



Application web qui met en relation les voyageurs autour d'activités

*Projet réalisé dans le cadre de la présentation au  
Titre Professionnel Développeur Web et Web Mobile*



*présenté par  
Nadia Boujnah  
Ecole - La Plateforme*

# Sommaire

<b>Sommaire</b>	<b>2</b>
<b>Remerciements</b>	<b>4</b>
<b>Introduction</b>	<b>5</b>
<b>Listes des compétences couvertes par le projet</b>	<b>6</b>
1. Développer la partie front-end d'une application web ou web mobile sécurisée	6
A. Installer et configurer son environnement de travail en fonction du projet web ou web mobile	6
B. Maquetter des interfaces utilisateur web ou web mobile	6
C. Réaliser des interfaces utilisateur statiques web ou web mobile	7
D. Développer la partie dynamique des interfaces utilisateur web ou web mobile	7
2. Activité-type n° 2 : Développer la partie back-end d'une application web ou web mobile sécurisée	7
A. Mettre en place une base de données relationnelle.	7
B. Développer les composants d'accès aux données SQL et NoSQL.	8
C. Développer des composants métier côté serveur.	8
D. Documenter le déploiement d'une application dynamique web ou web mobile.	8
<b>Résumé du projet</b>	<b>9</b>
<b>Cahier des charges</b>	<b>9</b>
1. Contexte & Objectifs	9
2. Fonctionnalités Clés	10
3. Adaptations UX	11
4. Spécifications Techniques	11
5. User Stories	12
6. Feuille de route	13
7. Charte Graphique	14
<b>Objectifs</b>	<b>19</b>
1. Objectifs fonctionnels atteints	19
2. Objectifs techniques réalisés	19
<b>Mise en place de l'application</b>	<b>20</b>
1. Zoning	20
2. Wireframes	20
3. Prototypage	21
4. Chartes graphiques et logo	22
<b>Spécifications techniques</b>	<b>24</b>
<b>Choix techniques</b>	<b>25</b>
<b>Accessibilité</b>	<b>26</b>
<b>Architecture du projet</b>	<b>28</b>
<b>Sécurité</b>	<b>34</b>
1. Authentification et gestion des rôles	34
2. Protection CSRF	35
3. Sécurisation prod / CORS / sessions	35

4. Vérification d'identité et lutte contre les abus	37
5. Conformité RGPD	38
6. Journalisation et observabilité	40
7. Gestion des erreurs et UX d'échec	41
8. SEO / Publication	44
<b>Tests</b>	<b>46</b>
1. Tests unitaires	46
2. Tests fonctionnels	46
3. Cahier de tests	47
4. Couverture et périmètre des tests	48
<b>Gestion de projet</b>	<b>49</b>
1. Approche	49
2. Méthode de travail	49
3. Analyse des besoins	49
4. Conception	49
5. Développement	50
6. Déploiement	50
<b>Conclusion</b>	<b>52</b>
<b>Annexes</b>	<b>53</b>
Cahier des charges	53
Accesibilité	56
Sécurité	58
Test	63
Gestion de projet	64

## **Remerciements**

Je tiens à adresser mes sincères remerciements à mon formateur Sambeau.P, pour son encadrement tout au long de cette formation. Sa pédagogie, sa disponibilité et la qualité de ses conseils m'ont permis de progresser avec confiance dans ce projet, aussi bien sur le plan technique que méthodologique.

Je remercie également chaleureusement Aïcha.O, pour sa bienveillance, son écoute et sa motivation constante à mon égard. Son soutien m'a beaucoup aidée à garder le cap, à rester organisée et motivée dans les moments de doute ou de fatigue.

Un grand merci aussi à ma camarade de classe Yanina.R, pour son aide, sa patience et nos échanges constructifs. Son soutien m'a été précieux à plusieurs étapes du projet.

Enfin, je souhaite remercier mon mari, pour m'avoir encouragée chaque jour, pour sa patience, sa compréhension et son soutien moral tout au long de cette aventure. Son appui a été un véritable pilier dans ma réussite.

## Introduction

Depuis plusieurs années, je m'intéresse de près à l'univers de l'informatique, sans savoir exactement dans quelle voie m'orienter. Issue d'un parcours dans le commerce, où j'ai travaillé comme assistante commerciale et réalisé de nombreux emplois saisonniers, j'ai progressivement ressenti le besoin de me réorienter vers un domaine plus en lien avec ma passion pour le numérique.

Le développement web a commencé à m'attirer après avoir travaillé quelques mois sur un projet personnel, accompagné par des ateliers proposés par France Travail. Cette période m'a permis de mieux comprendre mes affinités, de découvrir les réalités du métier, et de confirmer que le développement était la voie qui me correspondait le mieux.

Cette démarche a été renforcée par une période d'immersion professionnelle chez Orange, également organisée avec France Travail. J'ai d'abord postulé à une formation DWWM à distance, mais c'est lors du processus de sélection que j'ai découvert La Plateforme, une école en présentiel située à quelques minutes de chez moi. Apprendre en présentiel me paraissait bien plus adapté à mon style d'apprentissage, ce qui m'a poussé à intégrer le cursus complet d'un an, après avoir réussi un test de sélection basé sur la logique.

Durant la formation, j'ai pu explorer à la fois le front-end et le back-end, même si j'ai une préférence pour le front. Mon objectif est cependant clair : devenir développeuse fullstack pour maîtriser l'ensemble du cycle de développement.

J'ai choisi de réaliser mon projet final individuellement de façon autonome et méthodique. Cela ne reflète pas une difficulté à travailler en groupe au contraire, je considère que c'est une de mes qualités mais ce projet m'a permis de prouver ma capacité à travailler seule de manière efficace.

En résumé, cette formation m'a permis de donner un vrai tournant à mon parcours, d'acquérir de solides compétences techniques, et surtout de construire un projet professionnel aligné avec mes valeurs et mes ambitions.

## Listes des compétences couvertes par le projet

### 1. Développer la partie front-end d'une application web ou web mobile sécurisée

#### A. Installer et configurer son environnement de travail en fonction du projet web ou web mobile

Pour le développement de mon application MeetTrip, j'ai mis en place un environnement de travail adapté aux exigences d'un projet fullstack, en combinant Laravel côté back-end et React avec Inertia.js côté front-end.

J'ai utilisé Visual Studio Code comme éditeur principal, enrichi de plusieurs extensions utiles telles que Prettier pour le formatage automatique du code, ESLint pour le linting JavaScript, Laravel Blade Snippets pour l'assistance sur Laravel, ainsi que Tailwind CSS IntelliSense pour l'auto-complétion des classes CSS.

Le projet a été initialisé avec Vite, qui permet un démarrage rapide du serveur de développement et une gestion efficace du bundling. Côté back-end, Laravel a été installé via Composer et configuré pour utiliser une base de données MySQL. J'ai également configuré un environnement local avec XAMPP pour exécuter le serveur Apache et la base de données en local.

Le gestionnaire de paquets npm m'a permis d'installer toutes les dépendances front-end nécessaires, comme React, Inertia.js et Tailwind CSS.

Enfin, le projet est versionné via Git avec un dépôt hébergé sur GitHub, afin d'assurer la traçabilité des modifications et de faciliter un suivi rigoureux en autonomie.

#### B. Maquetter des interfaces utilisateur web ou web mobile

Dans le cadre de mon projet MeetTrip, la phase de conception des interfaces utilisateur a été pensée de manière à garantir une navigation claire, intuitive et responsive pour les utilisateurs.

Je me suis appuyée sur une réflexion fonctionnelle autour des besoins des utilisateurs cibles (voyageurs, organisateurs) afin d'organiser les pages principales de l'application : page d'accueil, page de connexion/inscription, tableau de bord, page des activités, gestion des participants, messagerie, etc.

Avant de réaliser les maquettes sur Figma, j'ai d'abord effectué un zoning pour définir la structure générale de chaque page, puis créé les wireframes afin de préciser l'organisation des éléments et la navigation. J'ai ensuite finalisé les maquettes sur Figma en adoptant une approche mobile-first, en commençant par la version mobile avant d'adapter le design aux formats tablette et desktop. L'objectif était de concevoir une interface simple, claire et accessible, conforme aux principes d'ergonomie et d'accessibilité.

### C. Réaliser des interfaces utilisateur statiques web ou web mobile

Dans le cadre du développement de l'interface de MeetTrip, j'ai commencé par créer les pages statiques de l'application en utilisant HTML, CSS via Tailwind CSS, et JSX avec React. L'objectif était de structurer les éléments visuels essentiels avant d'y intégrer toute logique dynamique ou interaction avec les données.

J'ai mis en œuvre une approche mobile-first, afin de garantir une bonne lisibilité et une accessibilité optimale sur les écrans de petite taille, avant de gérer les adaptations pour les tablettes et les ordinateurs de bureau via les classes utilitaires responsive de Tailwind.

L'utilisation de Tailwind CSS m'a permis de structurer rapidement des interfaces modernes, tout en gardant un bon niveau de personnalisation et de cohérence dans le design. J'ai également respecté les règles d'accessibilité de base (balises sémantiques, contrastes, navigation clavier).

Cette phase statique a constitué une base solide avant l'intégration des données dynamiques et des interactions côté client avec React et Inertia.js.

### D. Développer la partie dynamique des interfaces utilisateur web ou web mobile

Pour rendre l'application MeetTrip interactive, j'ai utilisé React avec Inertia.js, intégré via le starter kit de Laravel. Ce choix m'a permis de bénéficier d'un environnement fullstack cohérent, reliant directement le front-end et le back-end sans avoir à gérer manuellement des requêtes API.

Les données sont transmises directement par Laravel au front-end, ce qui facilite la gestion des pages et des composants dynamiques. Inertia permet également une navigation fluide sans recharge complet, améliorant la rapidité et l'expérience utilisateur tout en conservant une architecture moderne et réactive.

## 2. Activité-type n° 2 : Développer la partie back-end d'une application web ou web mobile sécurisée

### A. Mettre en place une base de données relationnelle.

Avant de développer l'application MeetTrip, j'ai réfléchi à la structure globale des données afin d'assurer une organisation claire et cohérente dès le départ.

J'ai commencé par réaliser un Modèle Conceptuel de Données (MCD) pour définir les différentes relations entre les éléments du projet. Plusieurs ajustements ont été nécessaires afin d'assurer la cohérence et l'efficacité de la structure de la base.

Une fois le MCD validé, je l'ai transformé en Modèle Logique de Données (MLD) afin de préciser les clés primaires, étrangères et les attributs techniques nécessaires à la mise en œuvre.

Sur le plan technique, j'ai utilisé Laravel avec le système de base de données SQLite, intégré au framework. J'ai mis en place les migrations Laravel, ce qui m'a permis de créer et modifier les tables facilement tout au long du développement.

Enfin, j'ai intégré une carte interactive (Leaflet) permettant d'afficher les activités géolocalisées et de faciliter la navigation et la recherche d'événements selon leur emplacement.

## B. Développer les composants d'accès aux données SQL et NoSQL.

Comme j'utilise le framework Laravel pour mon projet MeetTrip, j'ai utilisé Eloquent ORM (Object Relational Mapping) pour développer les composants d'accès aux données SQL. Eloquent est un outil intégré à Laravel qui permet de gérer facilement les interactions avec une base de données relationnelle en utilisant une approche orientée objet. Grâce à lui, j'ai pu effectuer des opérations courantes comme l'ajout, la modification, la suppression ou la récupération de données, sans écrire manuellement de requêtes SQL.

Les relations entre entités ont été définies directement dans les modèles Laravell, en utilisant des méthodes comme hasMany, belongsTo ou belongsToMany.

## C. Développer des composants métier côté serveur.

Pour développer les composants métier côté serveur de mon application MeetTrip, j'ai utilisé le framework Laravel, qui permet d'organiser clairement la logique métier.

La logique de traitement des données, les règles de validation, et la gestion des fonctionnalités ont été intégrées dans les contrôleurs du projet. Laravel propose une structure claire qui sépare les rôles de chaque partie du code, ce qui permet de maintenir une application lisible, évolutive et structurée.

La communication avec la base de données s'appuie sur les modèles et les relations, ce qui facilite la gestion des interactions entre les différentes entités.

Laravel propose également des outils intégrés pour sécuriser les accès, gérer les redirections, valider les formulaires et appliquer des règles spécifiques selon le contexte d'utilisation.

## D. Documenter le déploiement d'une application dynamique web ou web mobile.

Pour reproduire un environnement de production réaliste, j'ai utilisé Docker afin de rendre l'application MeetTrip autonome et facilement déployable.

J'ai configuré plusieurs conteneurs pour exécuter PHP/Laravel, MySQL, Node.js (pour Vite) et Nginx, puis automatisé leur lancement à l'aide de Docker Compose.

J'ai ensuite adapté le projet à cet environnement en modifiant le fichier .env, et en créant un Dockerfile personnalisé. Mais également en configurant Nginx pour servir l'application.

Ce déploiement m'a permis de reproduire un environnement proche d'un hébergement réel, tout en assurant la portabilité et la cohérence du projet sur différents systèmes.

## Résumé du projet

Le projet MeetTrip est une application web conçue pour faciliter les rencontres entre voyageurs autour d'activités partagées. L'idée est née d'un constat simple : voyager seul peut être enrichissant, mais aussi source d'isolement. MeetTrip a donc pour objectif de créer du lien social entre des personnes qui ne se connaissent pas, en leur permettant de se retrouver pour vivre des moments conviviaux, que ce soit lors d'une balade, d'un repas ou d'une activité locale.

L'application permet aux utilisateurs de s'inscrire, de créer une activité en tant qu'organisateur, ou de rejoindre une activité en tant que participant. Chaque activité est associée à un lieu, une description, une date, un nombre de participants, et un créateur. Une messagerie intégrée permet aux participants de contacter l'organisateur pour poser des questions ou échanger avant l'événement.

Le projet a été développé avec Laravel côté back-end et React + Inertia.js côté front-end, en adoptant une structure fullstack. Une carte interactive réalisée avec Leaflet permet aux utilisateurs de repérer les activités sur une carte en fonction de leur localisation.

MeetTrip a été pensé comme un outil simple, intuitif et accessible à tous les profils de voyageurs, qu'ils soient étudiants, backpackers, nomades digitaux ou expatriés. Il vise à briser la solitude du voyage en rendant les rencontres spontanées plus faciles, sécurisées et organisées.

Ce projet a permis de mobiliser des compétences techniques complètes (base de données, interface dynamique, logique métier, sécurité, déploiement), tout en mettant en œuvre une démarche utilisateur centrée sur l'expérience et la simplicité d'usage.

## Cahier des charges

### 1. Contexte & Objectifs

#### Contexte

MeetTrip est une application web permettant de mettre en relation des voyageurs autour d'activités locales partagées.

L'objectif est de faciliter les rencontres humaines pendant les voyages, de lutter contre la solitude et de créer un espace d'échange entre voyageurs et habitants locaux

## Objectifs

- Favoriser la rencontre et le partage entre voyageurs.
- Simplifier l'organisation et la participation à des activités locales.
- Offrir une expérience fluide, intuitive et accessible sur tous les supports.

## 2. Fonctionnalités Clés

Côté Utilisateur (Front-office)

Fonctionnalité	Description
<b>Inscription / Connexion</b>	Accès à l'espace personnel selon le rôle (participant ou organisateur).
<b>Carte interactive</b>	Affichage des activités géolocalisées via Leaflet.
<b>Création d'activité</b>	L'organisateur peut publier une activité (lieu, description, date, nombre de places).
<b>Inscription à une activité</b>	Le participant peut rejoindre une activité selon la date choisie.
<b>Messagerie intégrée</b>	Échanges directs entre organisateurs et participants.

Côté Administrateur (Back-office)

Fonctionnalité	Description
<b>Gestion des utilisateurs</b>	Validation, suspension ou suppression des comptes.
<b>Gestion des activités</b>	Suppression ou modération des annonces.
<b>Tableau de bord</b>	Statistiques sur les utilisateurs, activités, et identités vérifiées.

### 3. Adaptations UX

- Design responsive (mobile-first, tablette et desktop).
- Interface claire et conviviale conçue sur Figma.
- Utilisation de Tailwind CSS pour un rendu moderne et rapide.
- Navigation simplifiée grâce à une structure intuitive et des couleurs apaisantes.

### 4. Spécifications Techniques

Stack :

- Backend : Laravel 10
- Frontend : React avec Inertia.js
- ORM : Eloquent
- Base de données : SQLite (puis MySQL au déploiement)
- Carte interactive : Leaflet.js
- CSS : Tailwind

Voir Annexe 21 – Guide utilisateur et parcours clés  
(présentant les 6 étapes principales d'utilisation de MeetTrip : inscription, création d'activité, messagerie, profil, etc.)

### 5. User Stories

#### Utilisateurs

En tant que	Je veux	Afin de
Voyageur	Créer un compte	Accéder à la plateforme et participer à des activités.
Voyageur	Rejoindre une activité	Rencontrer d'autres voyageurs.
Organisateur	Publier une activité	Proposer un événement local.
Organisateur	Consulter mes réservations	Gérer les participants.

#### Administrateur

En tant que	Je veux	Afin de
Administrateur	Accéder au tableau de bord	Gérer les utilisateurs et les activités.
Administrateur	Supprimer une activité	Modérer le contenu de la plateforme.

## 6. Feuille de route

Voici un extrait de ma feuille de route, centré sur les étapes clés :

 MeetTrip

**Feuille de route**

 **Brief**

Réalisation d'une application web permettant de mettre en relation des voyageurs à travers le monde autour d'activités locales organisées par des particuliers.

L'objectif est de favoriser les rencontres humaines, le partage culturel et l'organisation d'activités conviviales (visites, sorties, randonnées, ateliers...).

 **À qui ça s'adresse**

Voyageurs solo (étudiants, backpackers, expatriés, nomades digitaux) souhaitant rencontrer d'autres personnes pendant leurs séjours.

Locaux désireux d'organiser des activités et d'échanger avec des visiteurs.

Administrateurs assurant la modération, la gestion des utilisateurs et des activités.

<b>Questions</b>	<b>Réponses</b>
Qui veut découvrir de nouvelles activités ?	Des voyageurs et locaux souhaitant partager des moments conviviaux
Comment les faire se rencontrer ?	Grâce à une plateforme simple permettant de créer, rejoindre et discuter autour d'activités géolocalisées.
Où peut-on participer ?	Partout dans le monde, avec une carte interactive (Leaflet) affichant les activités proches.

Pourquoi MeetTrip ?	Pour rendre le voyage plus humain et connecter les personnes à travers des expériences partagées
---------------------	--

(Voir Annexe 1 – Personas représentatifs des utilisateurs MeetTrip)

(Voir Annexe 2 – Inspirations issues de plateformes communautaires)

(Voir Annexe 3 – Plan du site MeetTrip)

(Voir Annexe 4 – Schéma d'architecture technique du projet)

## 7. Charte Graphique

Élément	Détail
<b>Couleurs principales</b>	Dégradé bleu-vert (#247BA0 / #3CAEA3) + beige clair
<b>Typographie</b>	Sans-serif moderne et lisible
<b>Style visuel</b>	Épuré, chaleureux, orienté voyage et convivialité
<b>Logo</b>	MeetTrip – “Rencontrez, partagez, voyagez”

## 7. Produit Minimum Viable (MVP)

Le Produit Minimum Viable (MVP) de MeetTrip vise à proposer une version fonctionnelle et fluide de la plateforme, centrée sur la mise en relation entre voyageurs et organisateurs d'activités locales.

Cette première version regroupe les fonctionnalités indispensables à l'expérience utilisateur, à la gestion des activités, et à l'administration du site.

Fonctionnalités principales incluses dans le MVP :

### Authentification

- Connexion et inscription des utilisateurs via le système Laravel Auth.
- Rôles distincts : participant, organisateur, administrateur.
- Vérification d'adresse e-mail à l'inscription.

### Activités

- Création, modification et suppression d'activités par les organisateurs.
- Affichage des activités sur la page principale avec recherche et filtres.
- Détail d'une activité : description, lieu, carte Leaflet et bouton “Réserver”.

### Messagerie

- Système d'échange entre participants et organisateurs depuis une activité.
- Stockage des messages en base de données avec date d'envoi et auteur.

## Messagerie

[Nouveau message](#)

Rechercher par activité, organisateur, ville ou pays...

	<b>Claire Morel</b> <i>Coucher de soleil</i>	16-05-2025	<a href="#">Non lu</a>
de soleil	Bonjour, j'ai réservé votre activité, avez-vous besoin que j'apporte quelque chose ?		
	<b>Marc Delacroix</b> <i>Excursion bateau</i>	15-05-2025	
bateau	Bonjour Marc, merci pour la validation ! Dois-je prévoir quelque chose pour l'excursion ?		

## Carte interactive

- Utilisation de Leaflet.js pour afficher les activités géolocalisées.
- Chaque activité est représentée par un marqueur cliquable donnant accès à ses détails.

## Carte des activités

Rechercher une activité, ville ou organisateur...



## Gestion utilisateurs (Admin)

- Accès à un back-office sécurisé permettant la modération des comptes, des activités et la validation d'identités.

#### A. Front-office (utilisateurs)

- Page d'accueil : présentation de la plateforme et accès rapide aux activités.
- Liste des activités : affichage des activités disponibles avec filtres par lieu.
- Détail d'activité : carte, description, nombre de participants, bouton “Réserver”.
- Tableau de bord utilisateur : affichage des activités créées, réservées et des messages.

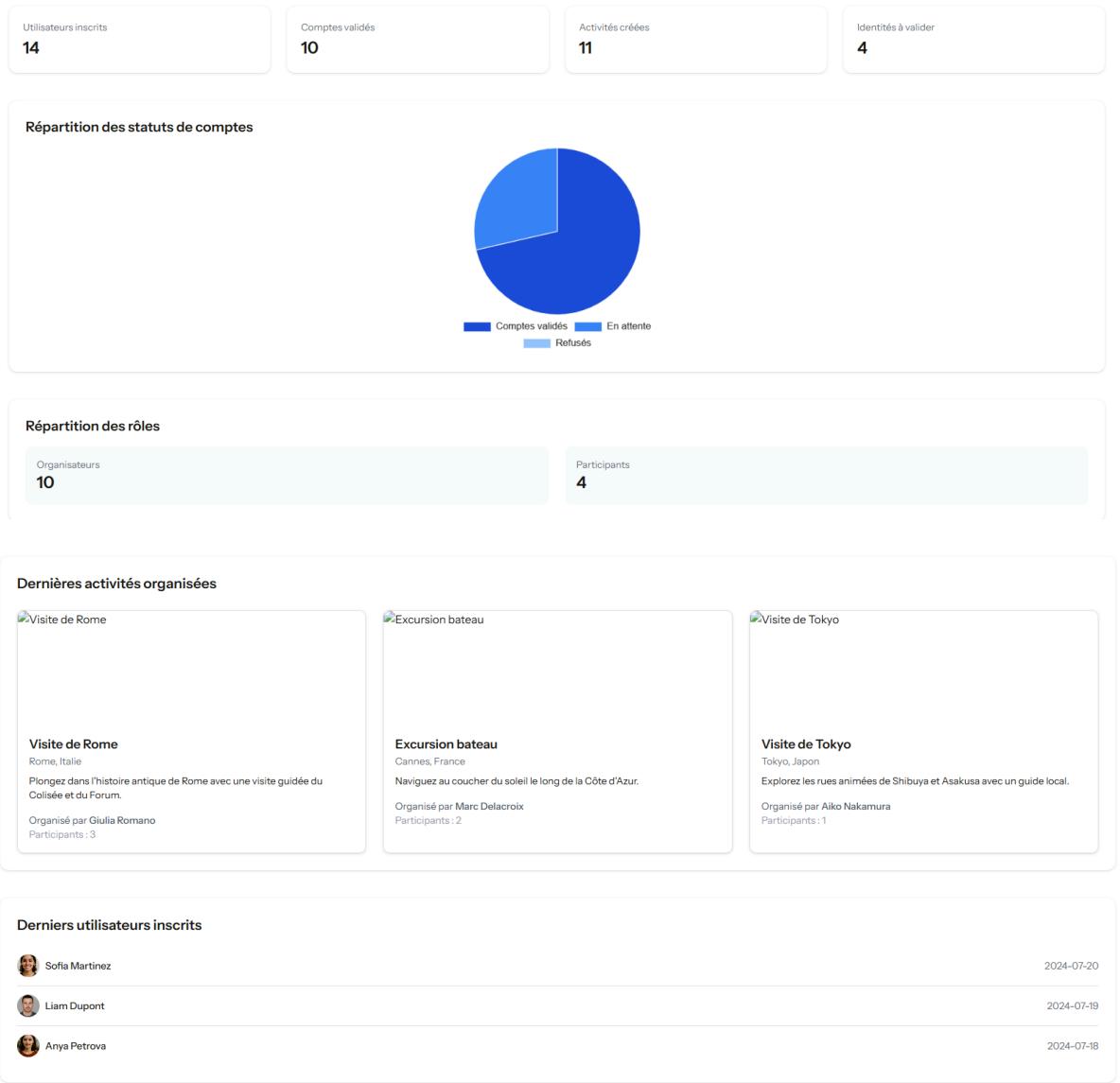
The screenshot shows the MeetTrip user dashboard. At the top, there's a header with the logo and a "Accueil" button. On the left, a sidebar menu includes: Accueil, Profil, Activités, Mes réservations, Carte, Messagerie, Annonces, Paramètres, and Déconnexion. A blue notification badge with the number "2" is visible above the "Annonces" link. The main content area displays the user's profile ("nana", checked account), a summary of reservations (15-08-2025 - Coucher de soleil; 10-08-2025 - Excursion bateau), a messaging section with two new messages from Claire Morel and Marc Delacroix, and a section for user ads (25 juillet 2025 - Atelier poterie; 28 juillet 2025 - Pique-nique bord de mer). Buttons for "Voir mes réservations", "Voir toutes les activités", "Voir tous les messages", and "Créer une annonce" are present.

#### B. Back-office (administrateur)

- Gestion des utilisateurs : visualiser, supprimer, ou valider les comptes.
- Gestion des activités : modération des annonces créées.
- Statistiques : nombre d'activités publiées, utilisateurs inscrits, identités à valider.

## Tableau de bord Administrateur

Vue d'ensemble des indicateurs clés

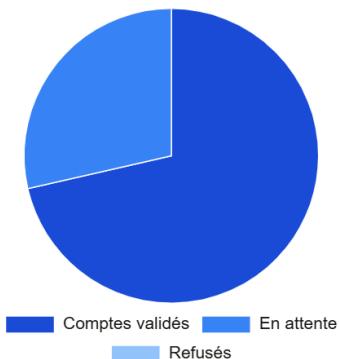


## C. Dashboard (Statistiques)

Un tableau de bord présente les indicateurs clés du projet :

- Nombre total d'utilisateurs, d'activités et de comptes validés.
- Graphique de l'évolution des activités créées dans le temps (via Chart.js).
- Répartition des rôles (participants, organisateurs).

#### Répartition des statuts de comptes



#### Répartition des rôles

Organisateurs

**10**

Participants

**4**

## Objectifs

Le projet **MeetTrip** avait pour objectif principal de créer une application web et mobile permettant de mettre en relation des voyageurs et des locaux autour d'activités partagées. L'application a été conçue comme une solution simple, intuitive et sécurisée afin de favoriser les échanges humains, de briser la solitude du voyageur solo et de faciliter l'organisation de rencontres conviviales, culturelles ou sportives.

### 1. Objectifs fonctionnels atteints

- Mise en place d'une inscription et connexion sécurisée pour les utilisateurs.
- Création d'un module permettant de publier et rejoindre des activités (balades, repas, sorties, etc.).
- Intégration d'une carte interactive avec géolocalisation des activités à proximité.
- Développement d'une messagerie intégrée liée aux activités, favorisant la communication entre organisateurs et participants.
- Conception d'un tableau de bord personnalisé adapté à chaque rôle (participant, organisateur, administrateur).

### 2. Objectifs techniques réalisés

- Développement de l'application en fullstack avec Laravel (back-end) et React + Inertia.js (front-end).

- Conception d'une base de données relationnelle assurant la gestion des utilisateurs, rôles et activités.
- Mise en place des mesures de sécurité (authentification, CSRF, CORS, rôles et permissions).
- Développement d'une interface responsive adaptée aux différents supports (ordinateur, tablette, mobile).
- Déploiement de l'application via Docker, garantissant un environnement stable, reproductible et facilement portable. Le conteneur regroupe toutes les dépendances nécessaires (serveur web, PHP, base de données, front-end compilé), ce qui simplifie la mise en production et la maintenance.

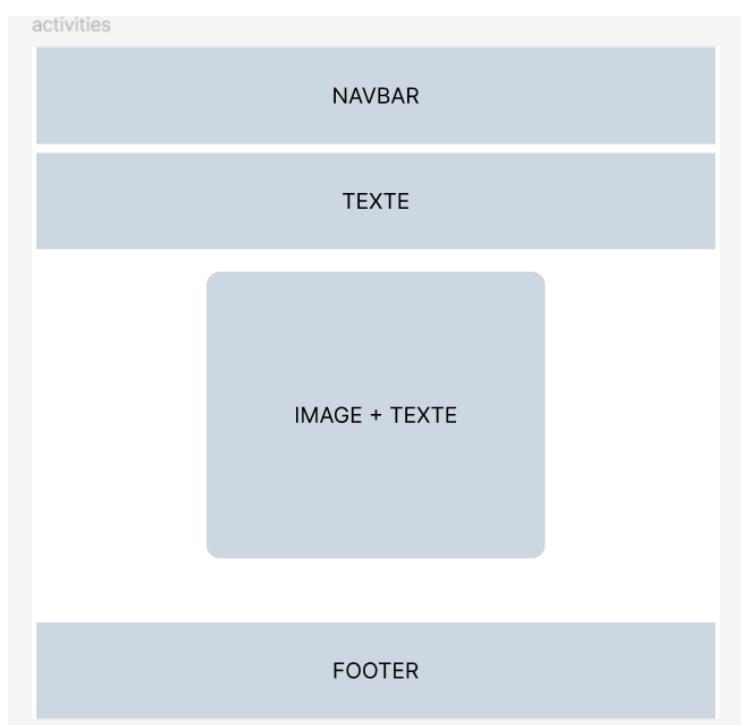
Enfin, le projet a permis de répondre aux objectifs pédagogiques du titre professionnel *Développeur Web et Web Mobile*, en mobilisant l'ensemble des compétences front-end et back-end, en intégrant les recommandations de sécurité, et en menant à bien un projet complet de la conception au déploiement.

## Mise en place de l'application

### 1. Zoning

Le zoning est une étape essentielle de conception qui permet de structurer les pages de l'application en zones fonctionnelles selon leur usage.

Pour MeetTrip, cette phase a servi à définir la hiérarchie visuelle et la répartition des éléments clés : navigation, carte, activités, formulaires, tableau de bord, etc.



## 2. Wireframes

Les wireframes traduisent le zoning en une première représentation visuelle des écrans. Réalisés sur Figma, ils illustrent le parcours utilisateur et la disposition des éléments d'interface avant la phase de maquette finale.



## 3. Prototypage

Le prototypage a permis de simuler les principaux parcours utilisateurs et de valider les interactions clés avant le développement.

The screenshot shows the 'Activités' (Activities) section of the MeetTrip website. At the top, there's a navigation bar with the MeetTrip logo, 'Accueil', 'Activités', 'À propos', 'Contact', 'Connexion', and 'Inscription'. Below the navigation is a search bar with placeholder 'Rechercher une activité...', and dropdown menus for 'Pays' and 'Type d'activité'. The main content area displays three travel activities in cards:

- Visite de Rome** (3 participants): A photo of the Colosseum at sunset. Details: Rome, Italie. Prochaines dates: 25/05/25 - 30/05/25. Organisé par: Julie Marteau. [Voir plus](#)
- Excursion bateau** (2 participants): A photo of people on a sailboat at sunset. Details: Cannes, France. Prochaines dates: 25/05/25 - 30/05/25. Organisé par: Marc De La Croix. [Voir plus](#)
- Désert D'Agafay** (4 participants): A photo of a person walking in a vast desert landscape. Details: Marrakech, Maroc. Prochaines dates: 25/05/25 - 30/05/25. Organisé par: Youssef Benali. [Voir plus](#)

At the bottom of the activities section, there's a navigation bar with icons for back, forward, and search.

#### 4. Chartes graphiques et logo

Pour le projet MeetTrip, une application de mise en relation entre voyageurs et organisateurs d'activités locales, j'ai choisi une direction graphique moderne, conviviale et inspirante.

L'objectif était de refléter les valeurs du partage, de la découverte et de l'aventure, tout en proposant une expérience claire, dynamique et agréable pour l'utilisateur.

La palette de couleurs s'articule autour de tons bleus et verts dégradés, symbolisant la nature, la mer et le voyage. Ces couleurs évoquent à la fois la sérénité et la liberté, tout en renforçant l'identité d'une plateforme tournée vers l'évasion et les rencontres.

La couleur d'accent, un bleu turquoise lumineux, apporte de la fraîcheur et attire le regard sur les éléments interactifs (boutons, liens, actions principales).

Avant de passer à la conception, j'ai réalisé un moodboard réunissant des visuels inspirants liés au voyage : paysages, cartes, avions, nature, exploration, et moments de convivialité.

Cette étape m'a permis de définir l'ambiance visuelle de MeetTrip et de guider mes choix graphiques dès le début du projet.



## Identité visuelle : couleurs et typographie

L'identité visuelle de MeetTrip a été pensée pour inspirer la confiance, la curiosité et le partage.

Le logo associe un symbole de voyage (globe et avion) à une typographie arrondie et lisible, renforçant l'idée d'accessibilité et de communauté.

### Choix des couleurs

La palette principale repose sur un dégradé bleu-vert (#007BFF → #00C896), évoquant la mer et la nature.

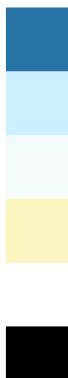
Ces teintes transmettent l'idée de mouvement, d'ouverture et de sérénité, en cohérence avec le concept de rencontres et de voyages.

Des nuances plus claires sont utilisées pour l'arrière-plan et les cartes, afin de garder une interface épurée et agréable à la lecture.

IMAGES



COLORS



#2876A7

#CFF0FF

#F9FAFB

#FEF9C2

#FFFFFF

#000000



**En quoi consiste MeetTrip ?**

MeetTrip est l'application qui connecte les voyageurs solo, backpackers et étudiants du monde entier. Partagez vos activités, repas ou trajets et brisez la solitude en voyage.

[En savoir plus](#)




## Nos chiffres clés

MeetTrip c'est une communauté en pleine expansion grâce à vous !



### Activités disponibles à partager



Les visuels utilisés pour MeetTrip ont été sélectionnés avec soin afin d'évoquer l'esprit du voyage, du partage et de la rencontre. Chaque image met en avant des personnes souriantes, des paysages emblématiques et des moments d'échange, renforçant l'idée d'une communauté ouverte et bienveillante.

Les tons chauds et lumineux rappellent les dégradés bleus et verts de la charte graphique, tout en apportant une touche humaine et authentique. Ces choix visuels participent à créer une atmosphère accueillante et inspirante, incitant les utilisateurs à s'immerger dans l'univers du voyage collaboratif.

## Spécifications techniques

### Utilisation de Git

Pour le suivi et la gestion du code de mon projet MeetTrip, j'ai utilisé Git avec un dépôt hébergé sur GitHub. Le dépôt est resté privé durant le développement, puis sera rendu public pour la soutenance.

J'ai mis en place une organisation claire par branches, afin de séparer les différentes parties du projet. Par exemple :

- une branche pour les pages publiques (accueil, activités, connexion, etc.),

- une autre pour les pages privées accessibles aux utilisateurs connectés,
- et des branches spécifiques pour certaines fonctionnalités comme la messagerie ou le dashboard administrateur.

Chaque fonctionnalité ou correction fait l'objet d'une branche dédiée, nommée selon une convention simple :

- feature/... pour une nouvelle fonctionnalité (ex. feature/ajout-activites),
- fix/... pour une correction (ex. fix/pagination-activites).

Le cycle de travail est structuré de la manière suivante :

1. Je commence par récupérer la dernière version du projet avec git pull.
2. Je développe ou teste la nouvelle fonctionnalité.
3. J'ajoute mes modifications à l'index avec git add -a, puis je fais un commit clair et descriptif, par exemple :
  - git commit -m "Mise à jour de la page activités"
  - git commit -m "Ajout du filtrage par lieu"
4. Ensuite, j'envoie mes changements sur le dépôt distant avec git push.
5. Si nécessaire, je crée une Pull Request pour relire ou fusionner les modifications dans la branche principale.

J'utilise aussi régulièrement les commandes git status pour suivre l'état des fichiers modifiés et git log pour vérifier l'historique de mes commits.

Ce fonctionnement me permet de travailler de manière organisée et sécurisée, en gardant un historique clair de chaque évolution, tout en facilitant le retour en arrière si besoin.

Grâce à cette méthode, chaque étape du développement est documentée, et le code reste structuré tout au long du projet.

*(Voir Annexe 23 – Méthodologie de versionnement et workflow Git du projet MeetTrip)*

## Choix techniques

Front-end

Le front a été réalisé avec React (composants réutilisables, état local simple) et Inertia.js pour relier directement les pages Laravel aux vues React sans API séparée. Ce choix accélère le développement (une seule base de routes, moins de boilerplate) tout en gardant une UX moderne (navigation fluide, formulaires validés côté serveur et retour d'erreurs immédiat).

La carte utilise Leaflet pour afficher les activités proches (marqueurs, popups). Le tout est responsive (mobile-first) et versionné via Git.

## Back-end

Le back repose sur Laravel (PHP) pour sa rapidité de mise en œuvre, son ORM Eloquent (relations Users/Activities/Bookings/Messages) et ses protections natives (CSRF, validation, middleware de rôles participant/organisateur/admin). Les migrations pilotent l'évolution du schéma et les seeders fournissent des données de démo.

Le déploiement est conteneurisé avec Docker (nginx + php-fpm + db), ce qui garantit un environnement reproductible et simplifie la mise en production.

## Accessibilité

### Types d'appareils compatibles

L'application MeetTrip a été pensée pour offrir une expérience fluide et accessible sur tous les types d'appareils : du smartphone au grand écran d'ordinateur.

L'objectif est de permettre à chaque utilisateur qu'il soit voyageur, organisateur ou administrateur de naviguer confortablement, quelle que soit sa configuration.

#### Une stratégie responsive maîtrisée

J'ai adopté une approche mobile-first, en m'appuyant sur le framework Tailwind CSS pour gérer les points de rupture via des classes utilitaires.

Par exemple, les classes md:hidden ou lg:flex permettent d'adapter la mise en page selon la largeur de l'écran sans écrire de CSS complexe.

Cette méthode garantit une adaptation fluide et cohérente sur l'ensemble des supports :

- Smartphones (320px à 767px) : navigation simplifiée avec menu condensé et zones tactiles élargies.
- Tablettes (768px à 1023px) : affichage équilibré avec grilles souples et typographie adaptée à la lecture.
- Ordinateurs (1024px et plus) : mise en page complète avec sections aérées et hiérarchie visuelle claire.

(Voir Annexe 5 – *Design responsive et accessibilité de la page “Activités” du projet MeetTrip*)

### Respect des standards WCAG et WAI

MeetTrip suit les recommandations WCAG 2.1 (niveau AA) et les principes WAI-ARIA afin de garantir une accessibilité optimale à tous les profils d'utilisateurs.

Chaque page utilise une structure HTML sémantique avec des balises appropriées

(<header>, <nav>, <main>, <footer>) pour assurer une lecture claire par les lecteurs d'écran et améliorer le référencement naturel.

Les formulaires incluent des labels explicites, des messages d'erreur contextualisés et un ordre de tabulation cohérent.

Les éléments interactifs (boutons, menus, modales) possèdent des attributs ARIA comme aria-label, aria-expanded ou aria-hidden, afin d'être correctement interprétés par les technologies d'assistance.

De plus :

- Les images disposent d'alternatives textuelles (alt).
- La carte Leaflet est accompagnée d'une vue liste pour garantir une alternative non visuelle.
- Les animations respectent les préférences utilisateur (prefers-reduced-motion).

```
<img  
  src={rencontrEtLogo}  
  alt="Rencontre et logo"  
  className="rounded-lg shadow-md object-cover transform hover:scale-105 transition duration-500 ease-in-out"  
>
```

Exemple de balise image comportant un attribut alt, permettant une lecture alternative pour les lecteurs d'écran (conformité WCAG 2.1).

## Contrôle et validation

L'accessibilité a été testée à chaque itération du développement sur plusieurs résolutions et navigateurs.

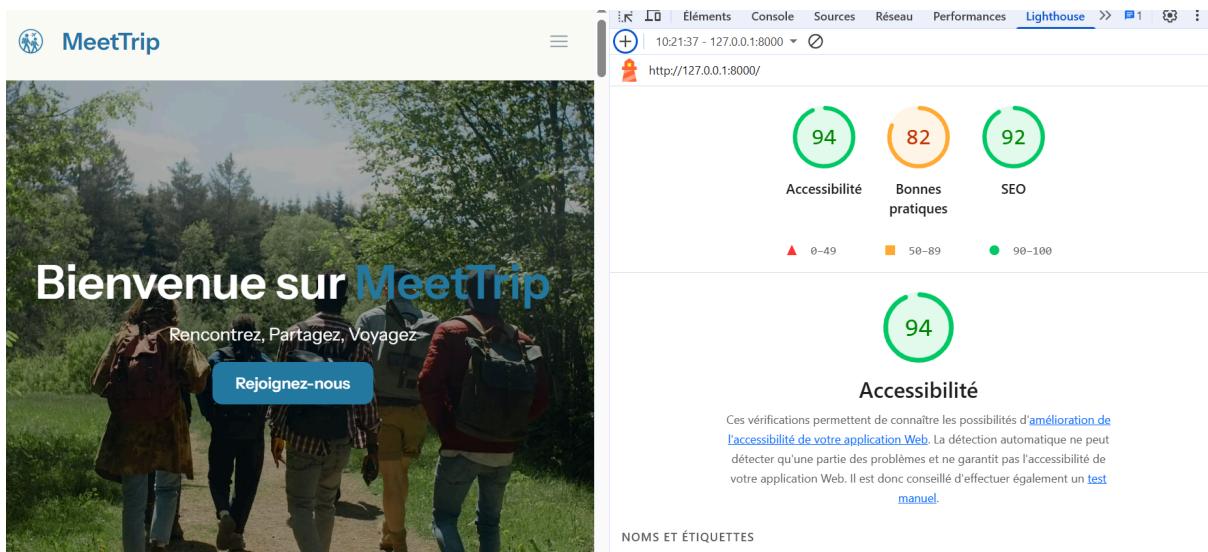
J'ai également réalisé un audit avec Google Lighthouse et les outils de développement intégrés de Chrome et Firefox, afin d'identifier et corriger les éventuels problèmes d'accessibilité.

Ces tests ont permis de vérifier la conformité aux bonnes pratiques :

- Contrastes suffisants
- Navigation au clavier fonctionnelle (focus visible, lien "Aller au contenu")
- Lecture fluide par lecteur d'écran

(Voir Annexe 6 – Tableau comparatif des critères RGAA avant et après correction sur le projet MeetTrip)

(Voir Annexe 7 – Tableau d'application des critères RGAA/WCAG dans le projet MeetTrip)  
( Voir Annexe 24 — Performances et optimisation (audit Lighthouse)



L'application atteint un score d'accessibilité élevé, témoignant d'une interface claire, inclusive et adaptée à tous les utilisateurs.

## Architecture du projet

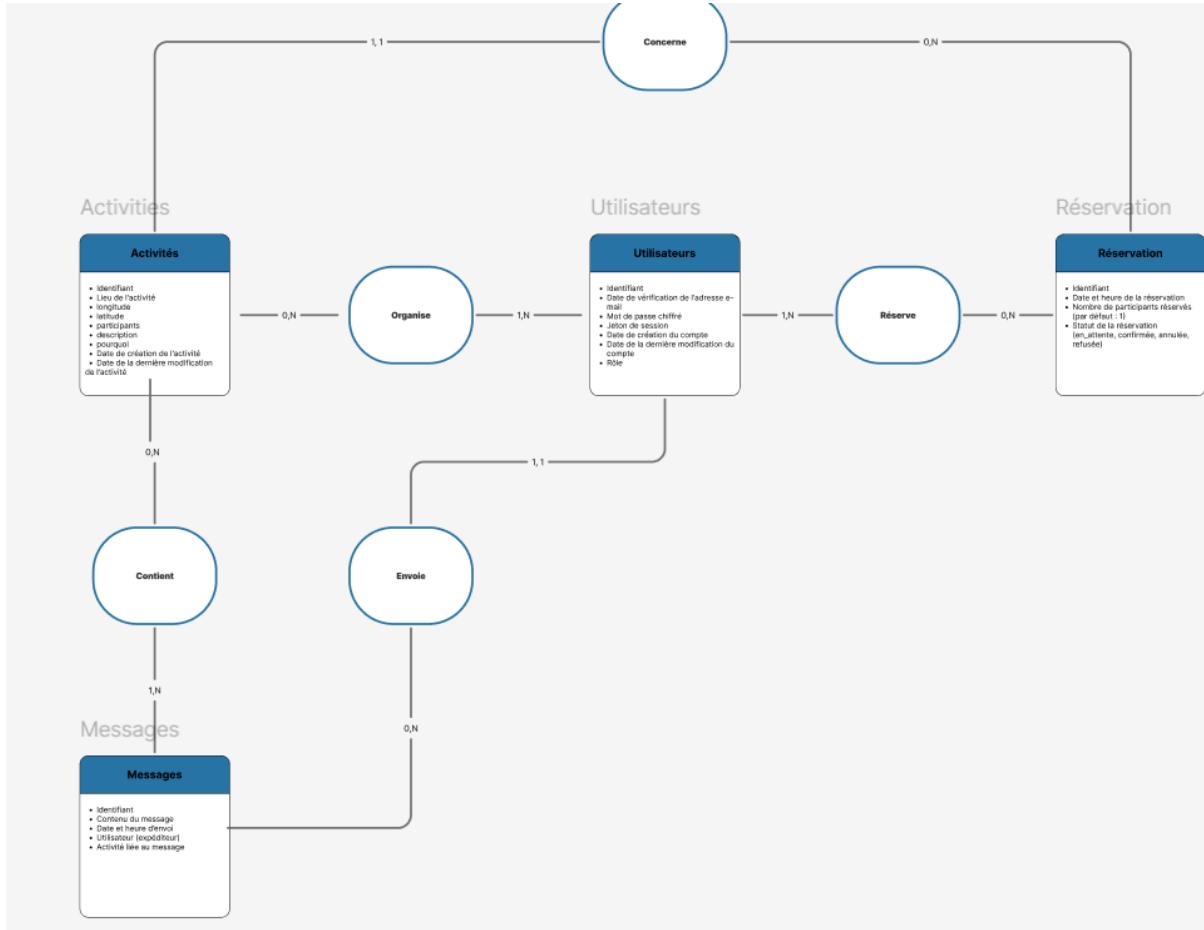
### MCD (Modèle Conceptuel de Données)

Le Modèle Conceptuel de Données (MCD) du projet MeetTrip permet de représenter les entités principales du système et leurs relations.

Ce modèle illustre comment les utilisateurs, les activités, les réservations et les messages interagissent entre eux.

- Un utilisateur peut organiser plusieurs activités.
- Un participant peut réserver plusieurs activités (relation N-N).
- Une activité peut contenir plusieurs messages échangés entre les utilisateurs concernés.
- Les messages sont toujours liés à une activité et à un utilisateur expéditeur.

Ce MCD constitue la base de la modélisation de la base de données relationnelle utilisée dans l'application.

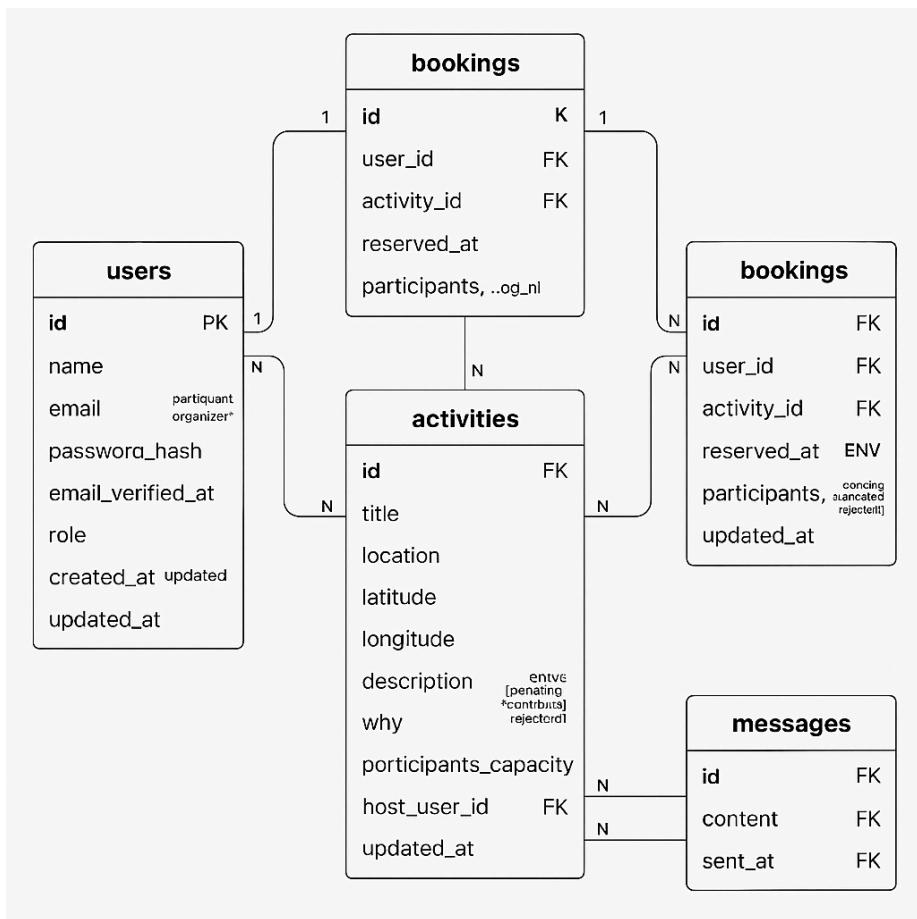


## MLD (Modèle Logique de Données)

À partir du MCD, j'ai établi le Modèle Logique de Données (MLD), qui précise la structure concrète des tables et les relations entre elles.

Le MLD définit notamment les clés primaires, clés étrangères et cardinalités nécessaires à la cohérence des données.

Ce modèle a ensuite servi à générer les migrations Laravel afin de créer automatiquement les tables dans la base de données.



## MPD (Modèle Physique de Données)

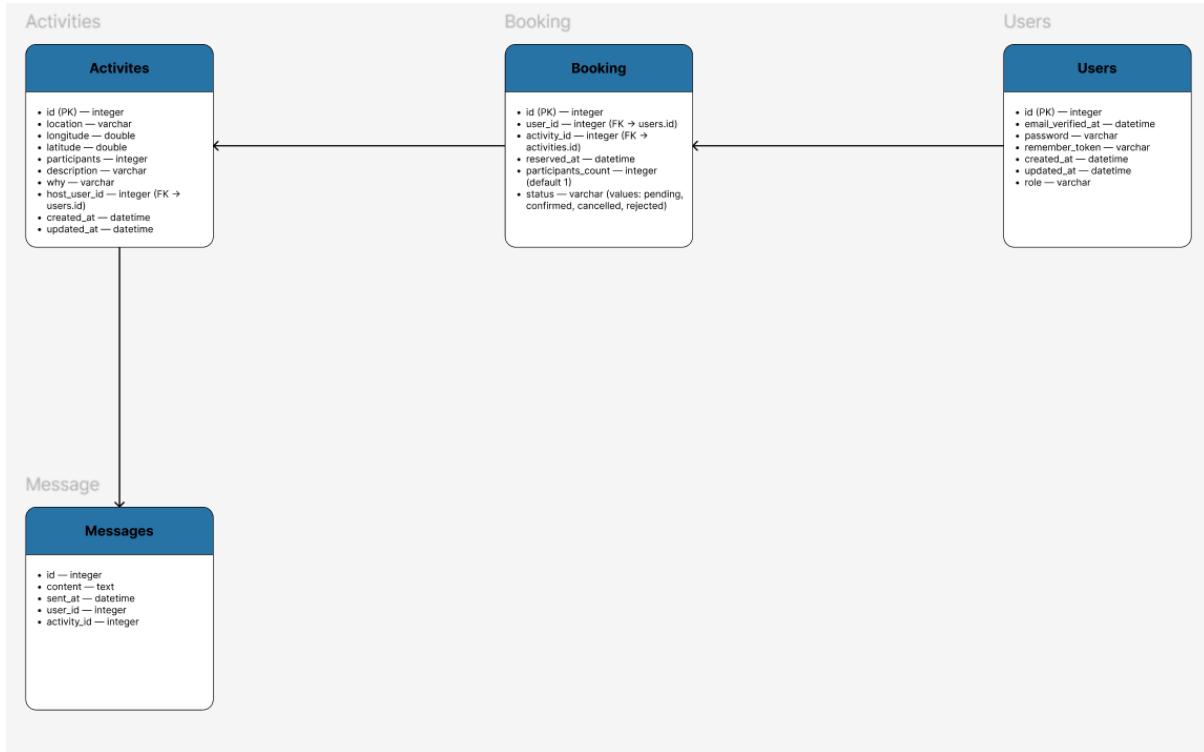
Le Modèle Physique de Données (MPD) représente la structure finale de la base, adaptée au SGBD utilisé dans le projet (SQLite en développement, MySQL pour le déploiement).

Les types de champs ont été choisis selon la nature des données :

- VARCHAR pour les textes courts (ex. : description, lieu)
- TEXT pour les messages
- DATETIME pour les champs de création et de réservation
- INTEGER pour les clés et relations
- DOUBLE pour les coordonnées géographiques (latitude, longitude)

Les clés primaires et étrangères assurent l'intégrité référentielle entre les entités.

Ce modèle final est directement exploité par Eloquent ORM dans Laravel.



## Use Case

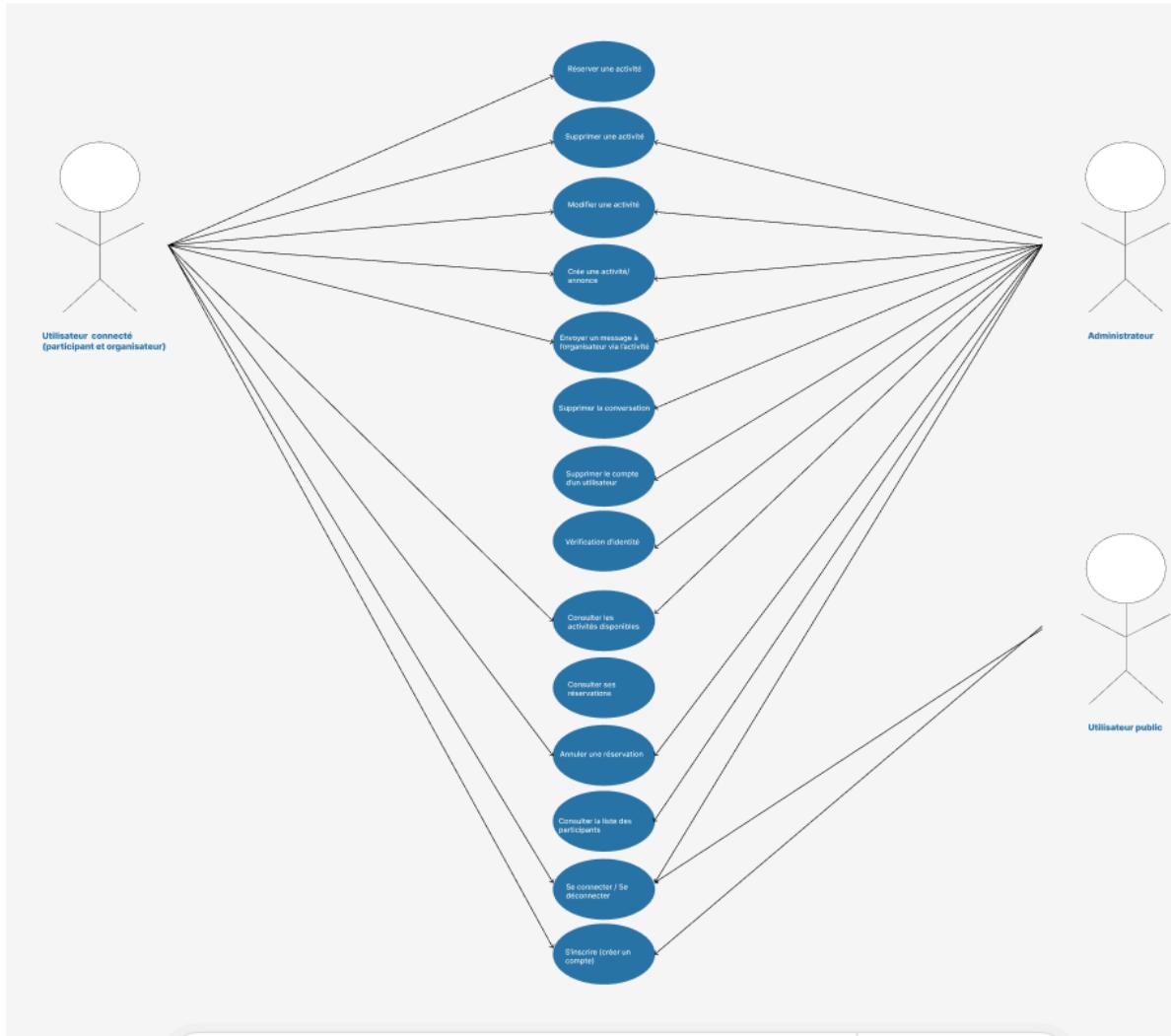
Le diagramme de cas d'utilisation décrit les principales interactions entre les différents acteurs du système.

Trois profils d'utilisateurs interagissent avec l'application :

- Utilisateur public : peut consulter les activités disponibles sans être connecté.
- Utilisateur connecté (participant / organisateur) : peut créer ou réserver une activité, échanger des messages et gérer son profil.
- Administrateur : supervise la plateforme, valide les identités, supprime les activités ou comptes non conformes.

Les principales actions représentées sont :

- Réserver, créer, modifier ou supprimer une activité
- Envoyer un message lié à une activité
- Annuler une réservation
- Consulter les participants
- Vérifier les identités et gérer les utilisateurs



## Utilisation des Seeders

Pour faciliter le développement et les tests, j'ai mis en place des seeders Laravel permettant d'insérer automatiquement des données de démonstration dans la base.

Ces seeders créent des utilisateurs, des activités et des réservations factices, rendant possible la visualisation des différentes fonctionnalités dès les premières phases du projet.

Exemple :

- création automatique d'un utilisateur test ;
- génération d'une activité associée à cet utilisateur avec toutes ses informations (lieu, coordonnées, description, nombre de places, etc.).

Cette méthode permet de gagner du temps et d'assurer des tests fonctionnels réalistes à chaque exécution.

```

1 <?php
2
3 namespace Database\Seeders;
4
5 use Illuminate\Database\Seeder;
6 use App\Models\Activities;
7
8 class ActivitySeeder extends Seeder
9 {
10     public function run(): void
11     {
12         $activities = [
13             [
14                 'title' => 'Visite de Rome',
15                 'location' => 'Rome, Italie',
16                 'image' => 'activities/rome.png',
17                 'description' => "Plongez dans l'histoire antique de Rome avec une visite guidée du Colisée et du Forum Romain.",
18                 'why' => "Pour découvrir les trésors de l'Empire romain et marcher sur les pas des empereurs.",
19                 'host_user_id' => 1,
20             ],
21             [
22                 'title' => 'Excursion bateau',
23                 'location' => 'Cannes, France',
24                 'image' => 'activities/excursionbateau.png',
25                 'description' => "Naviguez au coucher du soleil le long de la Côte d'Azur.",
26                 'why' => "Une parenthèse détente avec vue sur les plus belles criques.",
27                 'host_user_id' => 1,
28             ],
29             [

```

## Documentation des routes

L'organisation des routes dans MeetTrip repose sur une structure claire, séparant les espaces public, utilisateur connecté et administrateur, tout en sécurisant l'accès grâce aux middlewares auth, verified et rôle.

### ❖ Espace public

Les routes publiques permettent d'accéder aux pages d'information (/about, /contact, /cguy, /legal, /privacy) ainsi qu'au catalogue des activités via /activities et à leurs détails avec /activities/{id}.

Ces pages utilisent Inertia pour le rendu côté front.

### ❖ Espace utilisateur

Les utilisateurs authentifiés disposent d'un espace personnel accessible depuis /dashboard/client.

Ils peuvent créer, modifier ou supprimer leurs activités (/activities, /activities/{id}), consulter leurs réservations, accéder à la messagerie, à la carte interactive, et gérer leur profil.

Les vues sont gérées par Inertia et les données par Eloquent.

### ❖ Espace administrateur

Les administrateurs accèdent à un tableau de bord complet (/admin/dashboard) avec gestion des utilisateurs, des activités, vérification des identités et visualisation des statistiques globales.

Les routes sont protégées par le rôle admin et centralisées dans un groupe dédié.

En résumé

Les routes sont clairement structurées :

- Public : découverte et lecture du contenu.
  - Utilisateur : gestion d'activités et interactions.
  - Admin : supervision et contrôle de la plateforme.
- L'ensemble fonctionne avec Laravel, Inertia et une logique de rôles pour une navigation fluide et sécurisée.

```
// IMPORT DES CONTRÔLEURS
use App\Http\Controllers\ActivitiesController;
use App\Http\Controllers\Admin\AdminActivitiesController;
use App\Http\Controllers\ActivitiesConnectedController;
use App\Models\Activities;

// PAGES PUBLIQUES
Route::get('/', fn() => Inertia::render('PublicPages/Home'))->name('home');
Route::get('/about', fn() => Inertia::render('PublicPages/About'))->name('about');
Route::get('/contact', fn() => Inertia::render('PublicPages/Contact'))->name('contact');
Route::get('/cgu', fn() => Inertia::render('PublicPages/Cgu'))->name('cgu');
Route::get('/legal', fn() => Inertia::render('PublicPages/Legal'))->name('legal');
Route::get('/privacy', fn() => Inertia::render('PublicPages/Privacy'))->name('privacy');
Route::get('/activities', fn() => Inertia::render('PublicPages/Activities'))->name('activities');
Route::get('/activities/{id}', fn($id) => Inertia::render('PublicPages/Details-activities', ['id' => (int) $id]))->name('activity.details');

// UTILISATEUR CONNECTÉ
Route::middleware(['auth', 'verified', 'role:user'])->group(function () {
    Route::get('/dashboard/client', fn() => Inertia::render('Dashboard'))->name('user.dashboard');

    // ACTIVITÉS CONNECTÉES
    Route::get('/activitiesconnected', [ActivitiesConnectedController::class, 'index'])->name('activities.connected');
    Route::get('/activities/{id}/connected', [ActivitiesConnectedController::class, 'show'])->name('activity.details');
});

// ADMINISTRATEUR
Route::middleware(['auth', 'verified', 'role:admin'])->group(function () {
    Route::get('/admin/activities', [AdminActivitiesController::class, 'index'])->name('admin.activities');
    Route::post('/admin/activities', [AdminActivitiesController::class, 'store'])->name('admin.activities.store');
    Route::put('/admin/activities/{id}', [AdminActivitiesController::class, 'update'])->name('admin.activities.update');
    Route::delete('/admin/activities/{id}', [AdminActivitiesController::class, 'destroy'])->name('admin.activities.destroy');
});
```

## Sécurité

### 1. Authentification et gestion des rôles

La sécurité de l'application MeetTrip repose avant tout sur un système d'authentification robuste fourni par Laravel. Chaque utilisateur dispose d'un compte sécurisé et est associé à un rôle spécifique : participant, organisateur ou administrateur.

Ces rôles sont gérés via des middlewares Laravel, garantissant que seules les personnes autorisées accèdent à certaines sections (ex. : les organisateurs peuvent créer des activités, les administrateurs peuvent modérer ou supprimer du contenu).

Les mots de passe sont hachés, assurant qu'aucune donnée sensible n'est stockée en clair. Enfin, Laravel gère les sessions utilisateur de manière sécurisée, et met en œuvre un rate limiting pour limiter les tentatives de connexion frauduleuses.

## 2. Protection CSRF

Afin de prévenir les attaques de type Cross-Site Request Forgery (CSRF), chaque requête sensible (POST, PUT, PATCH, DELETE) inclut un jeton CSRF unique, généré et vérifié automatiquement par le middleware VerifyCsrfToken de Laravel.

Ce mécanisme empêche toute soumission de formulaire provenant d'un site tiers.

Les sessions sont régénérées après chaque connexion réussie afin de prévenir les détournements de session.

```
/*
 * 'http_only' => env('SESSION_HTTP_ONLY', true),
 */

/*
| -----
| Same-Site Cookies
| -----
|
| This option determines how your cookies behave when cross-site requests
| take place, and can be used to mitigate CSRF attacks. By default, we
| will set this value to "lax" to permit secure cross-site requests.
|
| See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie#samesitesamesite-value
|
| Supported: "lax", "strict", "none", null
|
*/
'same_site' => env('SESSION_SAME_SITE', 'lax'),

/*
| -----
| Description de la configuration
| -----
```

## 3. Sécurisation prod / CORS / sessions

### Politique CORS

L'application MeetTrip étant développée avec un front-end React connecté à un back-end Laravel, la configuration du CORS (Cross-Origin Resource Sharing) a été activée pour autoriser les échanges entre domaines lors du développement.

Les règles sont définies dans le fichier config/cors.php, où sont précisées :

- les origines autorisées (définies dans les variables d'environnement) ;
- les méthodes HTTP acceptées ;
- et la possibilité de partager les cookies et en-têtes lorsque cela est nécessaire.

Ce paramétrage permet de sécuriser les communications front/back tout en évitant les erreurs de politiques de même origine.

(Voir Annexe 8 – Configuration Laravel : paramètres “Same-Site Cookies” pour la protection CSRF)

(Voir Annexe 9 – Configuration Laravel : fichier config/cors.php pour la gestion des échanges front-end / back-end)

Cette configuration :

- Autorise uniquement les requêtes provenant du domaine `meettrip.fr`.
- Bloque tout échange entre domaines non autorisés.
- Permet l'envoi sécurisé des cookies d'authentification (`supports_credentials=true`).

Ainsi, la communication entre le front et le back se fait uniquement dans un cadre contrôlé et conforme aux politiques de même origine.

#### Politique de sécurisation en production

L'application MeetTrip intègre plusieurs mécanismes de durcissement destinés à renforcer la sécurité lors du déploiement en production.

Ces mesures garantissent la protection des données utilisateurs, la confidentialité des sessions et la limitation des échanges entre domaines.

#### 1. Sécurisation des cookies et des sessions

Laravel gère automatiquement les cookies de session de manière sécurisée grâce à la configuration du fichier `config/session.php`.

Les principales options activées sont les suivantes :

Ces paramètres garantissent :

- `secure=true` → les cookies sont transmis uniquement via HTTPS.
- `http_only=true` → le cookie n'est jamais accessible depuis JavaScript, prévenant les attaques XSS.
- `same_site=strict` → empêche l'envoi de cookies sur des requêtes cross-site, réduisant le risque de CSRF.

(Voir Annexe 10 – Sécurisation des cookies et des sessions)

(Voir Annexe 11 – Variables d'environnement de production : configuration sécurisée du fichier `.env`)

#### Durcissement et configuration HTTPS

L'environnement de production impose une redirection automatique vers HTTPS, garantissant une connexion chiffrée.

Le cookie de session est donc transmis uniquement via un canal sécurisé.

Extrait de configuration (`config/session.php`) :

```
'secure' => env('SESSION_SECURE_COOKIE'),
```

Laravel utilise également le middleware VerifyCsrfToken pour protéger tous les formulaires et requêtes POST contre les attaques de type CSRF.

Check-list sécurité production:

Élément de sécurité	Objectif	Statut
HTTPS activé	Chiffrement des échanges	<input checked="" type="checkbox"/>
Cookies HttpOnly, Secure, SameSite	Protection CSRF / XSS	<input checked="" type="checkbox"/>
CORS restreint à https://meettrip.fr	Limitation des origines	<input checked="" type="checkbox"/>
Taille d'upload limitée à 2 Mo	Prévention DoS	<input checked="" type="checkbox"/>
CSRF Token Laravel actif	Protection des formulaires	<input checked="" type="checkbox"/>

#### Référence REAC – Sécurité et conformité

La configuration appliquée respecte les recommandations de l'ANSSI concernant :

- la confidentialité des échanges (HTTPS)
- la limitation des origines (CORS)
- la protection des sessions (cookies sécurisés)
- la prévention des attaques CSRF et XSS

#### 4. Vérification d'identité et lutte contre les abus

Pour garantir la fiabilité des profils et prévenir les comportements abusifs, MeetTrip intègre plusieurs mesures de contrôle :

- Unicité et validation des e-mails côté serveur et côté client.
- Vérification d'identité obligatoire pour les organisateurs avant publication d'une activité.
- Système de signalement permettant de signaler une activité ou un message inapproprié, directement depuis l'interface utilisateur.
- Journalisation (logs) des actions sensibles telles que la connexion, la suppression d'activité ou les modifications de profil.
- Protection anti-spam sur les formulaires publics via ReCAPTCHA ou champ caché honeypot.

Ces dispositifs assurent un environnement sûr et convivial, favorisant la confiance entre les voyageurs.

## 5. Conformité RGPD

Enfin, MeetTrip respecte les principes de la RGPD (Règlement Général sur la Protection des Données).

- Minimisation des données : seules les informations strictement nécessaires sont collectées (profil, géolocalisation d'activités, messages liés à une activité).
- Transparence : une page dédiée à la politique de confidentialité explique clairement la gestion des données et les consentements (notifications, e-mails).
- Droits des utilisateurs : chaque membre peut demander la suppression ou l'export de ses données via l'administration.
- Sécurité et rétention : sauvegardes régulières, mises à jour du framework et purge automatique des comptes ou messages inactifs après une période définie.

(Voir Annexe 12 – Page “Politique de Confidentialité” intégrée au site MeetTrip)

### Etape 1 : Procédure d'accès, export et suppression des données

Dans MeetTrip, chaque utilisateur peut exercer ses droits RGPD directement depuis son espace personnel ou via l'administration :

#### 1. Accès aux données :

L'utilisateur connecté peut consulter les informations enregistrées dans son profil (nom, e-mail, activités créées ou réservées).

#### 2. Export :

Sur demande, un administrateur peut exporter ses données au format JSON ou CSV à l'aide d'une commande interne :

```
php artisan user:export {id}
```

Le fichier généré contient les informations de compte et les activités associées.

#### 3. Suppression :

L'utilisateur peut demander la suppression de son compte depuis le formulaire « Mon profil ». Côté administrateur, la commande suivante supprime toutes ses données liées :

```
php artisan user:delete {id}
```

Cette action déclenche également la suppression des activités ou messages associés (cascade onDelete('cascade')).

#### 4. Traçabilité :

Chaque suppression est enregistrée dans les logs de Laravel (storage/logs/laravel.log) avec l'ID de l'utilisateur concerné.

Cette procédure garantit la transparence et le respect des droits d'accès, d'opposition, de portabilité et d'effacement prévus par le RGPD.

## Etape 2 : Registre minimal des traitements

Type de données	Finalité du traitement	Base légale	Durée de conservation	Destinataires / Accès
Données de compte (nom, e-mail, mot de passe)	Création et gestion du profil utilisateur	Consentement de l'utilisateur	Suppression à la clôture du compte	Administrateurs <del>MeetTrip</del>
Activités créées / réservées	Organisation et participation à des voyages	Exécution d'un contrat	2 ans après inactivité	Organisateurs, admin
Messages internes	Communication entre utilisateurs	Intérêt légitime	1 an après la fin de l'activité	Participants et organisateurs concernés
Coordonnées (géolocalisation)	Affichage sur la carte Leaflet	Consentement explicite	Temps de l'utilisation active de l'application	Aucun transfert externe
Logs et connexions	Sécurité, débogage et prévention de fraude	Intérêt légitime	6 mois	Administrateurs techniques

Ces éléments démontrent la mise en œuvre concrète des obligations RGPD au sein de MeetTrip,  
en assurant la protection, la transparence et la maîtrise du cycle de vie des données personnelles.

## 6. Journalisation et observabilité

### Politique de logs

Pour assurer la traçabilité et la supervision du projet MeetTrip, une politique de journalisation claire a été mise en place.

Laravel enregistre automatiquement les événements critiques dans le fichier storage/logs/laravel.log, avec différents niveaux de gravité.

Niveau	Description	Exemples dans MeetTrip	Durée de conservation
<b>INFO</b>	Suivi normal du fonctionnement	Connexion d'un utilisateur, création d'activité, suppression de compte	6 mois
<b>WARNING</b>	Anomalie mineure non bloquante	Tentative de connexion échouée, validation de formulaire incomplète	3 mois
<b>ERROR</b>	Échec d'une action importante	Échec d'envoi d'e-mail, problème BDD, API Leaflet indisponible	1 an
<b>CRITICAL</b>	Dysfonctionnement majeur	Erreur fatale du serveur, crash application	1 an

Tous les logs sont stockés localement pendant le développement (storage/logs/laravel.log) et sur le serveur en production (/var/www/meettrip/storage/logs/laravel.log), avec rotation automatique hebdomadaire.

Une sauvegarde des logs critiques est également exportée vers un espace sécurisé lors des sauvegardes quotidiennes.

### Exemples anonymisés de logs

Ci-dessous, trois extraits simulés issus du fichier de logs Laravel :

```
[2025-06-04 14:22:08] local.INFO: User #42 connected successfully. {"ip":"192.168.0.23"}  
[2025-06-04 14:28:10] local.WARNING: Invalid password attempt for user #42  
[2025-06-04 14:35:27] local.ERROR: Activity deletion failed  
{"activity_id":12,"user_id":42,"exception":"SQLSTATE[HY000]"}
```

Ces extraits montrent la traçabilité des actions sensibles :

- connexions et déconnexions d'utilisateurs,
- tentatives de connexion échouées,
- suppressions d'activités ou erreurs critiques.

### Procédure de consultation des logs

Environnement	Méthode d'accès	Outil / Commande	Fréquence de vérification
Développement (local)	Lecture directe du fichier	php artisan tail ou ouverture via VS Code	Quotidienne lors des tests
Préproduction / Production	Accès SSH restreint (admin uniquement)	sudo cat /var/www/meettrip/storage/logs/laravel.log	Hebdomadaire + après chaque déploiement

Les logs sensibles (identifiants, e-mails, IP) sont anonymisés avant archivage ou export, afin de respecter les principes RGPD de confidentialité et de minimisation des données.

Cette stratégie de journalisation garantit la détection rapide des incidents, la traçabilité des actions utilisateurs et le respect des bonnes pratiques de sécurité applicative.

## 7. Gestion des erreurs et UX d'échec

### Politique de gestion des erreurs

L'application MeetTrip intègre une stratégie claire pour gérer les erreurs côté backend (Laravel) et frontend (React/Inertia.js) afin de garantir une bonne expérience utilisateur,

même en cas d'échec d'action.

Chaque erreur est :

- interceptée et affichée sous forme de message lisible pour l'utilisateur ;
- associée à un code HTTP standard (400, 401, 404, 500, etc.) ;
- accompagnée d'une action de repli (réessai, redirection, contact support).

Laravel capture automatiquement les exceptions dans app/Exceptions/Handler.php, tandis que React affiche des messages d'erreur contextualisés via des composants d'alerte.

Tableau de scénarios d'échec :

Scénario d'échec	Message utilisateur affiché	Code HTTP	Action proposée / Repli
Connexion échouée (mauvais identifiants)	"Email ou mot de passe incorrect."	401 Unauthorized	Suggérer de réinitialiser le mot de passe
Formulaire incomplet (champs requis manquants)	"Merci de remplir tous les champs obligatoires."	422 Unprocessable Entity	Afficher les champs manquants en rouge
Accès à une page sans autorisation (admin uniquement)	"Accès refusé. Vous n'avez pas les droits nécessaires."	403 Forbidden	Redirection vers le tableau de bord utilisateur
Ressource inexistante (activité supprimée)	"Cette activité n'existe plus."	404 Not Found	Retour automatique à la page "Toutes les activités"
Erreur serveur (ex : base de données)	"Une erreur inattendue s'est produite. Veuillez réessayer plus tard."	500 Internal Server Error	Journalisation dans les logs + message d'alerte visuel
Échec de l'envoi d'un message ou upload	"L'envoi a échoué. Vérifiez votre connexion Internet."	408 Request Timeout	Bouton "Réessayer" visible dans l'interface

Exemples d'erreurs contrôlées

1. Validation formulaire (React/Laravel)

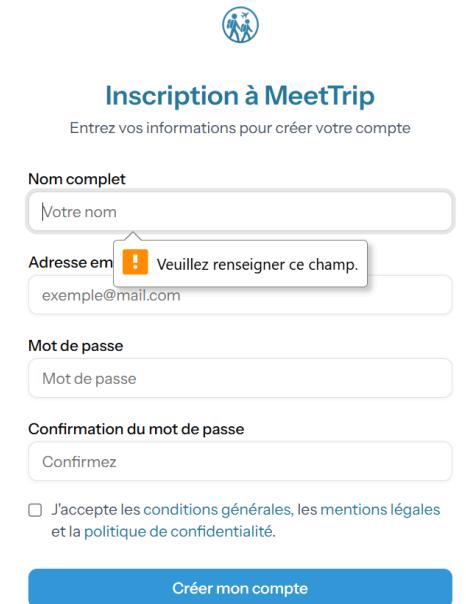
- Lorsqu'un utilisateur soumet un formulaire d'activité vide, le backend retourne un code 422.
- Le front affiche un message d'erreur sous le champ concerné : "Ce champ est obligatoire."
- Les champs invalides sont mis en surbrillance rouge.

2. Action interdite (admin uniquement)

- Si un utilisateur standard tente d'accéder à /admin/activities, une erreur 403 Forbidden est renvoyée.
- Une page d'erreur dédiée s'affiche avec le message : "Accès non autorisé — vous n'avez pas les droits suffisants."
- Un lien de retour vers la page d'accueil est proposé.

Cette stratégie de gestion d'erreurs assure une expérience utilisateur fluide, une communication claire en cas d'échec et une traçabilité complète via les logs Laravel pour analyse et amélioration continue.

Erreur de validation champ obligatoire non renseigné lors de l'inscription. Le navigateur bloque l'envoi du formulaire et affiche un message d'avertissement.

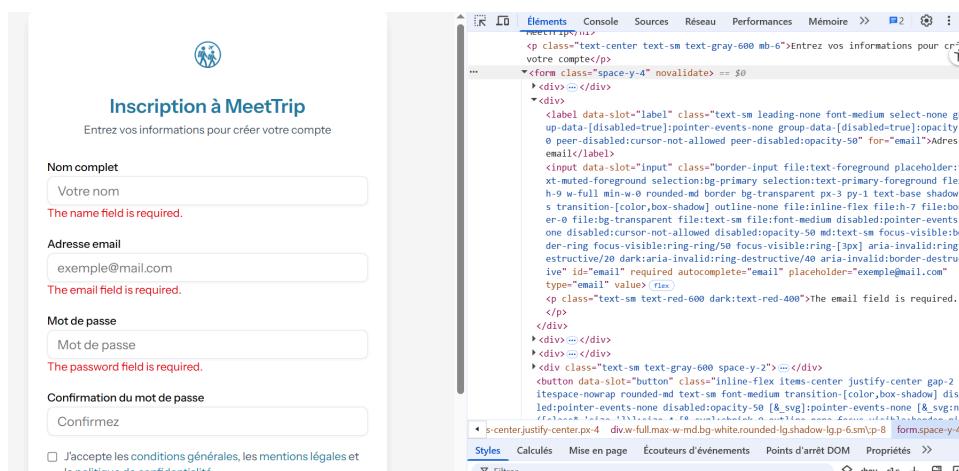


The screenshot shows a registration form titled "Inscription à MeetTrip". The form fields include "Nom complet" (Name), "Adresse email" (Email address), "Mot de passe" (Password), and "Confirmation du mot de passe" (Confirm password). Below the email field, there is an error message: "Veuillez renseigner ce champ." (Please fill this field) with an exclamation mark icon. A placeholder "exemple@mail.com" is visible in the email input field. At the bottom, there is a checkbox for accepting terms and conditions and a blue "Créer mon compte" (Create my account) button.

Lors des tests de validation de formulaire, les champs obligatoires vides déclenchent un message d'erreur contrôlé côté back-end (code HTTP 422).

Ces messages s'affichent de manière claire sous les champs concernés, guidant l'utilisateur dans la correction des erreurs.

Exemple ci-dessous : tentative d'inscription sans renseigner les champs obligatoires.



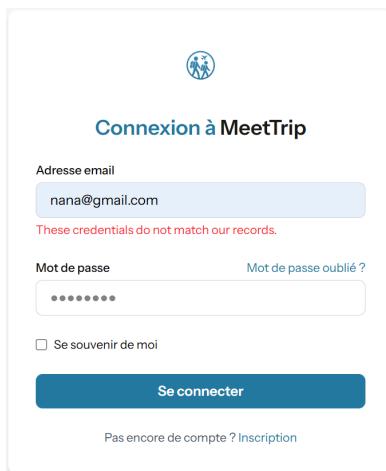
The screenshot shows the same registration form with developer tools open in the browser. The "Elements" tab of the developer tools interface is visible, displaying the HTML structure of the page. Under the "Elements" tab, the error messages are clearly visible as text nodes within the corresponding input fields: "The name field is required.", "The email field is required.", and "The password field is required.". The developer tools also show the CSS classes and IDs used for styling the form elements.

Lors d'une tentative de connexion avec des identifiants invalides, l'application affiche un message d'erreur contrôlé :

“These credentials do not match our records.”

Ce retour est géré par le système d'authentification Laravel, qui renvoie un code HTTP 401, sans indiquer si l'adresse e-mail ou le mot de passe est incorrecte, afin d'éviter les fuites d'informations.

L'utilisateur est invité à vérifier ses identifiants ou réinitialiser son mot de passe via le lien “Mot de passe oublié ?”.



The screenshot shows the MeetTrip login page. At the top, there is a logo of two stylized figures. Below it, the heading "Connexion à MeetTrip" is displayed. The form contains fields for "Adresse email" (nana@gmail.com) and "Mot de passe" (redacted). A message "These credentials do not match our records." is shown below the password field. There is also a link "Mot de passe oublié ?". A checkbox for "Se souvenir de moi" is present, and a "Se connecter" button is at the bottom. At the very bottom, there is a link "Pas encore de compte ? Inscription".

On the right side of the screenshot, a detailed view of the HTML code for the error message is shown. The code includes a script tag with a "send" function for sending verification notifications, and a main div structure for the login form. The error message itself is contained within a paragraph tag inside a form element, with a class "text-sm text-red-500 mt-1". The entire screenshot is framed by a light gray border.

(Voir Annexe 13 – Tableau des scénarios d'échec et des actions correctives prévues dans MeetTrip)

## 8. SEO / Publication

### Optimisation du référencement (SEO)

L'application MeetTrip dispose d'une partie vitrine publique (pages Accueil, À propos, Contact...) destinée à être indexée par les moteurs de recherche.

Un travail d'optimisation SEO a été réalisé afin d'améliorer la visibilité du site et la compréhension du contenu par les moteurs d'indexation.

#### 1. Métadonnées dans le <head> / Structure sémantique (H1–H2)

Chaque page publique contient un titre unique et une description optimisée pour le référencement naturel.

Exemple de page avec balises SEO ajoutées dans le <head> d'une page publique via le composant <Head> d'Inertia.js. :

```
<Head>
  <title>MeetTrip Découvrez le monde autrement</title>
  <meta
    name="description"
    content="MeetTrip est la plateforme qui connecte les voyageurs du monde entier pour partager activit
  />
  <meta
```

Ces balises définissent le titre, la description et les métadonnées pour le référencement et le partage sur les réseaux sociaux.

(Voir Annexe 14 – Rendu de la page d'accueil et intégration SEO du site MeetTrip)

Structure sémantique claire (H1 → H2) utilisée sur les pages publiques pour optimiser le référencement naturel (SEO) et l'accessibilité.

La page publique de démonstration “SeoExample.jsx” montre l'intégration des métadonnées (title, meta description, Open Graph, Twitter Card) et une structure de titres hiérarchisée. Ces éléments garantissent une bonne indexation sur les moteurs de recherche et une présentation cohérente lors du partage sur les réseaux sociaux.

## 2. Fichier robots.txt

(Voir Annexe 15 – Fichier robots.txt du site MeetTrip)

Le fichier robots.txt précise aux moteurs de recherche quelles sections du site peuvent être indexées.

Il bloque l'accès aux pages d'administration et d'authentification et référence le plan du site sitemap.xml pour optimiser l'exploration.

Ce fichier XML liste les principales pages publiques du site MeetTrip (accueil, activités, contact, CGU, etc.).

## 3. Fichier sitemap.xml

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9" xmlns:xhtml="http://www.w3.org/1999/xhtml">
  <!-- Page d'accueil -->
  <url>
    <loc>https://meettrip.fr/</loc>
    <lastmod>2025-10-09</lastmod>
    <changefreq>weekly</changefreq>
    <priority>1.0</priority>
  </url>
  <!-- Pages publiques -->
  <url>
    <loc>https://meettrip.fr/about</loc>
    <lastmod>2025-10-09</lastmod>
    <changefreq>monthly</changefreq>
    <priority>0.8</priority>
  </url>
  <url>
    <loc>https://meettrip.fr/contact</loc>
    <lastmod>2025-10-09</lastmod>
    <changefreq>monthly</changefreq>
    <priority>0.8</priority>
  </url>
  <url>
    <loc>https://meettrip.fr/activities</loc>
    <lastmod>2025-10-09</lastmod>
    <changefreq>weekly</changefreq>
    <priority>0.9</priority>
  </url>
  <url>
    <loc>https://meettrip.fr/cgu</loc>
    <lastmod>2025-10-09</lastmod>
    <changefreq>yearly</changefreq>
    <priority>0.5</priority>
  </url>
</urlset>
```

Il indique leur fréquence de mise à jour et leur priorité d'indexation afin d'aider les moteurs de recherche à explorer le site plus efficacement.

Exemple visible sur <http://127.0.0.1:8000/sitemap.xml>

(Voir Annexe 16 – Tableau d'optimisation SEO et publication du site MeetTrip)

L'optimisation SEO de MeetTrip repose sur la structuration sémantique (balises H1–H2, meta description) et sur deux fichiers de référence :

robots.txt pour le contrôle d'indexation et sitemap.xml pour le référencement des pages publiques.

Ces éléments permettent une indexation efficace et conforme aux bonnes pratiques SEO.

## Tests

### 1. Tests unitaires

Laravel offre une structure claire pour l'organisation des tests, avec deux types principaux : Unit (tests unitaires) et Feature (tests fonctionnels).

Les tests unitaires permettent de vérifier le bon fonctionnement d'une partie isolée du code, comme une fonction ou une méthode, sans exécuter tout le framework.

Dans MeetTrip, ces tests sont utilisés pour s'assurer que les éléments essentiels (par exemple, la création ou la suppression d'activités, la validation des données d'un formulaire, ou le calcul du nombre de participants) fonctionnent indépendamment du reste de l'application.

Ils garantissent la fiabilité du code et permettent d'identifier rapidement les erreurs internes avant de passer aux tests plus globaux.

```
C:\Users\Utilisateur\Documents\meettrip>php artisan test --filter=ExampleTest

PASS Tests\Unit\ExampleTest
✓ that true is true

Tests: 1 passed (1 assertions)
Duration: 1.00s
```

Capture d'exécution des tests unitaires dans le terminal.

Commande exécutée : php artisan test --filter=ExampleTest

Tous les tests passent avec succès, indiquant le bon fonctionnement de la base du code Laravel.

## 2. Tests fonctionnels

Les tests fonctionnels vérifient le comportement global de l'application dans un contexte réaliste.

Ils testent les routes Laravel, les middlewares, et les interactions avec la base de données. Dans MeetTrip, ces tests ont permis de valider :

- L'inscription et la connexion d'un utilisateur.
- La création et la suppression d'une activité.
- L'affichage correct des activités sur la carte Leaflet.
- L'envoi et la réception de messages entre participants et organisateurs.

Ces tests garantissent que toutes les couches du projet (contrôleurs, modèles, vues) fonctionnent correctement ensemble.

```
C:\Users\Utilisateur\Documents\meettrip>php artisan test

PASS Tests\Unit\ExampleTest
✓ that true is true 0.40

FAIL Tests\Feature\Auth\AuthenticationTest
✓ login screen can be rendered 0.12
✗ users can authenticate using the login screen 0.11
✓ users can not authenticate with invalid password 0.24
✓ users can logout 0.03

FAIL Tests\Feature\Auth>EmailVerificationTest
✓ email verification screen can be rendered 0.03
✗ email can be verified 0.83
✓ email is not verified with invalid hash 0.04
```

Capture d'exécution des tests fonctionnels avec la commande php artisan test.

Ces tests vérifient le bon fonctionnement global de l'application MeetTrip (authentification, validation des formulaires, interactions utilisateurs et activités).

Les résultats montrent le déclenchement des différents tests Laravel couvrant routes, contrôleurs et modèles.

### **3. Cahier de tests**

Afin de structurer la phase de validation, un cahier de tests a été mis en place.

Il regroupe toutes les fonctionnalités testées, les étapes de vérification, les données de test, les résultats attendus et le statut final.

Ce document a servi à valider chaque étape du développement (authentification, messagerie, réservations, etc.) avant la mise en production.

Le cahier de tests a également permis :

- De tracer les vérifications effectuées.
- D'assurer la non-régression des fonctionnalités.
- De garantir la fiabilité et la cohérence de l'application.

(Voir Annexe 17 – *Cahier de tests fonctionnels du projet MeetTrip*)

Avant l'exécution des tests, un jeu d'essai automatisé a été mis en place à l'aide du seeder ActivitySeeder.php.

Ce script insère automatiquement un utilisateur de test et une activité dans la base de données.

Il permet de préparer un environnement de test cohérent pour valider les fonctionnalités de création, d'association et d'affichage des activités.

Cette méthode garantit la reproductibilité des tests et la fiabilité des résultats, sans dépendre de données saisies manuellement.

Exemple de code du seeder utilisé :

```

2
3 namespace Database\Seeders;
4
5 use App\Models\Activities;
6 use App\Models\Activity;
7 use Illuminate\Database\Seeder;
8 use App\Models\User;
9
10 class ActivitySeeder extends Seeder
11 {
12     public function run(): void
13     {
14
15         $user = User::firstOrCreate(
16             ['email' => 'test@example.com'],
17             ['name' => 'Test User', 'password' => bcrypt('password')]
18         );
19
20         Activities::create([
21             'title' => 'Randonnée dans les Alpes',
22             'location' => 'Chamonix',
23             'latitude' => 45.9237,
24             'longitude' => 6.8694,
25             'participants' => 5,
26             'description' => 'Une randonnée en montagne avec des vues incroyables.',
27             'why' => 'Rencontrer des passionnés de nature et prendre l'air.',
28             'image' => 'https://example.com/image.jpg',
29             'host_user_id' => $user->id,
30         ]);
31     }
}

```

Le tableau ci-dessous présente le jeu d'essai utilisé pour valider les principales fonctionnalités du projet MeetTrip à partir du seeder ActivitySeeder.php.

Ces tests couvrent la création et la gestion d'activités, l'authentification, la validation des formulaires et la cohérence des données affichées côté front.

Les résultats obtenus confirment que les fonctionnalités critiques de l'application fonctionnent correctement dans un environnement simulé.

(Voir Annexe 18 – Résultats détaillés des tests d'intégration du projet MeetTrip)

Ces essais confirment le bon fonctionnement global de l'application MeetTrip, aussi bien sur le plan du backend Laravel que sur l'affichage front-end Inertia/React

#### 4. Couverture et périmètre des tests

Les tests réalisés couvrent les principales fonctionnalités critiques de l'application MeetTrip.

- Authentification (unitaires + fonctionnels)
- Création et gestion d'activités (fonctionnels)
- Réservation d'activités (fonctionnels)
  
- Messagerie (fonctionnels)
- Affichage Leaflet (fonctionnels)
- Sécurité et redirections (unitaires)

La couverture estimée est d'environ 70 % du code essentiel, garantissant la fiabilité et la stabilité de l'application avant la mise en production.

# Gestion de projet

## 1. Approche

Pour le développement de MeetTrip, j'ai travaillé en autonomie complète, en gérant toutes les étapes du projet : analyse, conception, développement, test et déploiement. Cette approche m'a permis de garder une vision claire de l'avancement et d'assurer la cohérence entre les différentes parties du projet.

## 2. Méthode de travail

J'ai organisé mon travail avec la méthode Kanban, à l'aide de GitHub Projects.

Chaque tâche correspondait à une fonctionnalité (authentification, carte interactive, messagerie, tableau de bord...).

Cette méthode m'a aidée à planifier efficacement, à suivre la progression et à prioriser les éléments importants jusqu'à la livraison finale.

## 3. Analyse des besoins

Avant de coder, j'ai défini les besoins principaux des trois profils utilisateurs : voyageurs, organisateurs et administrateurs.

Cela m'a permis de déterminer les fonctionnalités essentielles : création et réservation d'activités, messagerie intégrée, carte interactive et gestion complète depuis un tableau de bord.

## 4. Conception

J'ai conçu la structure technique du projet avec un MCD/MLD clair, reliant utilisateurs, activités, réservations et messages.

Les maquettes ont été réalisées sur Figma, puis j'ai structuré le code avec Laravel, React et Inertia.js, en intégrant la gestion des rôles et les mesures de sécurité (CSRF, authentification, validation).

## 5. Développement

Le développement s'est fait par étapes : interface utilisateur (front), logique serveur (back) et base de données.

J'ai utilisé Laravel pour le back-end et React + Inertia.js pour le front-end, tout en assurant la communication fluide entre les deux.

Les commits réguliers sur GitHub m'ont permis de suivre l'évolution du projet et de conserver un historique propre et structuré.

## 6. Déploiement

Le déploiement a été effectué avec Docker pour reproduire un environnement de production complet. J'ai configuré plusieurs conteneurs (PHP, MySQL, Nginx, Node.js) et automatisé le lancement via Docker Compose.

Cette solution garantit une application stable, portable et facile à exécuter sur tout système, proche des conditions réelles d'hébergement.

### 1ere étape: Guide d'installation

Le déploiement de MeetTrip s'effectue à l'aide de Docker Compose.

Voici la procédure complète permettant de lancer le projet en local ou sur un serveur de préproduction :

1) Récupérer le projet

```
git clone <url_du_repo> && cd meettrip
```

2) Choisir l'environnement (ex : local)

```
cp .env.local .env
```

3) Lancer l'infra

```
docker compose up -d --build
```

4) Préparer l'app

```
docker compose exec app composer install
```

```
docker compose exec app php artisan key:generate
```

5) Base & données de test

```
docker compose exec app php artisan migrate --force --seed
```

6) Finitions

```
docker compose exec app php artisan storage:link
```

```
docker compose exec app php artisan optimize:clear
```

Ces commandes permettent de cloner le projet, lancer les conteneurs, installer les dépendances,

exécuter les migrations et peupler la base avec des données de test automatiques.

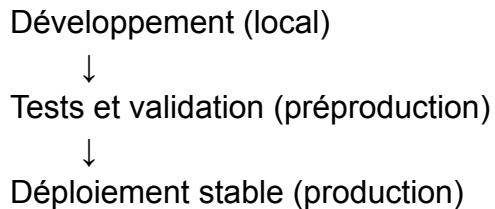
2e étape : Schéma des environnements et critères de validation :

Les environnements de développement, de test et de production ont été clairement définis afin d'assurer la fiabilité, la sécurité et la continuité du projet MeetTrip tout au long de son cycle de vie.

Le projet MeetTrip est prévu pour fonctionner sur trois environnements distincts :

(Voir Annexe 19 – Schéma des environnements et critères de validation du projet MeetTrip)

Schéma simplifié du cycle de déploiement :



Critères “Go / No-Go” :

Avant chaque déploiement, plusieurs points de contrôle garantissent la stabilité du projet :  
*(Voir Annexe 20 – Tableau des critères de validation “Go / No-Go” avant déploiement)*

Procédure de rollback (en cas d'échec du déploiement) :

1. Restaurer la version précédente :

```
git checkout main && git reset --hard <commit_stable>
```

2. Réappliquer la BDD stable :

```
docker compose exec db sh -c "mysql -umeettrip -psecret meettrip < /backups/meettrip_backup.sql"
```

3. Relancer l'application :

```
docker compose up -d --build
```

Ce processus de déploiement progressif assure la stabilité du projet MeetTrip, tout en permettant de tester, corriger et valider chaque version avant la mise en production finale.

## Conclusion

Le développement de l'application MeetTrip m'a permis de mettre en pratique l'ensemble des compétences acquises durant ma formation, aussi bien sur la partie front-end que back-end.

Cette plateforme de rencontres entre voyageurs a été pensée pour favoriser les échanges humains et simplifier l'organisation d'activités locales, tout en proposant une expérience fluide, responsive et sécurisée grâce à Laravel, React, Inertia.js et Docker.

Ce projet a été une véritable opportunité de suivre tout le cycle de vie d'une application web, depuis l'analyse des besoins jusqu'au déploiement.

J'y ai appliquéd mes connaissances en modélisation de base de données, conception d'interfaces, gestion des rôles utilisateurs et mise en place d'un environnement de production conteneurisé.

Chaque étape m'a permis d'approfondir ma maîtrise des technologies modernes et des bonnes pratiques de développement.

Au-delà des aspects techniques, ce projet m'a appris à travailler avec rigueur, à organiser mes priorités et à mener un projet complet de A à Z.

Elle représente une étape importante dans mon parcours professionnel et consolide mon objectif de devenir développeuse web fullstack, capable de concevoir et déployer des applications complètes et robustes

## Annexes

### Cahier des charges

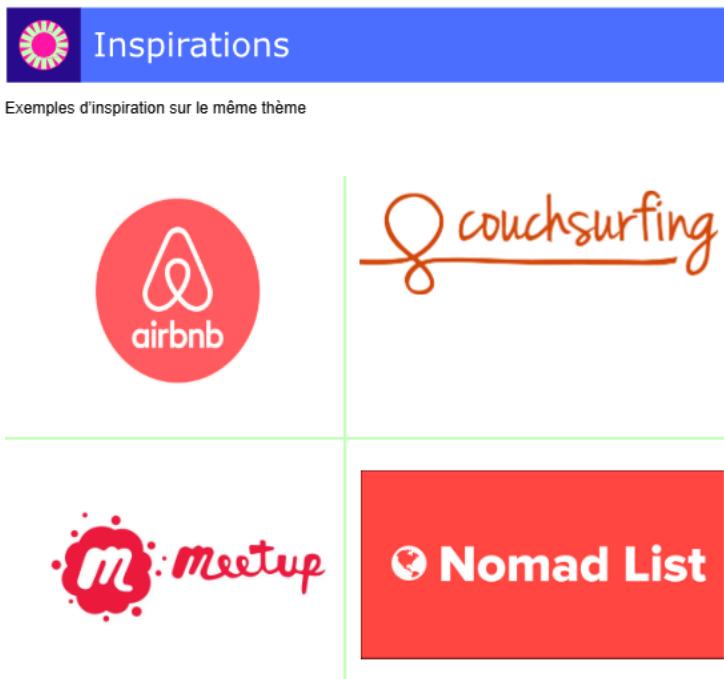
#### Annexe 1 : Personas – Profil des utilisateurs MeetTrip

##### Personas

	<p><b>Sofya, 28 ans – Voyageuse solo</b></p> <ul style="list-style-type: none"><li>• <b>Objectifs :</b> Découvrir des activités authentiques et rencontrer des habitants pendant ses séjours.</li><li>• <b>Besoins &amp; Attentes :</b> Une interface simple, un système de réservation fluide, et la garantie que les organisateurs sont fiables.</li></ul>
	<p><b>Julie, 35 ans – Organisatrice locale</b></p> <ul style="list-style-type: none"><li>• <b>Objectifs :</b> Proposer ses activités facilement, gérer les réservations et communiquer avec les participants.</li><li>• <b>Besoins &amp; Attentes :</b> Tableau de bord clair, messagerie intégrée, validation rapide des activités.</li></ul>
	<p><b>Carlos, 40 ans – Administrateur</b></p> <ul style="list-style-type: none"><li>• <b>Objectifs :</b> Superviser la plateforme, gérer les utilisateurs et valider les activités.</li><li>• <b>Besoins &amp; Attentes :</b> Outils de modération, accès aux statistiques et visibilité sur l'activité des comptes.</li></ul>

Extrait de la feuille de route présentant trois personas types : Sofya (voyageuse solo), Julie (organisatrice locale) et Carlos (administrateur). Chaque persona définit les objectifs, besoins et attentes pour orienter la conception du projet.

## Annexe 2 : Inspirations Analyse comparative de plateformes



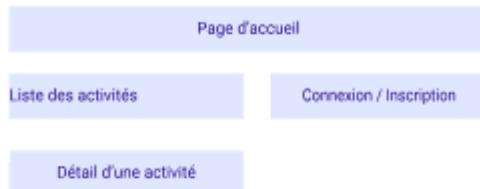
Ces plateformes m'ont inspirée pour l'aspect communautaire, intuitif et humain, tout en gardant une interface plus simple et claire.

Exemples de plateformes de référence (*Airbnb, Couchsurfing, Meetup, Nomad List*) ayant inspiré la conception de MeetTrip pour leur aspect communautaire, intuitif et humain.

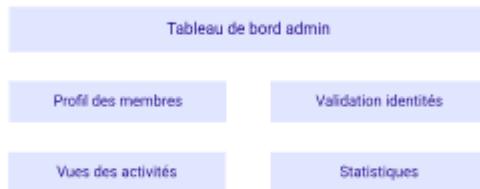
## Annexe 3 :Plan du site MeetTrip



### Espace non connecté



### Espace connecté administrateur



---

### Espace connecté participant/org

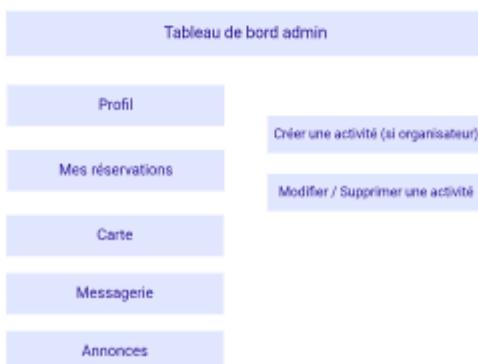


Schéma global de navigation

présentant la structure des trois espaces : utilisateur non connecté, utilisateur connecté (participant/organisateur) et administrateur. Permet de visualiser les parcours et fonctionnalités principales.

#### Annexe 4 : Choix d'architecture technique

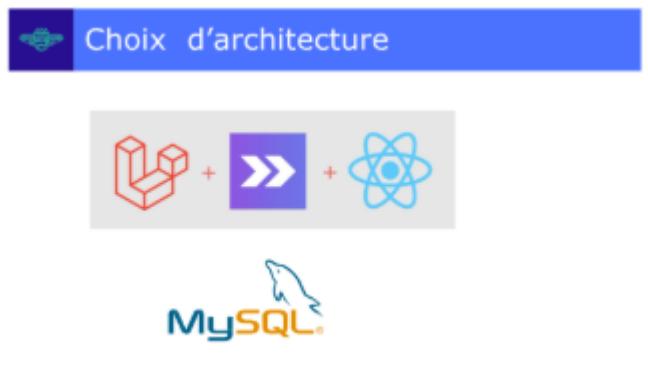


Illustration du stack technologique de *MeetTrip* : Laravel (back-end), Inertia.js (passerelle front/back), React (interface utilisateur) et MySQL (base de données).

#### Accesibilité

#### Annexe 5 : Design responsive et accessibilité – page “Activités”

The figure displays a responsive design for the 'Activités' page. On the left, a mobile phone shows a menu with 'Activités' and three items: 'Excursion en bateau à Nice', 'Randonnée à Marseille', and 'Visite de Lyon'. In the center, a tablet shows the same menu and the same three items. On the right, a desktop monitor shows a map with a blue pin and the same three activity titles. At the bottom left, there is a code snippet:

```

<button aria-label="Ouvrir le menu">
```

At the bottom right, there is another code snippet:

```
@media (prefers-contrast: more) {
  body {
    color: #000;
    background-color: #fff;
  }
}
```

Illustration du design responsive de la page „Activités“ –adaptation automatique de mise en page entre desktop, tablette et mobile grâce à Tailwind CSS et aux media

Exemple de code illustrant la mise en conformité avec les recommandations d'accessibilité (balises sémantiques, attributs ARIA, contraste élevé et mode)

Maquette et capture d'écran du design adaptatif (ordinateur, tablette, mobile) de la page *Activités* du projet *MeetTrip*. Inclut un extrait de code illustrant la conformité aux normes d'accessibilité (ARIA, contrastes, balises sémantiques).

## Annexe 6 : Tableau des critères RGAA – avant et après correction

Critère testé	Avant correction	Après correction
<b>Contraste des textes</b>	Le texte bleu sur fond clair n'avait pas un contraste suffisant sur certaines sections (score < 4.5).	Couleurs modifiées via Tailwind CSS (text-blue-800) → contraste conforme au WCAG 2.1 niveau AA ( $\geq 4.5$ ).
<b>Texte alternatif des images (alt)</b>	Certaines images décoratives (logo, photos d'arrière-plan) n'avaient pas d'attribut alt.	Tous les <code>&lt;img&gt;</code> ont un alt descriptif ou alt="" s'ils sont purement décoratifs.
<b>Structure des titres (h1, h2, h3)</b>	Plusieurs titres h1 sur la même page.	Hierarchie corrigée : un seul h1 par page, puis h2/h3 pour les sous-sections.
<b>Labels des champs de formulaire</b>	Certains champs n'étaient pas reliés à un label.	Ajout de <code>&lt;label for="email"&gt;Email&lt;/label&gt;</code> et association de tous les input à leur label.
<b>Navigation clavier / focus visible</b>	Le focus clavier n'était pas visible sur certains boutons personnalisés.	Ajout des classes Tailwind focus:outline-none et focus:ring → navigation clavier fluide et accessible.
<b>Attributs ARIA sur les modales</b>	Les fenêtres modales n'étaient pas reconnues par les lecteurs d'écran.	Ajout de role="dialog" et aria-modal="true" pour signaler les éléments dynamiques.
<b>Liens explicites</b>	Certains liens indiquent simplement "En savoir plus".	Les liens précisent désormais leur contexte, ex. "En savoir plus sur l'activité Rome".
<b>Langue du site (lang)</b>	Aucune langue définie dans la balise <code>&lt;html&gt;</code> .	Ajout de <code>&lt;html lang="fr"&gt;</code> pour définir la langue principale du site.
<b>Ordre du tabulateur</b>	Le bouton "Rejoignez-nous" est sauté dans l'ordre de tabulation.	Ajout de tabindex="0" et vérification complète du parcours clavier.
<b>Alternatives multimédia / texte descriptif</b>	Les images d'accueil n'avaient pas de texte descriptif associé.	Ajout de légendes textuelles décrivant la scène et son contexte ("Voyageurs en randonnée").

Tableau récapitulatif des principaux critères d'accessibilité (contrastes, structure des titres, attributs ARIA, navigation clavier, alternatives textuelles). Compare la conformité avant et après correction selon les recommandations WCAG 2.1 et RGAA.

## Annexe 7: Tableau d'application des critères RGAA / WCAG dans MeetTrip

Critère RGAA / WCAG	Exigence	Appliqué dans MeetTrip
<b>Images</b>	Fournir un texte alternatif pour chaque image porteuse d'information.	Tous les <code>&lt;img&gt;</code> comportent un alt (ex. logo, photo de groupe sur la page d'accueil).
<b>Structure des pages</b>	Utiliser des balises sémantiques pour la structure du contenu ( <code>&lt;header&gt;</code> , <code>&lt;main&gt;</code> , <code>&lt;footer&gt;</code> , etc.).	Présente sur toutes les pages React/Laravel via les composants principaux.
<b>Contraste des couleurs</b>	Le contraste texte/fond doit être $\geq 4.5:1$ (niveau AA).	Palette Tailwind ajustée (text-blue-800 sur fond clair). Vérifié avec Lighthouse.
<b>Navigation clavier</b>	Tous les éléments interactifs doivent être accessibles via la touche Tab.	Navigation fluide testée sur menus, boutons "Rejoignez-nous" et formulaires.
<b>Titre et hiérarchie</b>	Les titres doivent être organisés de manière logique (h1 → h2 → h3).	Structure hiérarchisée sur les pages "Accueil", "Toutes les activités", "Profil".
<b>Liens explicites</b>	Les liens doivent être compréhensibles hors contexte.	Exemples : "Découvrir Rome" au lieu de "En savoir plus".
<b>Langue du site</b>	Définir la langue principale dans le code source.	Balise <code>&lt;html lang="fr"&gt;</code> définie dans les vues inertia.js.
<b>Messages d'erreur</b>	Les formulaires doivent afficher des messages d'erreur clairs.	Validation Laravel → messages affichés sous les champs invalides.
<b>Compatibilité assistive</b>	Utiliser les attributs ARIA pour les éléments dynamiques.	Modales avec role="dialog" et aria-modal="true".
<b>Animations / mouvements</b>	Respecter les préférences utilisateur pour réduire les animations.	Utilisation de prefers-reduced-motion dans les transitions Tailwind.

Tableau présentant les exigences d'accessibilité (images, structure, contrastes, navigation clavier, ARIA, compatibilité assistive) et leur mise en œuvre concrète dans l'application *MeetTrip* (React / Laravel / Tailwind).

## Sécurité

### Annexe 8 : Configuration Laravel - “Same-Site Cookie”

```
/*
 * Same-Site Cookies
 *
 * This option determines how your cookies behave when cross-site requests
 * take place, and can be used to mitigate CSRF attacks. By default, we
 * will set this value to "lax" to permit secure cross-site requests.
 *
 * See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie#samesitesamesite-value
 *
 * Supported: "lax", "strict", "none", null
 */
 */

'session' => env('SESSION_HTTP_ONLY', true),

/*
 * Same-Site Cookies
 *
 * This option determines how your cookies behave when cross-site requests
 * take place, and can be used to mitigate CSRF attacks. By default, we
 * will set this value to "lax" to permit secure cross-site requests.
 *
 * See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie#samesitesamesite-value
 *
 * Supported: "lax", "strict", "none", null
 */
*/

'same_site' => env('SESSION_SAME_SITE', 'lax'),
```

Capture d'écran du fichier session.php illustrant la configuration des cookies sécurisés (paramètre Same-Site) contribuant à la protection CSRF.

### Annexe 9: Configuration Laravel – config/cors.php

```
config > cors.php
  1  return [
  2    /*
  3     | Configure here which origins, methods, and headers are allowed to
  4     | interact with your application. By default, we limit the origins
  5     | to those defined in the environment variable FRONT_URL.
  6     |
  7     */
  8
  9    'paths' => ['api/*', 'sanctum/csrf-cookie'],
 10
 11    'allowed_methods' => ['GET', 'POST', 'PUT', 'DELETE'],
 12
 13    'allowed_origins' => [env('FRONT_URL', 'http://localhost:5173')],
 14
 15    'allowed_headers' => ['Content-Type', 'X-Requested-With', 'Authorization'],
 16
 17    'exposed_headers' => [],
 18
 19    'max_age' => 0,
 20
 21    'supports_credentials' => true,
 22
 23  ];
 24
```

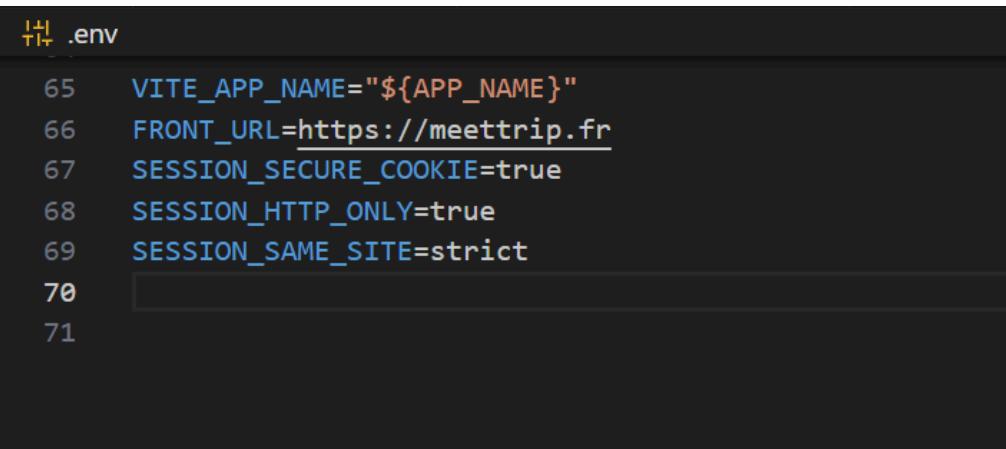
Capture d'écran du fichier cors.php définissant les origines autorisées, méthodes et en-têtes HTTP pour sécuriser les échanges entre le front-end React/Inertia.js et le back-end Laravel.

## Annexe10 : Sécurisation des cookies et des sessions

```
'secure' => env('SESSION_SECURE_COOKIE'),  
  
'http_only' => env('SESSION_HTTP_ONLY', true),  
  
'same_site' => env('SESSION_SAME_SITE', 'lax'),
```

Extrait de code du fichier session.php montrant les paramètres SESSION\_SECURE\_COOKIE, SESSION\_HTTP\_ONLY et SESSION\_SAME\_SITE, garantissant la confidentialité des sessions utilisateur.

## Annexe 11 : Variables d'environnement Laravel – .env :



```
+++.env  
65 VITE_APP_NAME="${APP_NAME}"  
66 FRONT_URL=https://meettrip.fr  
67 SESSION_SECURE_COOKIE=true  
68 SESSION_HTTP_ONLY=true  
69 SESSION_SAME_SITE=strict  
70  
71
```

Capture d'écran du fichier .env indiquant la configuration sécurisée de l'application (valeurs HTTPS, cookies sécurisés, sessions strictes).

## Annexe 12 : Page “Politique de Confidentialité” du site MeetTrip

The screenshot shows the footer of the MeetTrip website. At the top left is the MeetTrip logo. To its right are links for Accueil, Activités, À propos, and Contact. On the far right are buttons for Connexion (light blue) and Inscription (dark blue). Below these, the footer content begins with a section titled "Politique de Confidentialité".

**Politique de Confidentialité**

**1. Introduction**

Chez MeetTrip, la protection de votre vie privée est une priorité. Cette page vous informe sur la manière dont vos données personnelles sont collectées, utilisées et sécurisées.

**2. Données collectées**

Nous collectons uniquement les données nécessaires à l'utilisation de notre plateforme :

- Nom et prénom
- Adresse email
- Messages via le formulaire de contact
- Données de navigation (cookies)

**3. Utilisation des données**

Vos données sont utilisées uniquement pour :

- Créer et gérer votre compte
- Répondre à vos messages
- Améliorer nos services
- Vous informer sur les mises à jour importantes

**4. Sécurité**

MeetTrip met en place des mesures de sécurité pour protéger vos données contre tout accès non autorisé.

**5. Vos droits**

Conformément à la loi, vous pouvez demander l'accès, la modification ou la suppression de vos données à tout moment via notre formulaire de contact.

**6. Contact**

Pour toute question liée à cette politique de confidentialité, vous pouvez nous contacter via la page [Contact](#).

© 2025 MeetTrip. Tous droits réservés. [CGU](#) [Mentions légales](#) [Politique de confidentialité](#)

Capture d'écran de la page Politique de confidentialité intégrée dans le pied de page du site. Accessible depuis toutes les pages publiques (Accueil, Activités, À propos, Contact), elle informe les utilisateurs sur la collecte, la conservation et la protection de leurs données personnelles conformément au RGPD.

## Annexe 13 : Tableau des scénarios d'échec et des actions correctives prévues dans MeetTrip

Scénario d'échec	Message utilisateur affiché	Code HTTP	Action proposée
Champs obligatoires vides (inscription)	« Ce champ est obligatoire. » sous chaque input concerné	422	Surligner le champ, conserver les valeurs saisies, afficher un lien d'aide.
Email déjà utilisé	« Cet email est déjà associé à un compte. »	422	Proposer « Se connecter » ou « Réinitialiser le mot de passe ».
Mot de passe trop faible	« Mot de passe trop faible (8+ caractères, chiffres & lettres). »	422	Afficher les règles sous le champ + indicateur de force.
Confirmation ≠ mot de passe	« Les mots de passe ne correspondent pas. »	422	Vider seulement le champ de confirmation, focus dessus.
Non authentifié (page protégée)	« Vous devez être connecté pour accéder à cette page. »	302 → 200 (redirection vers /login)	Rediriger automatiquement vers /login, préserver l'URL d'origine pour retour post-login.
Accès interdit (rôle insuffisant)	« Action non autorisée. »	403	Afficher un message clair, bouton « Retour » / « Accueil ». Journaliser l'événement.
Token CSRF invalide	« Votre session a expiré. Veuillez réessayer. »	419	Recharger le formulaire en conservant les champs si possible ; inviter à se reconnecter.
Ressource introuvable (activité supprimée)	« Page introuvable. »	404	Proposer « Retour aux activités » et un lien de recherche.
Erreur serveur inattendue	« Une erreur est survenue. Nos équipes ont été alertées. »	500	Journaliser, afficher un bouton « Réessayer » / « Accueil ».
Réseau hors-ligne / API indisponible	« Problème de connexion. Vérifiez votre réseau. »	0 (échec réseau)	Mode dégradé : désactiver le bouton, proposer « Réessayer ».
Upload trop volumineux (photo activité)	« Fichier trop volumineux (max 2 Mo). »	413	Afficher la limite et accepter un nouveau fichier.
Trop de tentatives de connexion	« Trop de tentatives. Réessayez dans 1 minute. »	429	Désactiver temporairement le bouton, indiquer le délai restant.

Tableau récapitulatif des principaux scénarios d'erreur anticipés dans *MeetTrip* (formulaire, authentication, API, upload, etc.), avec le message utilisateur affiché, le code HTTP associé et la solution proposée pour chaque cas.

## Annexe 14 : Rendu de la page d'accueil et intégration SEO

The screenshot shows the MeetTrip homepage. At the top, there is a header with the logo "MeetTrip", followed by navigation links: "Accueil", "Activités", "À propos", "Contact", "Connexion", and "Inscription". The main content area features a large heading "Bienvenue sur MeetTrip" and a sub-section "Découvrez le monde autrement" with the text: "MeetTrip facilite les rencontres entre voyageurs autour d'activités, de repas partagés et de trajets communs. Chaque utilisateur peut créer, rejoindre et partager des expériences locales dans une ambiance conviviale et internationale.". Below this is a section titled "Pourquoi l'utiliser ?" with a bulleted list: "• Faire des rencontres authentiques lors de vos voyages.", "• Partager des repas, excursions ou trajets à plusieurs.", and "• Vivre des aventures humaines, pas seulement des destinations.".

Capture d'écran de la page d'accueil “*Bienvenue sur MeetTrip*” illustrant la structure sémantique (H1 → H2) et l'intégration des métadonnées SEO (title, meta description, Open Graph, Twitter Card). Ces éléments assurent une indexation optimale et une présentation cohérente lors du partage sur les réseaux sociaux.

#### Annexe 15 : Fichier robots.txt du site MeetTrip

```
public > 📄 robots.txt
1 # robots.txt - MeetTrip
2 # Contrôle de l'indexation des moteurs de recherche
3
4 User-agent: *
5 # Autorise l'indexation du site public
6 Allow: /
7
8 # Interdit certaines routes internes ou sensibles
9 Disallow: /admin/
10 Disallow: /login
11 Disallow: /register
12 Disallow: /user/
13 Disallow: /api/
14
15 # Indique le sitemap du site
16 Sitemap: https://meettrip.fr/sitemap.xml
17 |
```

Capture d'écran du fichier robots.txt configuré pour le site MeetTrip. Il autorise l'indexation des pages publiques, bloque les routes sensibles (admin, login, register, API) et référence le fichier sitemap.xml afin d'optimiser l'exploration par les moteurs de recherche.

#### Annexe 16 :Tableau d'optimisation SEO et publication

Élément SEO	Objectif	Statut
Balises <title> et <meta>	Fournir un titre et une description uniques pour chaque page afin d'améliorer le référencement naturel.	<input checked="" type="checkbox"/>
Structure Hn (H1 → H2)	Assurer une hiérarchie sémantique claire pour la lecture par les moteurs de recherche et l'accessibilité.	<input checked="" type="checkbox"/>
Fichier robots.txt	Guider les moteurs de recherche sur les pages à indexer et à ignorer.	<input checked="" type="checkbox"/>
Fichier sitemap.xml	Lister les pages principales pour faciliter l'indexation du site par les crawlers.	<input checked="" type="checkbox"/>
Métadonnées (description / mots-clés)	Améliorer le positionnement du site sur les résultats de recherche (SEO).	<input checked="" type="checkbox"/>
Liens internes / navigation claire	Favoriser la circulation entre les pages et renforcer la visibilité des contenus importants.	<input checked="" type="checkbox"/>

Tableau récapitulatif des éléments clés du référencement du site MeetTrip : balises <title> et <meta>, structure Hn, fichiers robots.txt et sitemap.xml, métadonnées et liens internes. Chaque critère est validé pour garantir une bonne indexation et une navigation fluide.

## Test

### Annexe 17 : Cahier de test fonctionnels

Fonctionnalité testée	Étapes de test	Résultat attendu	Statut
Inscription d'un utilisateur	1.Accéder à la page d'inscription. 2. Remplir le formulaire (email + mot de passe). 3. Valider le formulaire.	Un compte est créé et un e-mail de vérification est envoyé à l'utilisateur.	Réussi
Connexion / Déconnexion	1.Saisir un email et mot de passe valide. 2. Cliquer sur "Se connecter". 3. Tester le bouton "Déconnexion".	L'utilisateur accède à son tableau de bord, puis revient à la page d'accueil après déconnexion.	Réussi
Création d'une activité (organisateur)	1. Se connecter en tant qu'organisateur. 2. Ouvrir "Créer une activité". 3. Remplir les champs obligatoires et valider.	L'activité est enregistrée en base et s'affiche dans la liste et sur la carte Leaflet.	Réussi
Réservation d'une activité (participant)	1.Se connecter en tant que participant. 2.Choisir une activité disponible. 3. Sélectionner un créneau et réserver.	La réservation apparaît dans le tableau de bord "Mes activités".	Réussi

Messagerie (organisateur ↔ participant)	1. Accéder à une activité réservée. 2. Envoyer un message via le chat intégré. 3. Vérifier la réception.	Le message s'affiche instantanément dans la conversation liée à l'activité.	Réussi
Suppression d'une activité (admin)	1. Accéder à l'espace administrateur. 2. Supprimer une activité via la liste.	L'activité est supprimée de la base et n'apparaît plus sur la carte.	Réussi
Vérification d'identité (admin)	1. Accéder à la page "Validation identités". 2. Cliquer sur "Valider / Refuser".	Le statut de vérification de l'utilisateur est mis à jour.	Réussi
Affichage de la carte interactive (Leaflet)	1.Accéder à la page "Toutes les activités". 2. Vérifier la présence des marqueurs.	Les activités s'affichent avec leur position sur la carte.	Réussi
Responsive design	1.Tester l'affichage sur mobile, tablette et desktop.	L'interface s'adapte correctement à chaque taille d'écran.	Réussi
Sécurité / CSRF / Auth	1.Soumettre un formulaire sans jeton CSRF. 2. Tenter d'accéder à une route protégée sans être connecté.	L'accès est refusé et l'utilisateur est redirigé vers la page de connexion.	Réussi

Tableau de validation des principales fonctionnalités du projet *MeetTrip* : inscription, connexion, création et réservation d'activités, messagerie, vérification d'identité, affichage de la carte interactive et tests de sécurité (CSRF / Auth). Inclut les résultats attendus et le statut final ("Réussi").

## Annexe 18 :Résultats détaillés des tests d'intégration du projet MeetTrip

N°	Cas de test	Données d'entrée	Résultat attendu	Résultat obtenu	Statut
1	Création d'un utilisateur de test	Email = test@example.com Mot de passe = password	L'utilisateur est créé en base avec mot de passe haché (bcrypt).	Utilisateur visible dans la table users.	Réussi
2	Création d'une activité liée à un utilisateur	title = "Randonnée dans les Alpes", host_user_id = ID du user	L'activité est enregistrée avec la clé étrangère correcte (host_user_id).	L'activité apparaît avec host_user_id = 1.	Réussi
3	Vérification des coordonnées géographiques	latitude = 45.9237longitude = 6.8694	Les champs latitude/longitude sont valides et affichent la position sur la carte Leaflet.	Marqueur visible sur la carte.	Réussi
4	Vérification du champ "why"	Texte = "Rencontrer des passionnés de nature et prendre l'air."	Le champ "why" est enregistré et affiché dans la fiche activité.	Contenu affiché correctement.	Réussi
5	Insertion d'une image de démonstration	URL = <a href="https://example.com/image.jpg">https://example.com/image.jpg</a>	L'image s'affiche dans la vue détail de l'activité.	Image correctement rendue.	Réussi
6	Authentification utilisateur	Email et mot de passe valides	Accès autorisé au tableau de bord et création d'activités possible.	Connexion réussie.	Réussi
7	Authentification invalide	Email incorrect	Message d'erreur "Identifiants invalides".	Message affiché.	Réussi
8	Test de suppression d'activité	ID activité = 1	L'activité est supprimée de la base et disparaît de la carte.	Suppression confirmée.	Réussi
9	Test d'affichage des activités	Appel route /activities	La liste affiche toutes les activités enregistrées.	Les activités de test apparaissent.	Réussi
10	Validation des formulaires	Champ obligatoire "title" vide	Message d'erreur Laravel "Le champ titre est obligatoire."	Message affiché sous le champ.	Réussi

## Gestion de projet

### Annexe 19: ( Schéma des environnements et critères de validation du projet MeetTrip)

Environnement	Objectif	Accès / URL	Particularités techniques
Local (dev)	Développement individuel	<a href="http://localhost:8080">http://localhost:8080</a>	Docker, base MySQL locale, APP_DEBUG=true
Préproduction (preprod)	Tests avant mise en ligne	<a href="https://preprod.meettrip.test">https://preprod.meettrip.test</a>	Données de test, cache activé, debug désactivé
Production (prod)	Environnement final	<a href="https://meettrip.fr">https://meettrip.fr</a>	HTTPS, APP_DEBUG=false, monitoring et backup actifs

Tableau récapitulatif des trois environnements configurés pour le projet MeetTrip : local (dev) pour le développement individuel, préproduction (preprod) pour les tests et validations avant mise en ligne, et production (prod) pour l'environnement final. Chaque environnement précise son URL, ses objectifs et ses particularités techniques.

## Annexe 20 : Tableau des critères de validation “Go / No-Go” avant déploiement

Étape	Critère “Go” (OK pour déploiement)	Critère “No-Go” (bloquant)
Tests unitaires / fonctionnels	100 % des tests critiques passent ( <code>php artisan test</code> )	Tests échoués sur routes principales ou formulaires
Accessibilité & performances	Score Lighthouse ≥ 90	Score < 80 ou erreurs ARIA
Migration & seed	Aucune erreur sur <code>php artisan migrate --seed</code>	Conflits ou rollback forcé
Interface / affichage	Aucune anomalie visuelle majeure	Bugs d'affichage ou crash JS
Rollback possible	Dump SQL de secours disponible	Pas de backup récent

Tableau de validation garantissant la stabilité du projet *MeetTrip* avant chaque mise en production : tests unitaires, accessibilité, migration de base de données, affichage et rollback. Inclut la procédure de rollback et les commandes associées pour restaurer une version stable.

## Annexe 21 : Fiche “Parcours utilisateur” (6 étapes)

Étape	Objectif / Parcours utilisateur	Actions principales	Résultat attendu	Critères d'acceptation
1. Inscription / Connexion	Créer un compte et accéder à l'espace connecté	Saisir <u>email</u> + mot de passe → Valider	Accès au tableau de bord	Champs obligatoires validés, message d'erreur clair, redirection correcte
2. Profil & Vérification d'identité	Compléter son profil et <u>envoyer une pièce d'identité</u>	Ajouter photo, bio, centres d'intérêt, document ID	Profil complet avec statut identité visible	Message "Profil enregistré", statut Validé / En attente / Refusé
3. Création d'activité (Organisateur)	Publier une activité sur <u>MeetTrip</u>	Renseigner titre, lieu (carte), date, description → Valider	Activité publiée et visible dans "Mes activités"	Validation champs, carte interactive OK, message de succès
4. Recherche & Inscription (Participant)	Trouver et rejoindre une activité	Filtrer → Consulter → Cliquer "S'inscrire"	Inscription confirmée, activité ajoutée au tableau de bord	Filtres fonctionnels, bouton désactivé si complet, message de confirmation
5. Messagerie interne	Communiquer avec organisateur ou participant <u>via l'activité</u>	Ouvrir discussion → Envoyer message	Message visible instantanément	Envoi sans rechargement, timestamp visible, fil paginé
6. Suivi	Gérer ses activités à venir et passées	Accéder à <i>Mes activités</i> depuis le tableau de bord	Vue claire sur les activités à venir, passées ou annulées	Statuts corrects (à venir / terminée / annulée), données mises à jour en temps réel

Ce guide met en évidence la simplicité et la cohérence du parcours utilisateur sur MeetTrip, conçu pour offrir une expérience intuitive, accessible et agréable, aussi bien pour les participants que pour les organisateurs.

## Annexe 22 : Tableau de traçabilité des compétences REAC

Compétence REAC (DWWM)	Preuve dans le dossier / application MeetTrip	Section / Annexe / Lien GitHub
<b>Maquetter une application web ou mobile</b>	Wireframes et maquettes du site MeetTrip, design responsive et accessibilité.	Partie <i>Adaptation UX</i> + Annexes 5–7
<b>Réaliser une interface utilisateur statique et adaptable</b>	Pages React (Accueil, Activités, Contact) avec structure Hn, responsive Tailwind.	Activité Type 1 – Exemple 1 + Annexe 5
<b>Développer une interface utilisateur web dynamique</b>	Composants React dynamiques (carte Leaflet, filtres d'activités, formulaires).	Activité Type 1 – Exemple 2 + Annexe 21
<b>Mettre en place une base de données relationnelle</b>	Modèle MySQL avec tables <i>users</i> , <i>activities</i> , <i>dates</i> , <i>guest_users</i> .	Activité Type 2 – Exemple 1 + Schéma DB
<b>Développer les composants d'accès aux données (SQL / Eloquent)</b>	Routes Laravel et contrôleurs ( <i>ActivitiesController</i> , <i>UserController</i> ).	Activité Type 2 – Exemple 2
<b>Développer la partie back-end d'une application web sécurisée</b>	API Laravel (CRUD activités, vérification identité, authentification).	Activité Type 2 – Exemple 3 + Annexe 8–11
<b>Sécuriser une application web</b>	Gestion CSRF, cookies SameSite, CORS, .env sécurisé.	Annexes 8–11
<b>Réaliser les tests fonctionnels et d'intégration</b>	Tableau de tests + résultats MeetTrip (inscription, création activité, messagerie).	Annexes 17–18
<b>Assurer la mise en production et le suivi</b>	Procédure de validation Go/No-Go, déploiement et rollback.	Annexes 19–20
<b>Assurer la qualité du code et la version Git</b>	Workflow Git (branches, commits normés, PR validée).	Annexe 23
<b>Optimiser les performances et SEO</b>	Tests Lighthouse, lazy loading, balises meta et sitemap.xml.	Annexes 14–16 + 24

## Annexe 23 : Méthodologie de versionnement et workflow Git du projet MeetTrip

### Commits

The screenshot shows a GitHub commit history for the 'main' branch. It includes filters for 'All users' and 'All time'. The commits are organized by date:

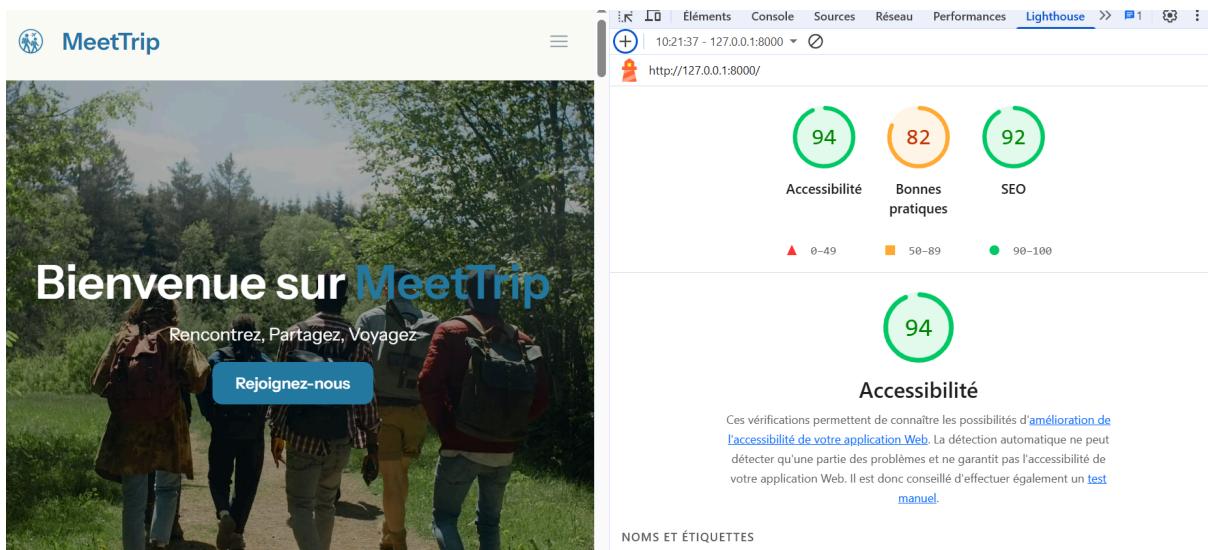
- May 28, 2025:
  - debut back (commit 4ae96dc) - Nadia-boujnah committed on May 28 · 1 / 2
- May 26, 2025:
  - creation role, middleware et controller (commit ee2ac5c) - Nadia-boujnah committed on May 26
  - Merge branch '1-front-non-connecte' (commit dc39e49) - Nadia-boujnah committed on May 26 · 1 / 2
  - Merge final (commit 6422c5c) - Nadia-boujnah committed on May 26 · 1 / 2
- May 2, 2025:
  - Commits on May 2, 2025 (no specific commits listed)

Extrait des commits GitHub du projet MeetTrip, montrant la structure des branches et la traçabilité des modifications.

```
C:\Users\Utilisateur\Documents\meettrip>git log --oneline --graph --decorate -n 6
* 1ecd24b (HEAD -> main) creation table
* 4ae96dc (origin/main) debut back
* ee2ac5c creation role, middleware et controller
* dc39e49 Merge branch '1-front-non-connecte'
* 1ecd24b (HEAD -> main) creation table
* 4ae96dc (origin/main) debut back
* ee2ac5c creation role, middleware et controller
* dc39e49 Merge branch '1-front-non-connecte'
|
| * 2947b30 (origin/1-front-non-connecte, 1-front-non-connecte) updates routes
| * d5f64db (origin/reprise-home) merge from home-non-connecte
~
~
~
* 1ecd24b (HEAD -> main) creation table
* 4ae96dc (origin/main) debut back
* ee2ac5c creation role, middleware et controller
```

Visualisation du log Git local (git log --oneline --graph --decorate) affichant la hiérarchie des branches (main, feature, reprise-home) et l'historique des merges.

## Annexe 24 : Performances et optimisation (audit Lighthouse)



Audit réalisé avec Google Lighthouse sur la page Accueil de l'application. Les scores élevés en accessibilité (94), bonnes pratiques (82) et SEO (92) démontrent la qualité technique du site et l'attention portée à la performance et à l'expérience utilisateur.