la visibilité globale du site, garantissant qu'il soit découvert par un public plus large. Cela se traduit non seulement par un meilleur positionnement dans les résultats de recherche, mais également par une meilleure compréhension des contenus par les moteurs de recherche, favorisant ainsi une indexation plus efficace.

Le référencement, au-delà d'un simple ajustement technique, est une stratégie globale. Nos objectifs s'articulent autour de deux axes principaux : améliorer le classement de notre site dans les résultats des moteurs de recherche et accroître sa visibilité auprès de notre audience cible. En comprenant les nuances du SEO, nous aspirons à créer une présence en ligne robuste, à favoriser l'accessibilité de notre contenu, et à garantir que notre site soit découvert par ceux qui recherchent activement les produits, services ou informations que nous offrons. Les sections à venir exploreront en détail les pratiques et les techniques spécifiques que nous mettrons en œuvre pour atteindre ces objectifs, en plaçant le SEO au cœur de notre approche de développement web.

Face à la prédominance des moteurs de recherche, nos objectifs de référencement sont des impératifs stratégiques. Il s'agit non seulement d'améliorer le classement de notre site dans les résultats de recherche, mais surtout de garantir sa visibilité, car c'est là que réside l'opportunité de toucher un public plus vaste. Dans un environnement où la recherche en ligne est la première étape de la plupart des parcours clients, le SEO devient le pilier essentiel de notre approche de développement web. Les sections à venir exploreront comment nous concrétisons ces objectifs, en respectant les règles édictées par les "gardiens" du cyberespace.

## 02. Aspects Front-End du SEO

### a. Structure HTML Sémantique

* Utilisation appropriée des balises HTML pour donner une structure sémantique au contenu.
* Importance des balises **<header>**, **<footer>**, **<nav>**, **<article>**, etc.

### b. Optimisation des Images

* Utilisation d'attributs **alt** pour décrire le contenu des images.
* Compression des images pour améliorer le temps de chargement de la page.
* Utilisation du format WebP pour des images de qualité avec des tailles de fichier plus petites.

### URLs Conviviales

Utilisation de slugs pertinents dans les URLs.

Éviter les paramètres excessifs dans les URLs dynamiques.

### d. Balisage des Données Structurées

Ajout de balises de données structurées (schema.org) pour aider les moteurs de recherche à comprendre le contenu.

### e. Responsive design

Création d'une mise en page adaptative pour garantir une expérience utilisateur optimale sur différents appareils.

## 03. Aspects Back-End du SEO

### a. Gestion des Redirects

* Mise en place de redirections appropriées (301, 302) en cas de changement d'URL.
* Éviter les redirections en chaîne.

### b. Gestion des Erreurs HTPP

* Personnalisation des pages d'erreur 404 avec des suggestions de navigation.
* Utilisation de codes d'état HTTP appropriés.

### c. Optimisation de la Vitesse du Chargement

* Compression de ressources (HTML, CSS, JavaScript).
* Mise en cache des fichiers statiques.
* Utilisation de CDN (Content Delivery Network) si nécessaire.

Utilisation de requêtes préparées et DQL optimisées pour améliorer les performances de la base de données.

Taille optimale des images et utilisation de formats de fichier légers.

### d. Propreté des Liens

* Utilisation de liens propres et significatifs pour une meilleure expérience utilisateur et une navigation fluide.
* Maîtrise de la profondeur des liens pour éviter une structure trop complexe.

### e. Web Crawling

* Mise en place d'un maillage interne et externe pour faciliter le web crawling.
* Utilisation d'une sitemap XML pour aider les moteurs de recherche à explorer le site.

### d. Sitemap XML et Robots.txt

* Création d'un sitemap XML pour aider les moteurs de recherche à explorer le site.
* Configuration d'un fichier robots.txt pour contrôler l'indexation.

## 04. Contenu de Qualité et Stratégie de Mots-clefs / Stratégies de Contenu de SEO

### a. Recherche de Mots-clefs

* Utilisation d'outils pour la recherche de mots-clés pertinents.
* Intégration naturelle de mots-clés dans le contenu.

### b. Eléments On-Page

* Optimisation des métadescriptions pour augmenter le taux de clics depuis les résultats de recherche.
* Utilisation appropriée de balises **<title>** et **<meta>**.

### b. Liens Externes et Internes

* Utilisation de liens externes vers des sources fiables et pertinents.
* Création de liens internes pour améliorer la structure du site et faciliter la navigation.

### b. Contenu unique et Engageant

* Création de contenu de qualité, informatif et engageant pour les utilisateurs.
* Éviter le contenu dupliqué.

## 05. Suivi et Analyse

### a. Intégration d’Outils d’Analyse

* Intégration de Google Analytics pour suivre les performances du site.
* Utilisation de Google Search Console pour surveiller les performances de recherche.

### b. Analyse des Données

Analyse des données pour identifier les tendances, les opportunités d'amélioration et les pages populaires.

## Conclusion

* Récapitulation des principaux points abordés en front-end et back-end.
* Souligner l'importance d'une approche holistique pour optimiser le référencement.

En traitant ces aspects, vous garantirez une approche complète du SEO, à la fois du côté du front-end, qui se concentre sur la présentation et la structure de la page, et du côté du back-end, qui gère les aspects techniques et les interactions avec les moteurs de recherche.

**Les Slugs dans les URLs : Un Guide Complet pour le SEO, la Navigation et le Confort Utilisateur**

*Qu'est-ce qu'un Slug ?*

Un slug dans une URL est une version conviviale et lisible par l'homme d'un fragment de texte, généralement le titre d'une page. Il est utilisé pour identifier de manière unique le contenu d'une page web. Par exemple, dans l'URL "https://www.mon-site-web.com/article/les-bienfaits-du-sport", le slug est "les-bienfaits-du-sport".

*Objectif du Slug :*

**SEO et Référencement :** Les slugs jouent un rôle essentiel dans l'optimisation pour les moteurs de recherche (SEO). Les moteurs de recherche accordent de l'importance aux mots-clés dans les URLs, et un slug bien choisi peut aider à améliorer le classement dans les résultats de recherche.

**Lisibilité et Compréhension :** Les slugs contribuent à la lisibilité des URLs. Un slug significatif, composé de mots clés pertinents, rend l'URL compréhensible pour les utilisateurs et les moteurs de recherche. Cela peut également favoriser un clic plus élevé dans les résultats de recherche.

**Compatibilité et Partage :** Les slugs facilitent le partage d'URL. Un slug clair et informatif permet à un utilisateur de comprendre le contenu de la page avant même de cliquer sur le lien. Cela encourage le partage sur les réseaux sociaux et améliore l'expérience utilisateur.

*Conseils pour les Slugs Optimisés :*

**Utilisation de Mots-Clés :** Intégrez des mots-clés pertinents dans vos slugs. Cependant, veillez à ce qu'ils restent naturels et informatifs.

**Éviter les Caractères Spéciaux :** Utilisez des caractères alphanumériques et des tirets. Évitez les caractères spéciaux, espaces et majuscules pour garantir une compatibilité maximale.

**Longueur Maîtrisée :** Les slugs ne devraient pas être trop longs. Ils doivent être suffisamment descriptifs, mais évitez les longues séquences de mots qui pourraient rendre l'URL difficile à lire.

**Unicité :** Chaque page doit avoir un slug unique pour éviter tout conflit ou confusion.

*Exemple de Slug vs. URL Dynamique :*

**Dynamique :** **https://www.mon-site-web.com/index.php?page=123**

**Avec Slug :** **https://www.mon-site-web.com/articles/les-bienfaits-du-sport**

*Confort Utilisateur :*

**Navigation Intuitive :** Des slugs bien conçus rendent la navigation du site plus intuitive. Les utilisateurs peuvent comprendre le contenu d'une page en examinant simplement son URL.

**Partage Facilité :** Les URLs avec des slugs significatifs sont plus faciles à mémoriser et à partager. Cela peut améliorer l'engagement des utilisateurs et favoriser le bouche-à-oreille.

En résumé, l'utilisation de slugs dans les URLs est une pratique bénéfique tant pour le SEO que pour l'expérience utilisateur. Ils contribuent à la lisibilité, à la compréhension du contenu, et à la convivialité des liens, jouant ainsi un rôle crucial dans la création d'un site web bien optimisé.

\_\_\_\_\_

**Fil d'Ariane et SEO :**

1. **Navigation Facilitée :**
   * Le fil d'Ariane offre une navigation facilitée en indiquant clairement la position de l'utilisateur dans la hiérarchie du site. Cela permet aux visiteurs de comprendre où ils se trouvent par rapport à la structure globale du site.
2. **Amélioration de l'Expérience Utilisateur :**
   * Une navigation claire et intuitive améliore l'expérience utilisateur, ce qui est un facteur pris en compte par les moteurs de recherche dans leurs algorithmes de classement.
3. **Réduction du Taux de Rebond :**
   * En aidant les utilisateurs à trouver rapidement ce qu'ils cherchent, le fil d'Ariane peut contribuer à réduire le taux de rebond. Les moteurs de recherche peuvent interpréter un faible taux de rebond comme un signe que le contenu est pertinent et satisfait les attentes des utilisateurs.
4. **Liens Internes :**
   * Les liens dans le fil d'Ariane créent des connexions internes entre les pages du site. Cela renforce la structure du site et peut aider les moteurs de recherche à explorer et indexer plus efficacement le contenu.
5. **Rich Snippets dans les Résultats de Recherche :**
   * Certains moteurs de recherche, comme Google, peuvent utiliser les informations du fil d'Ariane pour afficher des Rich Snippets dans les résultats de recherche, offrant ainsi aux utilisateurs une vue d'ensemble de la structure du site directement depuis la page de résultats.
6. **Mots-Clés dans le Fil d'Ariane :**
   * Si le fil d'Ariane intègre des mots-clés pertinents pour chaque niveau de la hiérarchie, cela peut également avoir un impact positif sur le référencement.

**Intégration dans la Structure SEO :**

* Intégrer le fil d'Ariane dans la structure de votre site d'une manière cohérente et logique.
* Utiliser des balises de schéma appropriées pour aider les moteurs de recherche à comprendre la signification du fil d'Ariane.
* Assurez-vous que les liens dans le fil d'Ariane sont conviviaux et descriptifs.

En résumé, bien que le fil d'Ariane puisse ne pas être directement classé comme une technique de SEO, son impact sur la navigation, l'expérience utilisateur, et les liens internes peut indirectement influencer positivement le référencement de votre site web.

Les balises de schéma (schema.org) sont des balises HTML spéciales utilisées pour marquer le contenu d'une page web d'une manière structurée et compréhensible par les moteurs de recherche. Elles aident les moteurs de recherche à comprendre la signification et le contexte des informations sur une page, ce qui peut améliorer la manière dont ces informations sont affichées dans les résultats de recherche et favoriser une meilleure compréhension globale du contenu.

Lorsque je mentionne l'utilisation de balises de schéma appropriées pour le fil d'Ariane, cela signifie que vous pouvez enrichir votre fil d'Ariane en ajoutant des balises de schéma pour définir clairement le rôle de chaque élément du fil d'Ariane dans la structure de la page. Cela peut inclure des balises comme **BreadcrumbList** et **ListItem** de schema.org.

<div itemscope itemtype="http://schema.org/BreadcrumbList">

<span itemprop="itemListElement" itemscope itemtype="http://schema.org/ListItem">

<a itemprop="item" href="https://www.example.com/page1">

<span itemprop="name">Accueil</span>

</a>

<meta itemprop="position" content="1" />

</span>

<span itemprop="itemListElement" itemscope itemtype="http://schema.org/ListItem">

<a itemprop="item" href="https://www.example.com/page2">

<span itemprop="name">Catégorie 1</span>

</a>

<meta itemprop="position" content="2" />

</span>

<span itemprop="itemListElement" itemscope itemtype="http://schema.org/ListItem">

<a itemprop="item" href="https://www.example.com/page3">

<span itemprop="name">Sous-catégorie 1</span>

</a>

<meta itemprop="position" content="3" />

</span>

</div>

Dans cet exemple, chaque élément du fil d'Ariane est enveloppé dans une balise **ListItem** de schema.org. Les propriétés **item** et **name** spécifient respectivement le lien et le texte du lien, et la propriété **position** indique la position de l'élément dans la liste. Cela fournit aux moteurs de recherche des informations structurées sur la séquence et la signification des éléments du fil d'Ariane.

L'utilisation de ces balises de schéma appropriées peut aider les moteurs de recherche à interpréter et à afficher correctement votre fil d'Ariane dans les résultats de recherche, offrant ainsi une expérience utilisateur améliorée et potentiellement une meilleure visibilité dans les moteurs de recherche.

# XIII. Sécurité

## 01. Gestion des failles et vulnérabilités

### a. Prévention des Attaques par Injection SQL

Les attaques par injections SQL visent à exploiter les failles des systèmes de gestion de bases de données en injectant du code SQL malveillant dans les requêtes, pouvant compromettre la base de données. L'injection SQL survient lorsqu'une application intègre directement des données non validées ou non échappées dans une requête, permettant à un attaquant d'exécuter du code malveillant sur la base de données. L'objectif principal est souvent d'accéder, modifier ou supprimer des données.

**Données non validées :** Les données non validées sont des informations entrées par les utilisateurs ou provenant de sources externes sans avoir subi une vérification adéquate de leur légitimité ou de leur conformité aux attentes.

*Par exemple, si une application web accepte un champ de formulaire pour le nom d'utilisateur et ne vérifie pas si les caractères spéciaux ou les commandes SQL sont présents, cela crée une situation où des données non validées peuvent être injectées.*

**Données non échappées :** L'échappement de données consiste à traiter les caractères spéciaux de manière à les rendre inoffensifs pour l'interpréteur SQL. Lorsque des données ne sont pas échappées correctement, les caractères spéciaux peuvent être interprétés comme faisant partie de la syntaxe SQL, permettant potentiellement l'exécution de commandes malveillantes.

*Par exemple, si un champ de formulaire accepte une chaîne de texte qui contient des guillemets simples (') non échappés, un attaquant peut introduire une chaîne telle que '; DROP TABLE user; --, ce qui pourrait conduire à une attaque de type injection SQL.*

***Comment s’en prémunir ?***

La sécurité contre les injections SQL repose sur une approche proactive de validation et d'assainissement rigoureux des données d'entrée. L'utilisation de pratiques telles que les requêtes préparées, les filtres de données, et la gestion de paramètres sécurisés joue un rôle essentiel dans la protection contre ces attaques.

Requête préparées :

Les requêtes préparées constituent une défense robuste, empêchant l'injection de code malveillant en dissociant clairement les données des requêtes SQL. Cette méthode assure que les données utilisateur ne sont pas interprétées comme du code SQL, préservant ainsi l'intégrité de notre base de données. En utilisant des requêtes préparées, notre application est résiliente aux attaques par injections SQL, notamment lors de l'exécution de requêtes de type SELECT.

Ces requêtes empêchent que les données utilisateur soient interprétées comme du code SQL, garantissant ainsi l'intégrité de notre base de données.

la validation et l'assainissement strict des données d'entrée. et de les échapper correctement avant de les utiliser dans des requêtes SQL

**Validation des Données :**

* + **Définition :** La validation des données implique la vérification de la conformité des informations fournies par l'utilisateur avec des critères prédéfinis. Elle s'assure que les données respectent les règles et les attentes définies par l'application.

**Assainissement des Données :**

* + **Définition :** L'assainissement des données consiste à nettoyer et à modifier les entrées utilisateur de manière à éliminer tout élément malveillant ou indésirable. Cela peut inclure la suppression de caractères spéciaux ou la transformation de données pour les rendre inoffensives.

**Filtrage des Données :**

* + **Définition :** Le filtrage des données consiste à appliquer des filtres ou des règles sur les informations pour en permettre le passage ou le rejet en fonction de certains critères. Les filtres peuvent être basés sur le type de données, des motifs spécifiques, ou d'autres conditions. contribue à garantir que les informations entrantes respectent les attentes définies par l'application, évitant ainsi toute manipulation malveillante.

**Gestion des Paramètres Sécurisés :**

* + **Définition :** La gestion des paramètres sécurisés concerne la manière dont les valeurs des paramètres, tels que ceux utilisés dans les requêtes SQL, sont manipulées pour éviter les vulnérabilités. Cela peut inclure l'utilisation de requêtes préparées, où les paramètres sont traités de manière à prévenir les attaques d'injection. Elles offrent une solution sécurisée en dissociant clairement le code SQL des valeurs des paramètres, empêchant ainsi les attaques d'injection SQL.

En résumé, la validation s'assure que les données sont conformes aux attentes, l'assainissement élimine les éléments malveillants, le filtrage contrôle le passage des données conformes, et la gestion des paramètres sécurisés prévient les attaques d'injection en manipulant les valeurs des paramètres de manière sécurisée. Ces pratiques combinées renforcent la sécurité globale de l'application contre les vulnérabilités, notamment les injections SQL.

**Comment cela est géré par Symfony :**

* Symfony offre des outils intégrés pour contrer les injections SQL.
* **Filtre de Données :** Symfony permet de filtrer et valider les données d'entrée, garantissant qu'elles respectent les attentes du système.
* **Requêtes Préparées :** Symfony encourage l'utilisation de requêtes préparées, ce qui empêche l'injection en dissociant les données des requêtes SQL.

Filtre de Données : Symfony propose un composant de filtre de données qui permet de nettoyer, valider et formater les entrées utilisateur. Les filtres peuvent être utilisés pour s'assurer que les données respectent les formats attendus et ne contiennent pas de code malveillant.

Requêtes Préparées : Symfony encourage fortement l'utilisation de requêtes préparées, une technique qui sépare les instructions SQL des données. Les requêtes préparées utilisent des paramètres pour représenter les données, évitant ainsi toute injection de code malveillant.

Doctrine ORM (Object-Relational Mapping) : Symfony utilise Doctrine comme ORM, facilitant la manipulation de la base de données de manière sécurisée. Doctrine automatise la création de requêtes SQL sécurisées, éliminant ainsi les risques d'injections.

Paramètres de Requête : Symfony offre la possibilité de définir des paramètres de requête, ce qui permet de transmettre les valeurs utilisateur de manière sécurisée sans compromettre la structure de la requête SQL.

Contraintes dans les Formulaires Symfony : Les contraintes permettent de définir des règles sur les données des inputs. Symfony propose des contraintes natives, comme Length, NotBlank, et Regex, qui contribuent à valider et filtrer les données dès leur saisie.

Requêtes DQL avec setParameters : Dans Symfony, l'utilisation de Doctrine Query Language (DQL) pour construire des requêtes sécurisées est essentielle. Lors de l'utilisation de paramètres dans les requêtes DQL, la méthode setParameters permet de spécifier ces paramètres de manière sécurisée, évitant ainsi toute injection SQL.

Utilisation de Requêtes Préparées : Symfony encourage fortement l'utilisation de requêtes préparées pour séparer les instructions SQL des données utilisateur. Dans le contexte de Symfony, cela se traduit souvent par l'utilisation de Doctrine ORM, qui automatise la création de requêtes SQL sécurisées.

Validation des Inputs : Symfony offre des mécanismes de validation au niveau des formulaires avec l'utilisation de contraintes qui définissent les règles pour les inputs. Ces contraintes aident à garantir que les données respectent les attentes définies par l'application.

Symfony contient un système de validation des inputs utilisateurs basé sur les contraintes de validation. Dans tous mes formulaires Symfony, j’utilise les contraintes natives pour valider les types d’input, vérifier leur format (email, regex), longueur minimale et maximale, etc…

De plus, l’utilisation de Doctrine sécurise les requêtes DQL générées par défaut. Pour les requêtes DQL personnalisées, j’ai utilisé des placeholders et la méthode setParameter() afin de préparer mes requêtes et éviter les injections SQL.

Lorsqu'un utilisateur utilise le filtre de recherche pour trouver un spot par son nom, notre requête utilise un paramètre marqué par :search au lieu d'inclure directement la valeur dans la requête SQL.

Ce paramètre agit comme un espace réservé pour la véritable valeur de recherche. Ensuite, nous utilisons la méthode setParameter() pour affecter une valeur à ce paramètre.

**Ce mécanisme est appelé "requête préparée".**

Il permet de séparer la structure SQL de la donnée, rendant ainsi l'injection SQL beaucoup plus difficile. Doctrine, l'ORM que nous utilisons, gère ce processus en arrière-plan et s'assure que la valeur insérée est correctement échappée et sécurisée. Ce choix de conception nous permet de maintenir un haut niveau de sécurité dans l'accès à nos données.

### b. Protections contre les failles XSS (Cross-Site Scripting)

Détaillez les pratiques mises en place pour prévenir les attaques XSS, comme la validation des entrées, l'encodage, et l'utilisation de Content Security Policy (CSP).

s’agit d’une faille de sécurité permettant à l’attaquant d’injecter dans un site web un code **client** malveillant (souvent Javascript). Ce code sera ensuite exécuté par les victimes à leur insu et permet en général à l’attaquant de contourner les contrôles d’accès et d’usurper l’identité des utilisateurs. 71 Dossier de synthèse – Basile KUNTZ

Pour lutter contre ce type d’attaques, il convient de filtrer les inputs utilisateur et d’échapper les outputs dans les templates.

L’échappement Twig permet d’éviter l’exécution de scripts potentiellement présents dans les outputs en remplaçant par exemple les « < > » par leurs entités HTML équivalentes « &lt ; »… (comparable à htmlSpecialChars() en PHP). Cet échappement est exécuté par défaut par Symfony, mais peut-être explicité avec le filtre Twig « escape » : {{ var|escape }} ou annulé grâce au filtre « raw » : {{ var|raw }}

La faille XSS, ou Cross Site Scripting, est une vulnérabilité qui permet à un attaquant d'injecter du code malveillant dans une page web consultée par d'autres utilisateurs. L’attaquant peut voler des cookies, des sessions d'utilisateurs ou altérer le contenu de la page à des fins malveillantes. Les données malveillantes sont généralement injectées dans des champs de saisie, tels que les formulaires ou les commentaires. Lorsque d'autres utilisateurs consultent la page et affichent ces commentaires, le code malveillant est exécuté.

Pour prévenir les attaques XSS, nous utilisons une approche en deux temps. Tout d'abord, nous effectuons l'encodage des données de sortie au format HTML, ce qui signifie que les données sont converties en une forme qui ne peut pas être interprétée comme du code. Nous utilisons des filtres et des méthodes de désinfection pour nettoyer les données, éliminer les caractères suspects, et les valider selon les règles établies.

### c. Faille CSRF (Cross-Site Request Forgery) /

Prévention des Cross-Site Request Forgery (CSRF)

Décrivez les mesures adoptées pour prévenir les attaques CSRF, y compris l'utilisation de jetons anti-CSRF et une conception sécurisée.

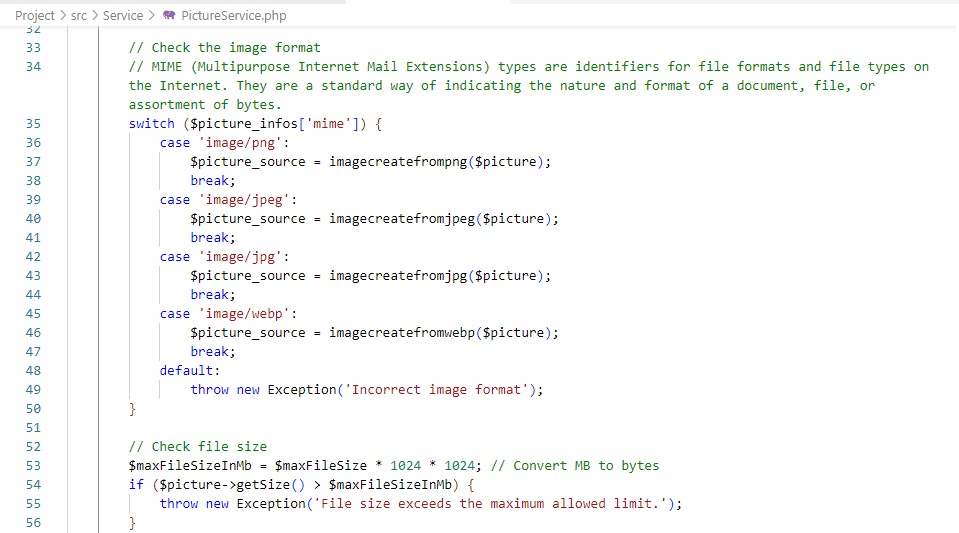
Décrivez comment votre application prévient les attaques CSRF en utilisant des jetons anti-CSRF et en adoptant des pratiques de conception sécurisée.

Expliquez comment votre application prévient les attaques CSRF, notamment par l'utilisation de jetons anti-CSRF et des pratiques de conception sécurisée.

### Sécurité lors de l’Upload de fichiers

La "faille upload" fait référence aux vulnérabilités associées à la fonctionnalité de téléchargement de fichiers dans une application web. Cette faille peut être exploitée par des attaquants pour télécharger et exécuter du code malveillant sur le serveur, compromettant ainsi la sécurité du système.

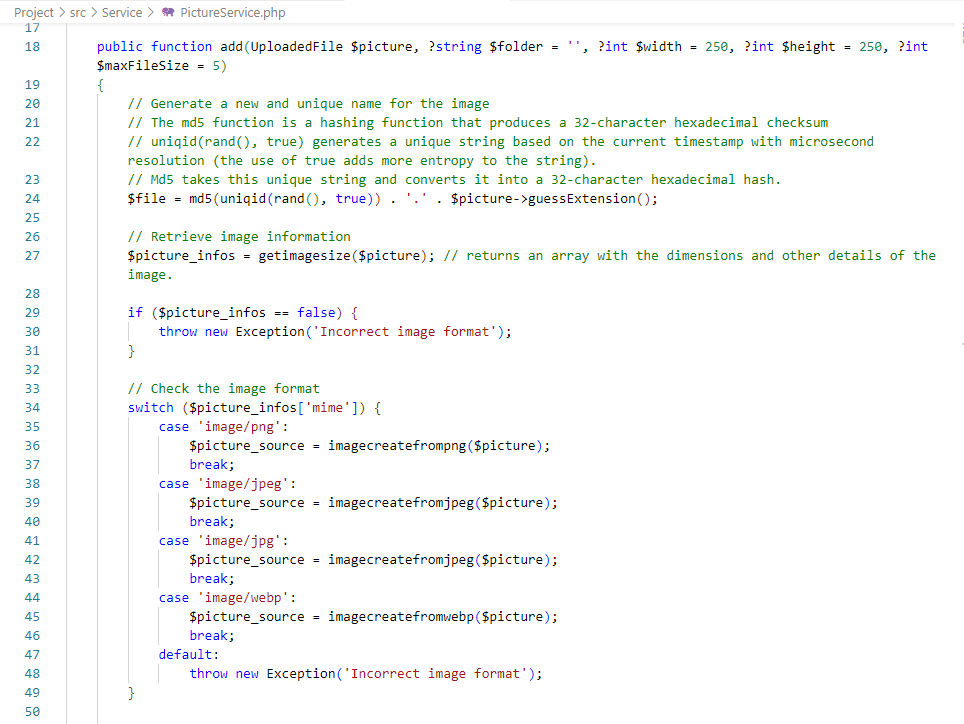
Afin de prévenir de telles attaques, plusieurs mesures ont été mises en place. Une validation exhaustive est effectuée à la fois côté client et côté serveur. Du côté client, une vérification rigoureuse s'assure que les fichiers téléchargés respectent des critères tels que la taille maximale (fixée à 5 Mo) et le type de fichier autorisé, déterminé par l'analyse du **MIME type[[3]](#footnote-3)** (limité aux extensions jpeg, jpg, png et webP dans mon cas). Cette validation est également implémentée côté serveur pour garantir que les fichiers téléchargés répondent aux critères définis.



Une limitation quant au nombre de fichiers pouvant être téléchargés par un utilisateur a également été instaurée, notamment pour les utilisateurs avec le statut "Artist" qui ont la possibilité de remplir une galerie d'images pour exposer leur travail. La limite d'images est fixée à XX, renforçant ainsi la sécurité du système.

SCREEN : Limitation nb de fichier pour artistes

Enfin, lors du processus d'upload d'une image, au niveau du service d'upload en backend, les fichiers sont automatiquement renommés avec des noms générés aléatoirement. Cette pratique vise à éviter tout problème lié aux noms de fichiers qui pourraient enfreindre les conditions générales d'utilisation (CGU), prévenir l'écrasement d'un fichier existant en cas de téléchargement d'un fichier portant le même nom, et empêcher l'utilisation de noms de fichiers prévisibles exploitables par des attaquants.



La faille XSS, abréviation de Cross-Site Scripting, est une vulnérabilité qui permet à un attaquant d'injecter du code JavaScript malveillant dans une page web. Ce code est ensuite exécuté par le navigateur de l'utilisateur final qui visite la page. Il y a principalement trois types de failles XSS :

XSS stocké : Le code malveillant est stocké de manière persistante sur le serveur et est servi à chaque utilisateur qui accède à une page particulière.

XSS réfléchi : Le code malveillant est inclus dans une URL et est exécuté lorsqu'un utilisateur clique sur un lien malveillant. **56**

XSS DOM-based : Le code malveillant manipule le Document Object Model (DOM) de la page, permettant à l'attaquant de modifier la structure de la page.

**b) Exemple**

Imaginons un formulaire sur un site web où les utilisateurs peuvent laisser des commentaires. Si le code récupère le contenu de ces commentaires et les insère directement dans la page, l’utilisateur qui la visite est vulnérable à une attaque XSS.

En effet, quand un utilisateur soumet un commentaire comme "Bonjour, c’est un super site !", il est directement inséré dans le code HTML de la page. Jusqu'ici, tout va bien.

Mais si un attaquant soumet un commentaire comme :

« <script>alert('Vous avez été piraté');</script> »

Ce commentaire malveillant est donc directement inséré dans le code HTML. Dès lors quand un autre utilisateur visite la page, son navigateur exécutera ce script déclenchant une alerte avec le message « Vous avez été piraté ». Cet exemple est simple, et en sois sans dangerosité réelle.

Mais un attaquant pourrait tout aussi bien utiliser un script plus sophistiqué pour rediriger l’utilisateur vers un autre site. Par exemple, un faux site qui imite l’original, et qui pousserait l’utilisateur à se connecter via un formulaire. Ce formulaire aurait pour effet de transmettre à l’attaquant les identifiants de l’utilisateur qui devient alors victime de la supercherie.

**c) Sécurité mis en place dans le projet**

Le framework Symfony offre des mécanismes de sécurité intégrés qui nous permettent de nous prémunir des failles XSS. Nous retrouvons l’échappement des variables par le moteur de template TWIG et la sécurisation des formulaires en utilisant les formType de Symfony, qui constitues 2 couches de sécurités pour s’en prémunir.

**(1) Echappement automatique des variables**

Lorsque le moteur de template Twig rencontre une variable, il applique une série de transformations notamment en employant la fonction native PHP « htmlspecialchars() » pour s'assurer que cette variable est affichée de manière sécurisée.

On dit alors qu’il « échappe » les variables.

Par exemple, si une variable contient des caractères spéciaux comme <, > ou &, qui sont typique des langages de programmation web, Twig les convertira en entités HTML sûres, comme &lt;, &gt; ou &amp;. De cette manière, le navigateur le traitera comme une simple chaine de caractères plutôt que comme un script à exécuter.

**(2) Sécurisation des formulaires**

Nous avons mis en place des formulaires en utilisant les FormType de Symfony, ce qui contribue à la sécurité de notre application. En effet, cette fonctionnalité nous aide à nous **57**

prémunir contre les risques d'injections XSS, en échappant automatiquement le contenu des champs de formulaire.

Par exemple, dans la classe SpotType, qui définit le formulaire permettant l’ajout d’un nouveau Spot :

Nous spécifions pour chaque champ du formulaire le type de donnée attendu. Cela permet d’éviter la validation du formulaire si des caractères qui ne correspondent pas sont présent, et qui pourrait faire part d’un script malveillant.

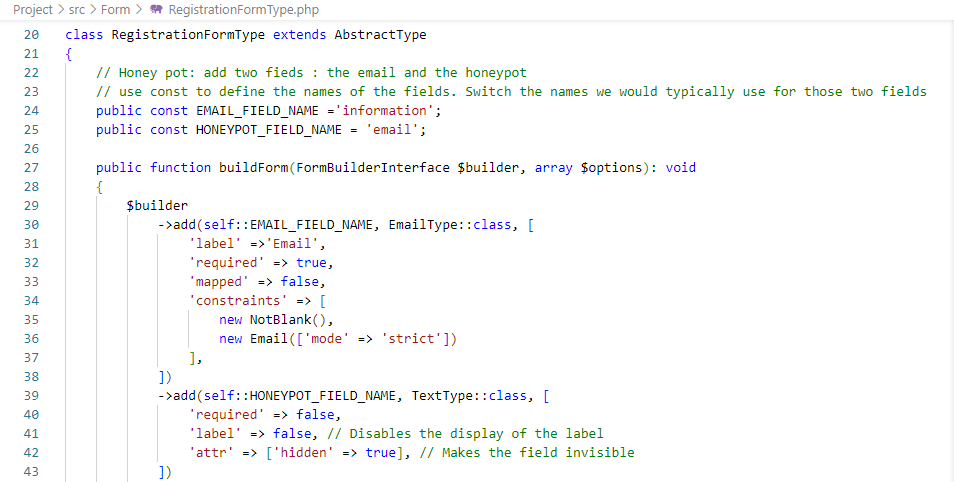
Symfony utilise la fonction PHP « filter\_var » pour valider et filtrer les données entrées par l'utilisateur dans les formulaires. Cette fonction est employée en fonction du type de champ défini dans le formulaire (par exemple, email, texte, nombre).

Dans l’exemple du formulaire de Spot, nous utilisons TextType pour créer un champ de texte appelé "name". Lorsque le formulaire est soumis, Symfony utilise filter\_var en interne pour valider et filtrer la valeur du champ "nom". Il s'assure que la valeur est une chaîne de caractères valide en utilisant le filtre FILTER\_SANITIZE\_STRING par défaut pour éliminer tout caractère non autorisé.

### Détection d’actions suspectes avec le Honey Pot

Dans le but de renforcer la sécurité de mon application, j'ai implémenté une technique appelée "Honey Pot" (ou « pot de miel). En termes simples, un honeypot est une méthode de surveillance qui consiste à créer des éléments factices pour attirer les attaquants et surveiller leurs activités. Ces leurres sont conçus pour simuler de véritables cibles et, lorsqu'un attaquant interagit avec eux, leurs actions sont surveillées, permettant ainsi de prendre des mesures pour bloquer ou atténuer les attaques.

Dans le cadre de mon application, la technique du Honey Pot a été mise en œuvre pour repérer des activités suspectes lors des inscriptions. Un champ caché, nommé de manière crédible (dans ce cas, "email"), a été intégré au formulaire. Le véritable champ d'email a été renommé "information". Un utilisateur légitime ne verra pas le champ caché et ne pourra, de ce fait, pas le compléter. Cependant, un bot le repérera et tentera de le remplir, dévoilant ainsi une tentative suspecte. Par la suite, des mesures appropriées telles que le blocage de l'adresse IP peuvent être mises en place.



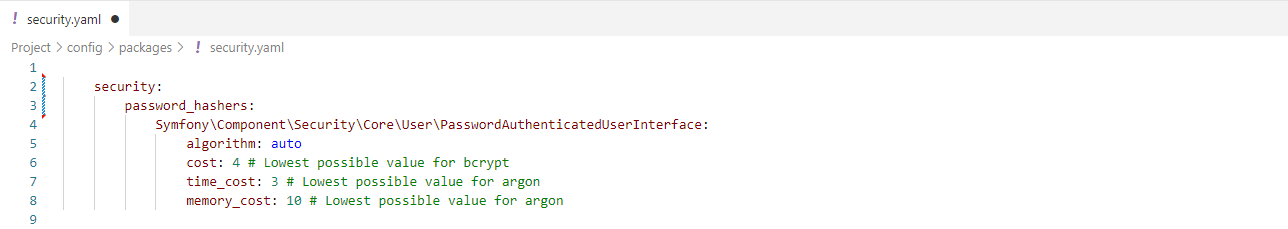
## 02. Gestion des mots de passe

### a. Hachage du mot de passe

Lorsqu'un utilisateur s'inscrit et que ses données sont enregistrées dans notre base de données, nous mettons en œuvre une mesure de sécurité essentielle en effectuant un processus de **hachage** (ou **chiffrement** en français) sur son mot de passe.

Ce processus de hachage crée **une empreinte numérique unique** du mot de passe, laquelle est rendue **irréversible et indéchiffrable**. C'est cette empreinte numérique cryptée qui est stockée dans notre base de données, jamais le mot de passe en texte clair de l'utilisateur. Cette approche garantit la protection complète du mot de passe et la sécurité des comptes utilisateurs, car il devient impossible de le récupérer directement. De plus, le choix d'algorithmes de hachage robustes tels que bcrypt et argon2i renforce la sécurité en rendant extrêmement difficile le décryptage du mot de passe, assurant ainsi la confidentialité des données des comptes clients et contrecarrant toute tentative de piratage.

Dans le cadre de notre application Symfony, la gestion automatisée de la sécurité des mots de passe est assurée par l'utilisation **du dernier et du plus sécurisé des algorithmes de hachage disponible[[4]](#footnote-4).** Symfony garantit que les mots de passe sont toujours sécurisés de la manière la plus robuste possible, même en anticipant l'introduction de nouveaux algorithmes dans les futures versions de PHP. Cette configuration est gérée de manière transparente dans le fichier security.yaml, où l'option "algorithm: auto" assure l'utilisation automatique du meilleur algorithme disponible. Actuellement, l'algorithme de hachage par défaut est « Bcrypt ».



Dans notre application, lorsque l'utilisateur soumet le formulaire d'inscription, le mot de passe est récupéré par le contrôleur dédié à la gestion des inscriptions (*RegistrationController*). À ce stade, le mot de passe est haché à l'aide de la fonction *hashPassword* issue de la classe *UserPasswordHasherInterface*. Cette fonctionnalité permet de convertir le mot de passe en une empreinte numérique sécurisée avant de le stocker de manière cryptée dans la base de données. Cette approche garantit la sécurité optimale des mots de passe des utilisateurs, renforçant ainsi la protection de leurs données sensibles.

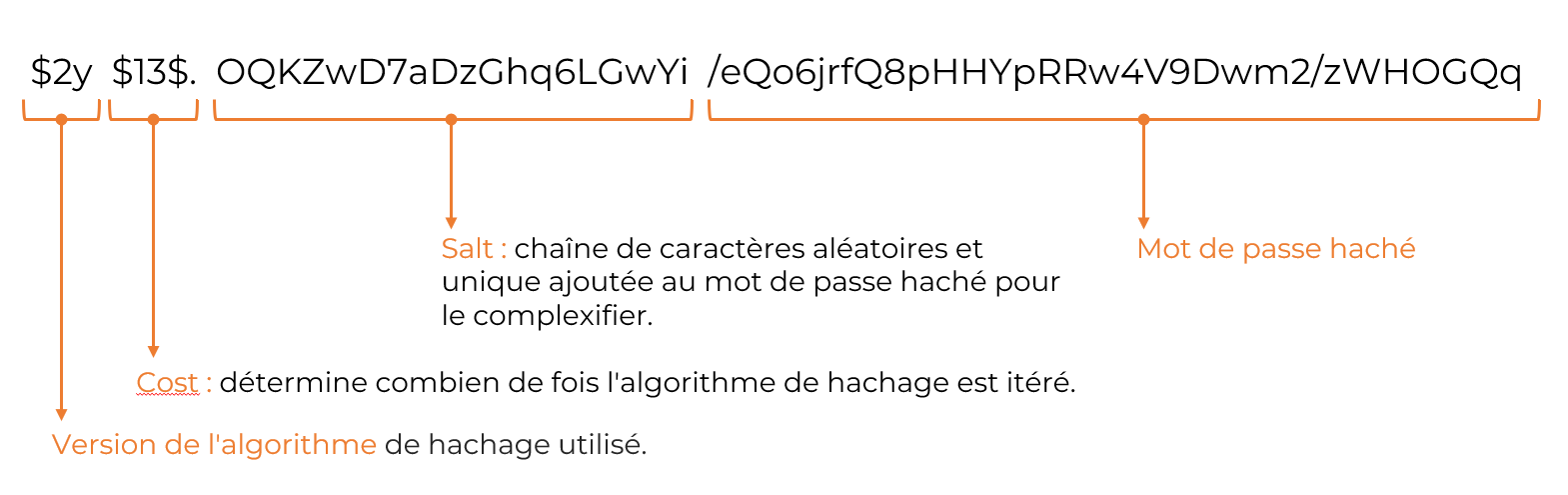


SCREEN : base de données avec empreinte numérique

L'empreinte numérique, une chaîne de caractères unique traitée dans la base de données, comprend plusieurs éléments cruciaux qui renforcent la sécurité des mots de passe en stockage. Cette empreinte intègre la version de l'algorithme de hachage utilisé, les options d'algorithme, le coût, un sel aléatoire, et enfin, le mot de passe haché proprement dit.

* La **version de l'algorithme** de hachage donne des informations sur la méthode spécifique utilisée pour sécuriser les mots de passe.
* Les options d'algorithme, notamment le **cost** (ou "coût) jouent un rôle essentiel dans la résistance aux attaques par force brute. Le coût, déterminant le nombre d'itérations de l'algorithme de hachage, rallonge délibérément le processus de hachage. Un coût plus élevé rend les attaques par force brute nettement plus lentes, complexifiant grandement leur réussite.
* **Le salt** (ou "set"), une chaîne de caractères aléatoire et unique ajoutée au mot de passe haché. Ce procédé renforce considérablement la sécurité initiale en prévenant les attaques par force brute, y compris celles exploitant des tables arc-en-ciel. Ces dernières sont des bases de données précalculées permettant de décrypter rapidement des mots de passe hachés lors d'attaques de force brute.
* Le **mot de passe haché** à proprement dit.

*Exemple d’empreinte numérique :*



De ce fait nous renforçons la sécurité des mots de passe en rallongeant délibérément le processus de hachage, complexifiant ainsi **les attaques par force brute[[5]](#footnote-5)**. De plus, l'utilisation du sel unique pour chaque mot de passe rend inefficaces **les attaques basées sur des dictionnaires[[6]](#footnote-6)**, tandis que la complexité accrue due à notre choix d'algorithme de hachage défie les **tentatives d'exploitation de tables arc-en-ciel[[7]](#footnote-7)**. En combinant ces mesures, notre système assure une protection robuste contre une variété d'attaques potentielles visant les mots de passe utilisateur.

### b. Politique du mot de passe

En plus du processus de hachage de mot de passe et d’empreinte numérique, il est crucial pour les utilisateurs d'adopter un mot de passe robuste dès la création de leur compte. La CNIL recommande **une politique du mot de passe** rigoureuse, incluant **des critères de complexité et d'unicité**. Ces règles définissent les exigences en termes de longueur et de complexité.

Cette politique vise à réduire les risques liés aux mots de passe faibles, renforçant ainsi la sécurité des comptes. Depuis la révision de 2022[[8]](#footnote-8), la CNIL préconise de nouvelles directives pour une robustesse équivalente à une entropie[[9]](#footnote-9) d'au moins 80 bits dans le cas où seul le mot de passe est utilisé

Dans le cadre de mon application, j'ai opté pour ce pattern proposé et approuvé par la CNIL à savoir :

"Les mots de passe doivent être composés d'au minimum 12 caractères, incluant des majuscules, des minuscules, des chiffres et des caractères spéciaux choisis parmi une liste d'au moins 37 caractères spéciaux possibles."

Ce schéma assure un niveau générique minimal de 80 bits d'entropie, vérifié via l'outil fourni par la CNIL.[[10]](#footnote-10)).

Malgré la recommandation de la CNIL d'une entropie minimale de 50 bits en cas de mesures supplémentaires, ma politique de mot de passe maintient une entropie de 80. Ceci garantit une sécurité renforcée, même avec la mise en place d'un mécanisme de restriction d'accès au compte, détaillé dans la partie suivante.

D’un point de vue technique, en back-end, nous utilisons des expressions régulières (RegEx pour « regular Expressions ») pour définir le modèle de chaîne de caractères à respecter. Cette RegEx impose des règles strictes sur la construction des mots de passe, assurant une complexité robuste.

La RegEx joue un rôle crucial en imposant un mot de passe complexe et robuste, compliquant l'accès aux comptes pour les utilisateurs malveillants. Elle contribue efficacement à contrer les attaques par dictionnaire en excluant des motifs communs. En exigeant des caractéristiques spécifiques, comme la longueur minimale, l'inclusion de chiffres et de majuscules, les RegEx préviennent les utilisateurs de choisir des mots de passe vulnérables.

En résumé, les RegEx sont un outil puissant pour imposer des règles strictes dans la politique de mots de passe, limitant les choix de mots de passe faibles. Toutefois, pour une sécurité robuste, le hachage des mots de passe et d'autres mesures de sécurité demeurent essentiels. La combinaison de ces méthodes offre une protection optimale contre diverses formes d'attaques.

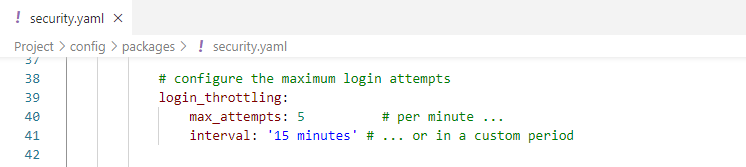
SCREEN : Regex mis en place + addflash quand non respect du regex sur la page d’inscription

### Limitation des tentatives de connexion

Pour renforcer davantage la sécurité contre les attaques de force brute, une mesure supplémentaire mise en place dans notre application consiste à limiter le nombre de tentatives de connexion (ou « login throttling) ». Cette mesure vise à dissuader les tentatives automatisées visant à compromettre les comptes utilisateurs.

Nous bloquons temporairement l'accès après 5 essais infructueux par minute, avec une période de blocage de 15 minutes en cas de dépassement. Cette stratégie est mise en œuvre grâce au bundle RateLimiter de Symfony, offrant la souplesse nécessaire pour ajuster les paramètres selon nos besoins spécifiques. Symfony/rate-limiter sert ainsi de dispositif clé pour limiter les tentatives de connexion infructueuses, réduisant efficacement le risque d'attaques par force brute.

En résumé, cette mesure vise à décourager les attaques automatisées en restreignant les tentatives infructueuses d'authentification. Le blocage temporaire après un certain nombre d'échecs assure une sécurité supplémentaire, tout en permettant aux utilisateurs légitimes de réessayer après une période d'attente définie.



SCREEN : login page avec addflash message et le temps d’attente

D’après le vérificateur mis en place par la cnil[[11]](#footnote-11)

1. **Attaques par Force Brute :**
   * **Politique de Mot de Passe Fort :** Imposer des politiques de mot de passe fort avec des exigences de complexité, y compris la longueur minimale, l'utilisation de caractères spéciaux, de chiffres et de lettres majuscules et minuscules.
   * **Bloquer les Tentatives Répétées :** Mettre en place des mécanismes de blocage des comptes après un certain nombre de tentatives de connexion infructueuses pour ralentir les attaques par force brute.
2. **Attaques par Dictionnaire :**
   * **Utilisation de Regex :** Définir des expressions régulières (regex) pour exclure des séquences de caractères couramment utilisées dans les mots de passe faibles ou qui pourraient être présentes dans un dictionnaire.
   * **Éducation des Utilisateurs :** Sensibiliser les utilisateurs à l'importance de choisir des mots de passe uniques et complexes.
3. **Attaques par Registre Arc-en-Ciel :**
   * **Sel (Salt) :** Utiliser des sels uniques pour chaque mot de passe avant le hachage, ce qui rend inefficaces les tables arc-en-ciel précalculées.
   * **Itérations de Hachage :** Augmenter le coût du hachage en utilisant des itérations élevées, rendant les attaques plus lentes et plus complexes.
4. **Utiliser des Algorithmes de Hachage Forts :**
   * **Bcrypt ou Argon2i :** Utiliser des algorithmes de hachage forts tels que Bcrypt ou Argon2i qui sont résistants aux attaques par force brute.
5. **Surveillance et Analyse des Tentatives Suspectes :**
   * **Systèmes de Détection d'Intrusion :** Mettre en place des systèmes de détection d'intrusion pour repérer et bloquer les activités suspectes, comme les tentatives massives de connexion.

**« HoneyPot »** ou **« Pot de miel »**

Bien que la mise en place d'un honeypot et l'utilisation de captchas partagent l'objectif général de renforcer la sécurité en détectant et prévenant les activités malveillantes, ce sont des mécanismes distincts avec des approches différentes.

1. **Honeypot :**
   * Un honeypot est une technique de surveillance qui consiste à créer de faux éléments, tels que de faux comptes ou de faux services, pour attirer les attaquants et surveiller leurs activités.
   * Les honeypots sont conçus pour tromper les attaquants en semblant être de véritables cibles. Lorsqu'un attaquant interagit avec un honeypot, son comportement est surveillé, et des mesures peuvent être prises pour bloquer ou atténuer les attaques.
2. **Captcha :**
   * Un captcha est un mécanisme de test de Turing conçu pour distinguer les utilisateurs humains des robots. Il présente généralement une tâche simple que les humains peuvent effectuer facilement, mais qui est difficile ou impossible pour les programmes automatisés.
   * Les captchas sont généralement utilisés dans les formulaires en ligne pour éviter les soumissions automatisées. Ils peuvent prendre la forme de puzzles, de questions, ou de défis visuels.

Bien que les honeypots et les captchas aient des objectifs similaires, ils sont utilisés dans des contextes différents et ont des mécanismes de fonctionnement distincts. Les captchas sont souvent utilisés pour protéger contre les soumissions automatisées de formulaires en ligne, tandis que les honeypots sont plus axés sur la détection d'activités malveillantes dans des environnements simulés.

En conclusion, bien que la mise en place d'un honeypot puisse contribuer à détecter les attaques automatisées, il n'est pas directement équivalent à un captcha. L'utilisation de captchas peut également être envisagée pour renforcer la sécurité en empêchant les soumissions automatisées dans certains contextes spécifiques.

### a. Captcha ?

## Authentification & Autorisation / Gestion des Accès et Authentification

### Connexion / / vérification par email / session !!

Processus d’authentification :

Détaillez les étapes de connexion, d'inscription, et de vérification par email, mettant en lumière les mécanismes de sécurité utilisés.

- Implémentation sécurisée des processus d'inscription et de connexion. - Utilisation de la vérification par email pour renforcer l'authentification.

### b. Authentification (forte)

Expliquez comment votre application met en œuvre des mécanismes d'authentification forte pour renforcer la protection des comptes utilisateurs.

Évoquez les méthodes telles que l'authentification à deux facteurs (2FA).

* Mentionnez la gestion des sessions pour renforcer la sécurité des connexions.

### c. Autorisation

### b. Contrôle des accès

* Décrivez comment votre application limite l'accès aux ressources uniquement aux utilisateurs autorisés.
* Mentionnez l'utilisation de rôles et de permissions pour renforcer la sécurité.

Mettez en avant la gestion des droits d'accès et les différents rôles dans l'application.

Détaillez la manière dont l'authentification et l'autorisation sont gérées pour assurer un accès sécurisé aux ressources.

### d. Gestion des droits d’accès:/ d’utilisateurs – Autorisation et gestion des droits d’accès

### user non co, user co, artist, supervisor, admin

* Présentez comment votre application gère les droits d'accès, avec une politique d'autorisation granulaire basée sur les rôles des utilisateurs.

### e. Les différents rôles

## 04. Sécurité dans l’application

### a. La gestion de la sécurité avec Symfony

* Décrivez comment votre application utilise les fonctionnalités de sécurité intégrées à Symfony pour protéger contre les vulnérabilités courantes.

### b. La sécurisation des formulaires

* Expliquez les pratiques de sécurisation des formulaires, y compris la validation côté serveur et côté client, la prévention des attaques CSRF, etc.

### a. Sécurisation des URLs / gestion error 404

* Présentez les mesures mises en place pour sécuriser les URLs, évitant les manipulations malveillantes et garantissant une navigation sûre.

### a. Sécurité système de paiement

transition/mini intro : Présentez brièvement l'importance de la sécurité dans le développement web et soulignez l'engagement de votre projet envers la protection des données et la confidentialité des utilisateurs.

Anonymisations

Google authentification

a.Chiffrement des Communications : Expliquez comment les communications entre le client et le serveur sont chiffrées en utilisant HTTPS.

c. **Echappement Automatique des Variables :** Détaillez comment votre application utilise l'échappement automatique des variables pour prévenir les attaques XSS.

# XIV. Traduction de texte

## 01. Contexte de la traduction

## 02. Texte original

## 03. Traduction

# XV. Axes d’améliorations

## 01. Améliorations court terme

## 02. Amélioration long terme

# XVI. Conclusion et Remerciements

## 01. Conclusion / défis ?

## 02. Remerciements

# XVII. Annexes

## 01. Schémas

## 02. Maquettage

## 02. Capture d’écran

## Notes :

Section sur l’upload des images ( redimensionnement, dossier, suppression etc)

Bien mettre légende pour screen quand nécessaire

Bien faire les phrases de transitions

Section sur OOP

1. Documentation et guide des bonnes pratiques de la RGPD destinés aux développeurs - https://www.cnil.fr/fr/guide-rgpd-du-developpeur [↑](#footnote-ref-1)
2. Documentation de Google concernant les bases de SEO. https://developers.google.com/search/docs?hl=fr [↑](#footnote-ref-2)
3. Un type de média (également connu sous le nom de types MIME pour Multipurpose Internet Mail Extensions) indique la nature et le format d'un document, fichier ou ensemble d'octets. Les types MIME sont définis et normalisés dans le RFC 6838 de l'IETF (Internet Engineering Task Force). [↑](#footnote-ref-3)
4. Documentation de Symfony concernant le hachage de mot de passe et la vérification. [https://symfony.com/doc/current/security/passwords.html#password-migration](https://symfony.com/doc/current/security/passwords.html%23password-migration) [↑](#footnote-ref-4)
5. Une attaque par force brute consiste à tester, l’une après l’autre, chaque combinaison possible d’un mot de passe ou d’une clé pour un identifiant donné afin se connecter au service ciblé. [↑](#footnote-ref-5)
6. Une attaque par dictionnaire est une attaque ciblant des mots de passe qui utilise des mots du dictionnaire ou des expressions courantes pour pirater les identifiants de connexion d'utilisateurs.  [↑](#footnote-ref-6)
7. Une attaque par table arc-en-ciel est une méthode utilisée pour déchiffrer les hachages de mot de passe dans une base de données à l'aide d'une table unique appelée « table arc-en-ciel ». [↑](#footnote-ref-7)
8. Documentation officielle de la CNIL concernant les nouvelles exigences pour la politique de mot de passe. <https://www.cnil.fr/sites/cnil/files/atoms/files/deliberation-2022-100-du-21-juillet-2022_recommandation-aux-mots-de-passe.pdf> [↑](#footnote-ref-8)
9. Définition de l’entropie par la CNIL : *« L’entropie peut être définie dans ce contexte comme la quantité de hasard. Pour un mot de passe ou une clé cryptographique, cela correspond à son degré d’imprédictibilité théorique, et donc à sa capacité de résistance à une attaque par force brute. »* [↑](#footnote-ref-9)
10. Outil mis à disposition par la CNIL pour mesurer l’entropie d’un mot de passe. <https://www.cnil.fr/fr/verifier-sa-politique-de-mots-de-passe> [↑](#footnote-ref-10)
11. Outil mis à disposition par la CNIL pour évaluer la robustesse et l’entropie des mots de passe. <https://www.cnil.fr/fr/verifier-sa-politique-de-mots-de-passe> [↑](#footnote-ref-11)