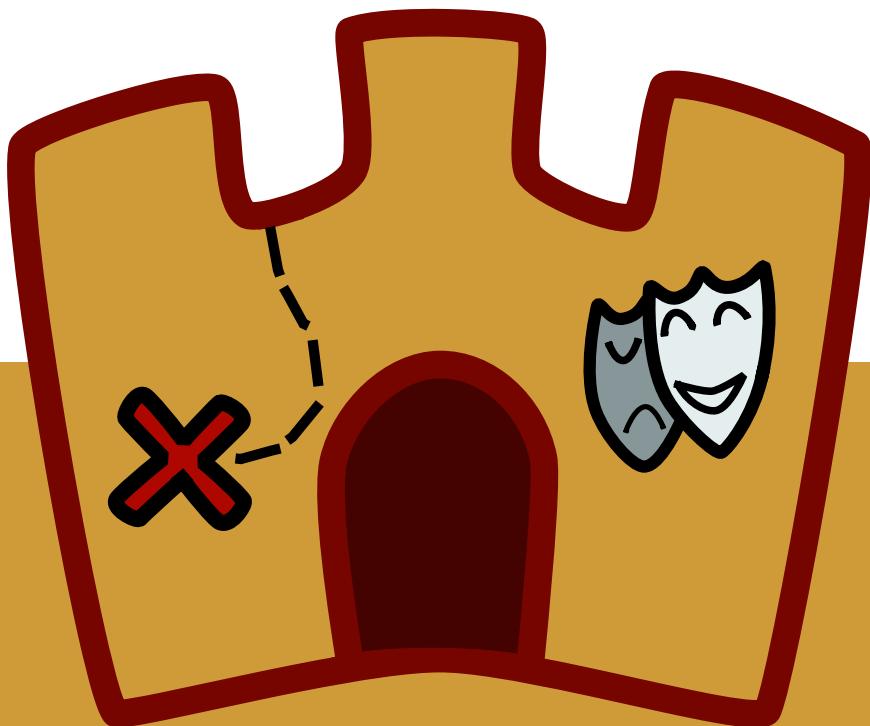


Mona

11/12/2024

IUT2A
Université Grenoble Alpes



MonaDev

GROUPE 14

Nils – Mathias – Yani – Loan – Raphaël – Joshua - Vladimir

CONTEXTE DU PROJET	2
Description	2
cadrage du projet	2
Besoins fonctionnels	3
Besoins non-fonctionnels	4
Cadrage du projet.....	4
Evolution des risques identifiés dans la première phase :	4
Retours d'expérience sur les contraintes et les risques	4
CONCEPTION DU PROJET	6
FONCTIONNALITÉS PHARES	6
Processus métiers décrits	6
Etapes création d'événement.....	6
Etapes pour devenir un prestataire	8
Critères ergonomiques	9
Guidage et Gestion des erreurs	10
Charge de travail et contrôle explicite.....	10
Structuration de la base de données	13
Structuration du code.....	14
Plan du site	15
REALISATION DU PROJET.....	16
Étude de faisabilité.....	16
Test de l'utilisation de ReactJS	16
Test de l'utilisation de GitLab	18
Stucture matérielle et technologique de notre site	18
Possibilité 1 : Serveur ASSR de l'IUT2	18
Possibilité 2 : Serveur dans un cloud IONOS	19
Effectué et restant à faire	21
Suivi de l'effectué (Phase 1 & 2)	21
Analyse du restant à faire (Phase 3)	21
Annexes	22
Contraintes et risques	22
Etude de faisabilité.....	23
Plan du site	25
BPMN	25
Schéma relationnel	26
Maquettes	27

CONTEXTE DU PROJET

DESCRIPTION

Dans le cadre d'un projet de développement d'application web, nous avons pour mission de concevoir une plateforme web dédiée à la valorisation du patrimoine culturel. Nous avons imaginé **Mona**, une application web qui donne la possibilité de transformer la manière dont les monuments et les activités culturelles sont découverts et réservés. Notre projet s'articule autour de quatre objectifs principaux : faire connaître les monuments, les faire aimer, les rendre accessibles et contribuer à leur préservation.

Inspirée du modèle de mise en relation comme *Airbnb*, Mona permet aux propriétaires de "poster" leurs monuments sur la plateforme en y intégrant des informations détaillées (histoire, caractéristiques, date de construction). Le processus d'ajout sera simplifié avec la possibilité de trouver et préremplir facilement les informations liées à leurs sites grâce à une base de données des monuments.

De plus, les propriétaires de monuments ont la possibilité de définir si leur site est uniquement ouvert aux visites classiques ou s'il peut accueillir des activités spécifiques (escapes games, chasses aux trésors, spectacles, films projetés et événements personnalisés) organisées par des prestataires. Ces événements thématiques offrent une expérience unique et permettant de redécouvrir le patrimoine culturel de manière originale et amusante.

L'objectif de Mona est donc triple :

- Pour les visiteurs : Offrir une plateforme centralisée qui facilite la découverte et la réservation de visites ou d'événements dans des monuments historiques.
- Pour les propriétaires de monuments : Fournir une solution pratique qui leur permet de promouvoir leurs bâtiments pour attirer un public plus large, et ainsi entretenir leur lieu.
- Pour les prestataires d'activités : Créer un espace de collaboration qui leur permet de proposer, organiser et gérer des expériences dans les monuments, enrichissant ainsi l'offre culturelle.

Avec Mona, nous voulons inciter les visiteurs à explorer la région Auvergne-Rhône-Alpes d'une manière nouvelle, tout en facilitant la gestion et l'organisation des événements pour les acteurs du patrimoine.

CADRAGE DU PROJET

Lors de la 1^{ère} phase, nous avons réalisé un dossier de conception pour cadrer notre projet pour y définir nos objectifs, et vous donner les détails concernant la veille effectuée pour analyser les concurrents existants ainsi que les besoins identifiés, le profilage des futurs utilisateurs, les risques/contraintes potentiels et leurs mitigations trouvées ainsi qu'un aperçu de notre répartition des tâches mais surtout vous montrez nos objectifs pour la réalisation finale.

Puis, nous avons également identifié les parties prenantes du projet notamment notre équipe, les donneurs d'ordre : l'association de plusieurs mairies en Auvergne-Rhône-Alpes, qui sont aussi des utilisateurs finaux, pour lesquels nous avons dressé des profils afin de mieux cerner leurs attentes. (Les visiteurs, les propriétaires de monuments et les prestataires d'évènements).

Voici les différents personas identifiés :

- Louis Martel**, Professeur d'histoire et passionné de culture, Âge : 29 ans, Profession : Professeur d'histoire dans un lycée, Objectifs : Participer à des visites immersives, explorer l'histoire de manière originale, capturer des images de sites historiques et échanger avec des experts locaux.
- Loan Dufresne**, Étudiant et amateur d'escape games, Âge : 18 ans, Profession : Étudiant en histoire, Objectifs : Vivre des expériences authentiques et immersives, combinant culture et jeux d'évasion dans des sites patrimoniaux, avec des défis réalistes et stimulants.
- Émilie Bernard**, Prestataire d'événements culturels, Âge : 34 ans, Profession : Organisatrice d'événements culturels, Objectifs : Promouvoir le patrimoine local à travers des événements uniques, attirer un public plus large, et collaborer avec des propriétaires de monuments pour organiser des activités immersives.
- Isabelle Dupont**, Responsable du département Patrimoine et Culture dans l'association des mairies d'Auvergne-Rhône-Alpes, Âge : 62 ans, Profession : Responsable culturelle, Objectifs : Réserver et valoriser les monuments historiques de la région, promouvoir le patrimoine local, collaborer avec des prestataires pour organiser des événements culturels innovants, et améliorer l'attractivité des communes grâce à des outils numériques modernes.

À partir de cela, nous avons recueilli, priorisé et hiérarchisé les besoins fonctionnels et non-fonctionnels de notre application. Enfin, notre répartition du travail efficace a été possible grâce à une répartition des tâches et l'utilisation d'outils collaboratifs comme GitLab et Notion. Cette première étape nous a permis de poser des bases solides pour aborder la suite en définissant nos objectifs, les fonctionnalités, les besoins et le profilage de nos utilisateurs finaux divisés en 3 groupes (visiteurs, propriétaire et prestataire).

BESOINS FONCTIONNELS

Nous rappelons dans ce tableau les différents besoins fonctionnels les plus importants pour Mona en fonction du type de l'utilisateur.

Visiteur	Prestataire	Propriétaire
Trouver facilement un lieu sur le site afin d'accéder à diverse informations sur le lieu (on y répond grâce au système de recherche et filtres)	Organiser ses événements (voir les événements passés, en ajouter de nouveaux simplement)	Ajouter ses propriétés à la base de données du site pour le rendre accessible à tous
Voir les événements à venir de la manière la plus adaptée à l'utilisateur (on y répond avec la liste, calendrier, carte)	Faire des propositions aux propriétaires si des paramètres sont trop restrictif	Préremplir facilement à l'aide de la base de données
Acheter des billets pour une visite ou un événement	Voir les monuments disponibles à une date donnée	Choisir les types d'évènements possibles

S'attribuer un rôle (prestataire, propriétaire)		
---	--	--

BESOINS NON-FONCTIONNELS

Notre logiciel doit satisfaire plusieurs contraintes techniques afin de garantir une plateforme stable et de qualité. Parmi elles :

- **Optimisation, performance et faible latence** : La plateforme doit offrir un temps de réponse rapide et être capable de gérer un grand nombre d'utilisateurs simultanément grâce à un code scalable et une optimisation du schéma de la base de données.
- **Sécurité** : La protection des données personnelles est essentielle, avec un chiffrement des données sensibles comme le mot de passe (chiffrement + hachage sha256 dans la base de données). Utilisation de d'un protocole de communication sécurisée (https).
- **Accessibilité** : L'application doit être compatible avec les différents navigateurs et appareils (ordinateurs, mobiles, tablettes), avec une interface responsive.
- **Fiabilité** : Le site doit être disponible continuellement. En cas d'indisponibilité de l'api open data, utiliser une base de données interne.
- **Utilisabilité** : La plateforme doit être intuitive, et fournir des interactions fluides, notre critère ergonomique démontre que ce besoin est respecté

CADRAGE DU PROJET

EVOLUTION DES RISQUES IDENTIFIES DANS LA PREMIERE PHASE :

Depuis notre dernier rendu, nous avons retravaillé les contraintes et risques en séparant plus clairement ces deux notions. Nous avons trouvé de nouveaux risques correspondant à nos attendus pour cette phase sur divers points :

- L'aspect réglementaire avec le respect du RGPD. En effet, notre application web conserve et récupère beaucoup d'informations sur nos utilisateurs (les différents formulaires).
- L'aspect budgétaire avec la collecte de données à effectuer sur des données gratuites, et donc limité à celles open source avec des informations qui peuvent être fausses.
- L'aspect technique avec l'utilisation d'un serveur externe IONOS pour pouvoir accéder à Mona depuis n'importe quel endroit qui pourrait ne pas être assez puissant.

Plus de détails sont présents dans le tableau de l'Annexe I qui présente surtout les nouveaux risques identifiés.

RETOURS D'EXPERIENCE SUR LES CONTRAINTES ET LES RISQUES

Durant cette phase 2, nous avons rencontré à plusieurs reprises des problèmes concernant la disponibilité des salles informatiques. La nécessité de changer de salle entre deux séances a souvent entraîné une perte de temps, généralement estimée à 20 minutes. Cette situation a mis en évidence que notre étude des risques, réalisée lors de la phase de conception, était trop optimiste. Notre stratégie de mitigation était de venir en avance pour réserver les postes de programmation en premier. Cependant, les salles n'étaient plus disponibles la 2^{ème} heure.

Nous avons également été confrontés à une contrainte temporelle importante. Une semaine avant le rendu, nous avons constaté qu'il restait une charge de travail conséquente à accomplir.

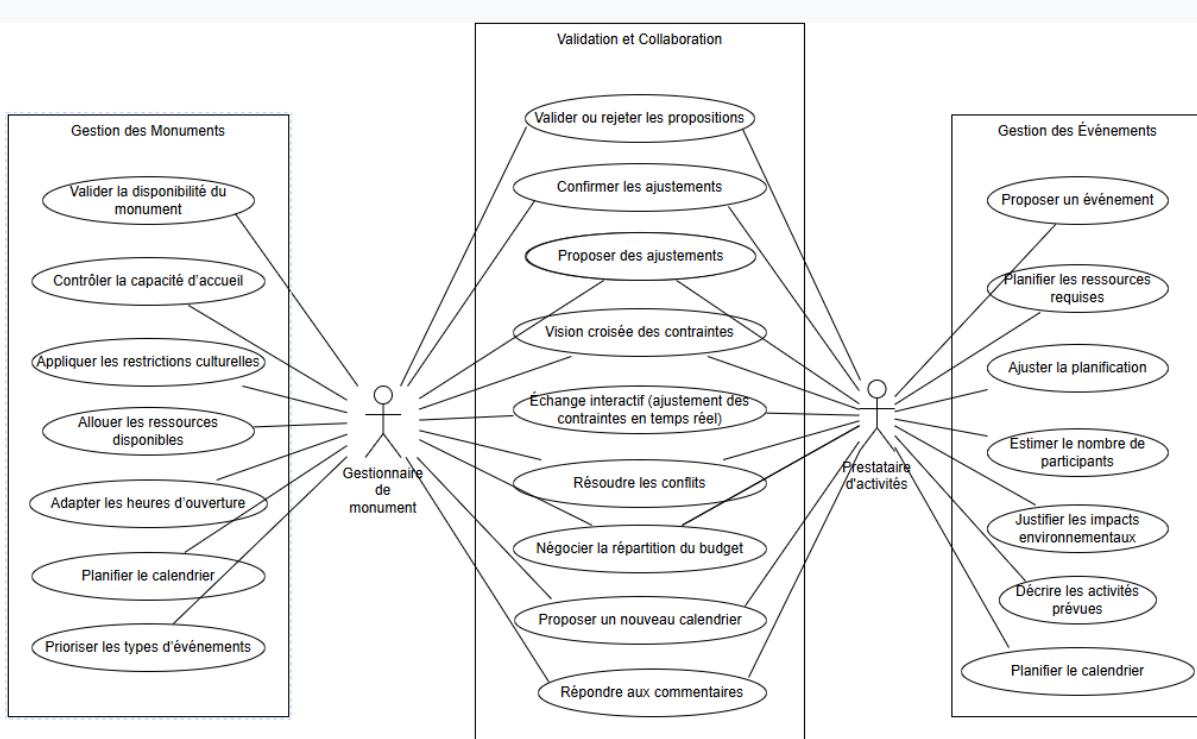
Face à cette situation, nous avons appliqué la stratégie de mitigation prévue : une réorganisation complète du planning avec une répartition claire et priorisée des tâches. Cette approche s'est avérée efficace, car elle nous a permis de respecter les délais sans accumuler de retard, même si cela a nécessité des efforts supplémentaires de la part de toute l'équipe.

Globalement, nous estimons que la criticité attribuée à ces deux risques était correcte. L'impact, bien que notable, est resté gérable grâce aux mesures prises. Ces expériences confirment la pertinence de notre analyse.

CONCEPTION DU PROJET

FONCTIONNALITÉS PHARES

L'objectif est d'offrir une application simple et efficace permettant aux gestionnaires de monuments de superviser et organiser leurs bâtiments, tout en donnant aux prestataires les outils nécessaires pour proposer et gérer des événements. La collaboration entre ces deux acteurs repose sur des processus clairs, allant de la planification initiale à la validation finale des événements. Ce système favorise ainsi une valorisation optimale des monuments, tout en offrant des expériences adéquates aux visiteurs.



PROCESSUS MÉTIERS DÉCRITS

ETAPES CRÉATION D'ÉVÉNEMENT

Le diagramme BPMN présenté illustre le processus de création d'événement par le prestataire d'événement. Ce processus ne peut être initié que par le prestataire lui-même, qui joue un rôle central dans le fonctionnement de notre application. En effet, la création d'un événement est à l'origine de l'existence même des événements sur notre plateforme, ce qui en fait un élément clé pour son bon fonctionnement.

Ce processus intègre plusieurs fonctionnalités essentielles que l'on retrouve dans les cas d'utilisation. Parmi elles, figure notamment les fonctions de propositions d'ajustement, validation d'ajustement ou encore la validation et rejet des propositions.

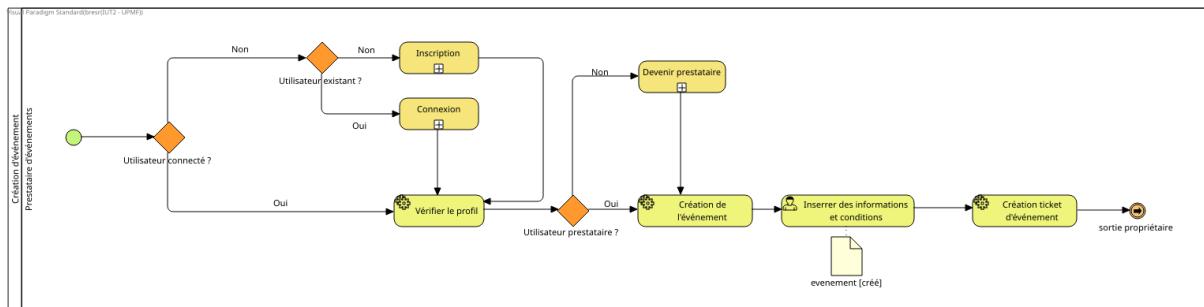


Schéma 1 - Création d'un événement (étape 1)

L'utilisateur initie le processus de création d'événement en cliquant sur le bouton « créer un événement ». L'utilisateur initiant ce processus étant obligatoirement prestataire, l'application va d'abord vérifier si l'utilisateur est connecté, si ce n'est pas le cas, on lui proposera de se connecter (Annexe 4), et s'il n'a pas déjà de compte, on lui proposera de s'inscrire (Annexe 5). Suite à cela, l'application va vérifier si le profil de l'utilisateur est bien « prestataire d'activité », et s'il ne l'est pas, on lui propose de le devenir (cf : Etapes pour devenir prestataire). Une fois l'utilisateur bien connecté sur un compte prestataire, il pourra créer l'événement qu'il souhaite en entrant les informations importantes et les conditions qu'il souhaiterait imposer.

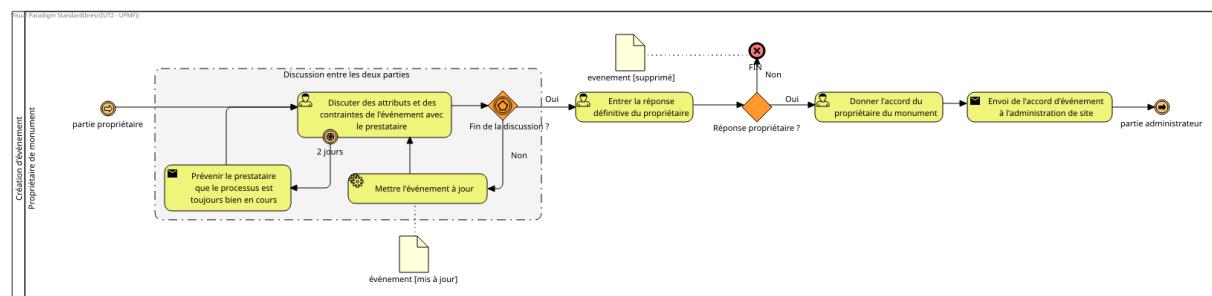


Schéma 2 - Création d'un événement (étape 2)

Ensuite, un ticket de création d'événement est directement envoyé au propriétaire du monument dans lequel le prestataire souhaiterait organiser l'événement. Une fois que le propriétaire du monument reçoit le ticket, il lui est libre d'analyser la proposition d'événement et de marchander avec le prestataire comme il leur convient. Une fois arrivés sur un accord en ce qui est des conditions de chacun, le propriétaire du monument peut refuser l'événement ou mettre fin au processus, ou valider la demande, et un ticket de création de l'événement validé par les deux partis est envoyé à l'administration de l'application.

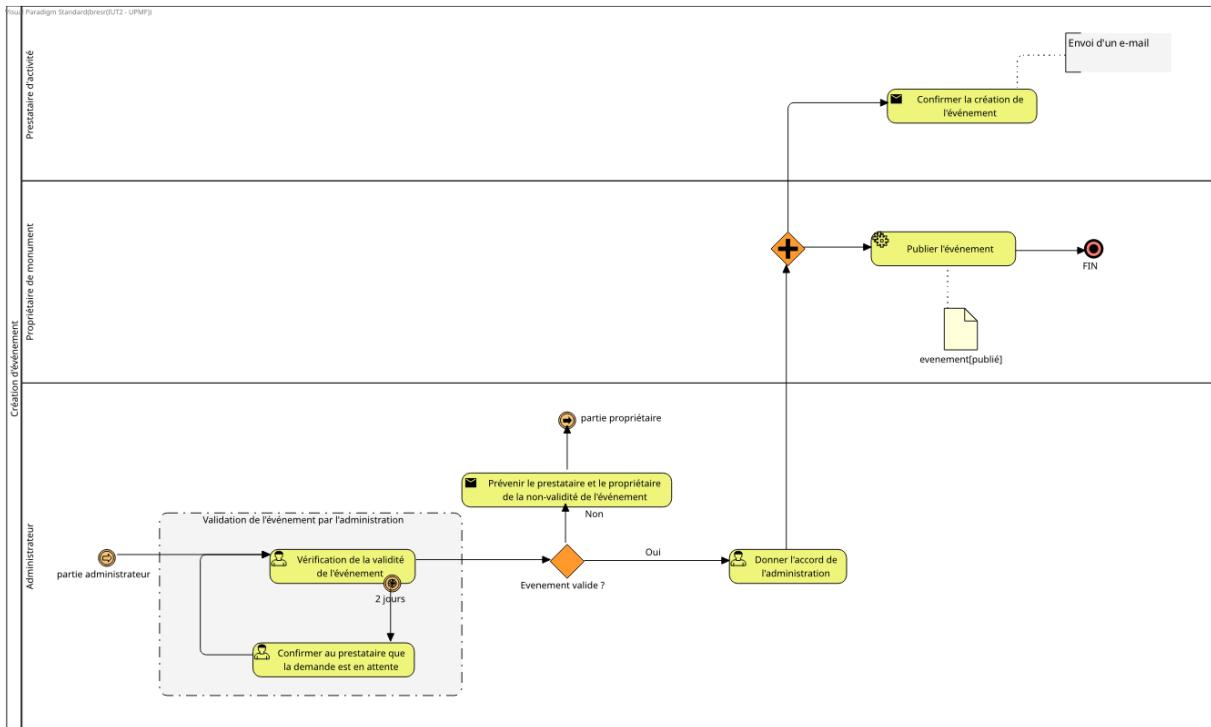


Schéma 3 - Création d'un événement (étape 3)

Une fois que l'administration reçoit le ticket, il est libre à eux de valider ou non l'événement demandé en fonction des conditions d'utilisation et termes de contrats de l'application. Si l'administration refuse, il est toujours libre au propriétaire du monument et au prestataire de l'activité de se mettre d'accord sur des modifications rentrant dans les permissions de l'application. Une fois l'événement validé par l'administration, il sera créé et ajouté au site, et une notification de validation de l'événement sera envoyée à tous les utilisateurs à l'origine de la création de cet événement.

ETAPES POUR DEVENIR UN PRESTATAIRE

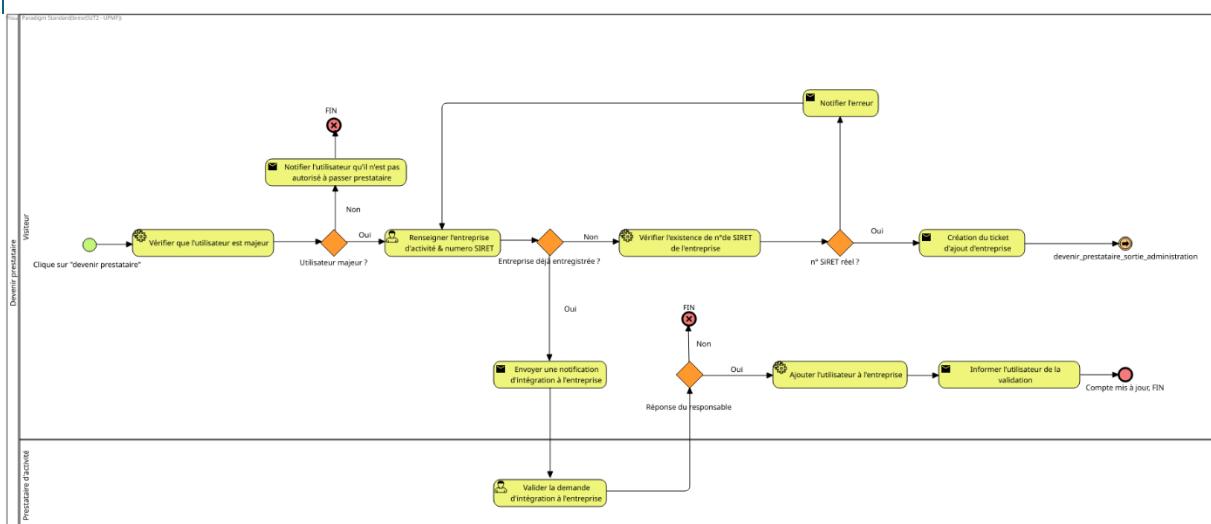


Schéma 4 - Devenir prestataire (étape 1)

L'utilisateur initie le processus de passage de compte client en compte prestataire d'activités en cliquant sur le bouton « devenir prestataire ». A l'appui du bouton, l'application va vérifier que l'utilisateur est majeur (âge ≥ 18), s'il ne l'est pas, on notifie l'utilisateur qu'il n'est pas autorisé à devenir prestataire d'activité, si l'utilisateur est majeur, pas de notification, et on le fait passer à la suite. On demande ensuite à l'utilisateur de renseigner le nom de son entreprise ainsi que son numéro de SIRET. A l'entrée de ces informations, l'application va vérifier si l'entreprise existe déjà dans la base de données. Si l'entreprise existe déjà dans la base de données, une notification de demande d'entrée de l'employé sera envoyée à l'utilisateur prestataire d'activité ayant le rôle de responsable de l'entreprise, si celui-ci accepte, l'utilisateur à l'origine du processus est ajouté à son entreprise, et si le responsable refuse, l'utilisateur sera refusé et le processus prendra fin. Si le numéro de SIRET n'est pas déjà présent dans la base de données, l'application va vérifier l'existence du numéro fourni, s'il existe et qu'il correspond au nom de l'entreprise renseigné, sa demande de création d'entreprise est enregistrée, et un ticket d'ajout d'entreprise est envoyé à l'administration.

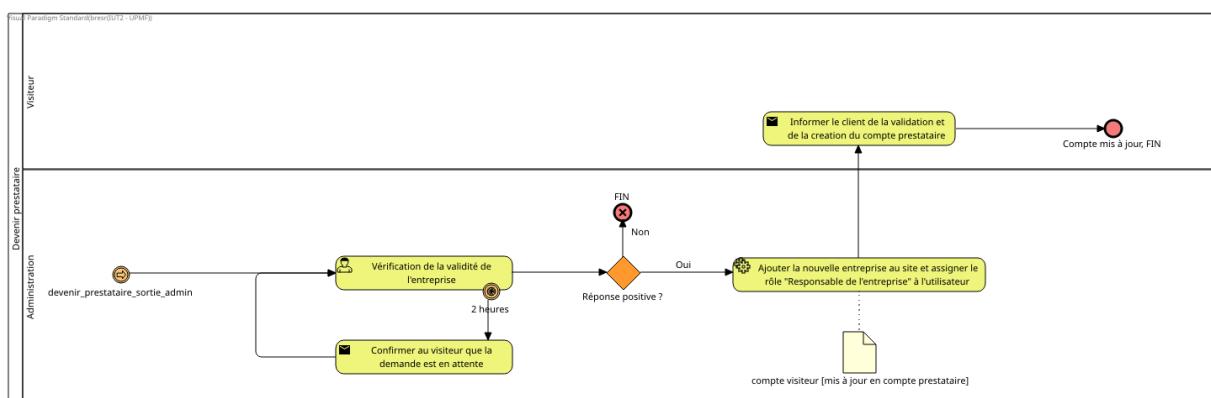


Schéma 5 - Devenir prestataire (étape 2)

Une fois que l'administration reçoit ce ticket, ils vont personnellement vérifier la validité de l'entreprise pour éviter toute fraude, si la réponse est négative, le processus prend fin, l'entreprise n'est pas créée, et l'utilisateur ne devient pas prestataire ; en revanche, si l'entreprise est validée par l'administration, alors celle-ci est rajoutée à la base de données, et l'utilisateur à l'origine de cette création est nommé responsable de ladite entreprise. Une notification de validation de l'entreprise et du compte est envoyée à l'utilisateur.

CRITERES ERGONOMIQUES

Dans cette section, nous présentons les différentes maquettes importantes des pages de l'application, en détaillant les critères ergonomiques qui ont guidé leur conception basée sur les besoins de nos différents personas. Cette analyse couvre l'ensemble de l'application, mettant en avant des principes clés nécessaire tels que :

- **Homogénéité :** Le design a été conçu de manière uniforme par un seul membre de l'équipe, ce qui garantit une cohérence visuelle et fonctionnelle à travers toutes les pages. Les utilisateurs retrouvent facilement leurs repères, aucune section n'est vraiment différente des autres.
- **Compatibilité :** Ce critère est important pour notre application puisqu'une partie de notre public cible sont de jeunes personnes qui ont des habitudes d'utilisation sur le web. L'interface s'aligne donc avec leurs habitudes tout en respectant une charte graphique qui évoque une impression d'ancienneté. Les conventions de navigation habituelles sont respectées, par exemple l'utilisation de la touche "Échap" pour fermer un pop-up. Les

éléments visuels, tels que l'icône de "bonhomme" pour le profil et les paramètres ou le "cœur" pour aimer, sont facilement compréhensibles par tout le monde.

Concernant nos choix, nous avons décidé d'utiliser les critères ergonomiques de Bastien Scapien en se focalisant sur ceux les plus adaptés à nos personas et nos sites : critères de guidage, charge de travail ainsi qu'homogénéité et compatibilité décrits plus haut. Le critère de guidage semble essentiel dans un site billetterie de type Airbnb comme le nôtre car il permettra aux visiteurs de pouvoir facilement réserver. Tandis que le critère de charge de travail est un critère à privilégier pour les propriétaires de monuments souvent âgés.

GUIDAGE ET GESTION DES ERREURS

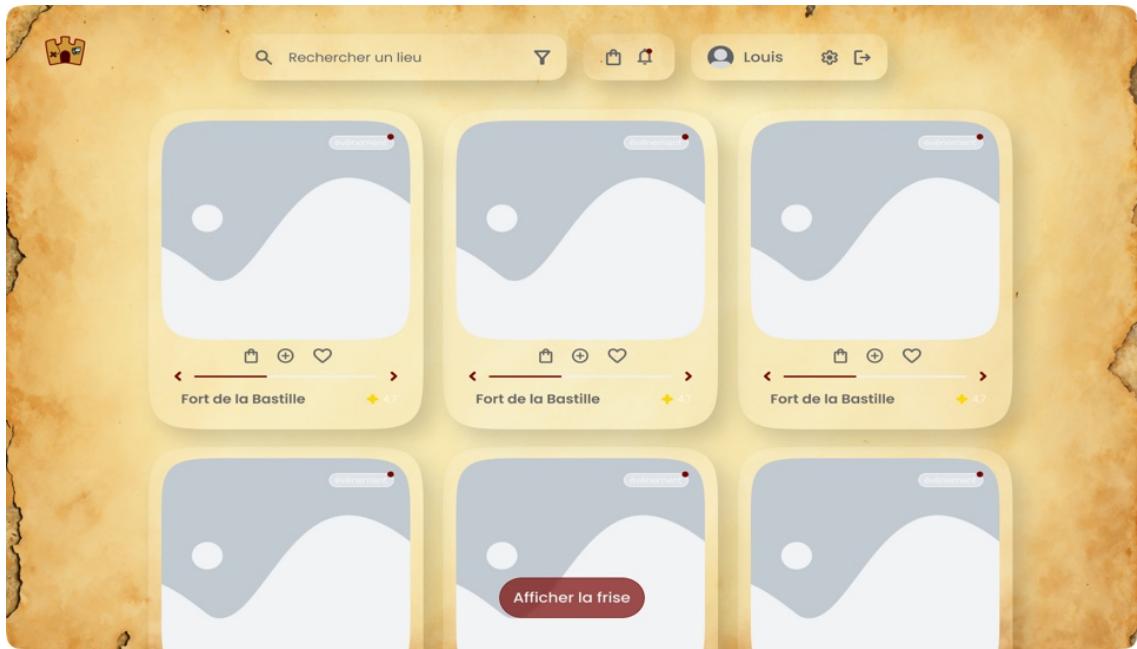
L'application offre un guidage clair et une bonne gestion des erreurs, ce qui est primordial pour toute application :

- Lisibilité
 - *Maquette 1* : les informations sont bien espacées et faciles à comprendre ce qui rend la densité d'information raisonnable.
 - *Maquette 2* : l'arrière-plan est flouté pour bien mettre en avant les informations utiles.
- Groupement
 - *Maquette 1* : On regroupe par localisation et format les monuments (avec ou sans événements) les informations affichées sont donc liées à ce bâtiment, les groupes sont homogènes et cohérents pour guider l'utilisateur sur la plateforme.
- Incitations visuelles
 - *Maquette 1* : Des boutons explicites incitent à l'action grâce à des animations comme la barre de recherche qui est animé pour inciter les utilisateurs à cliquer.
- Feedback immédiat
 - *Maquette 1* : Les erreurs identifiées sont clairement et immédiatement expliquées à l'utilisateur (Exemple "Votre mot de passe doit contenir au moins 8 caractères") et on guide les utilisateurs pour en sortir.

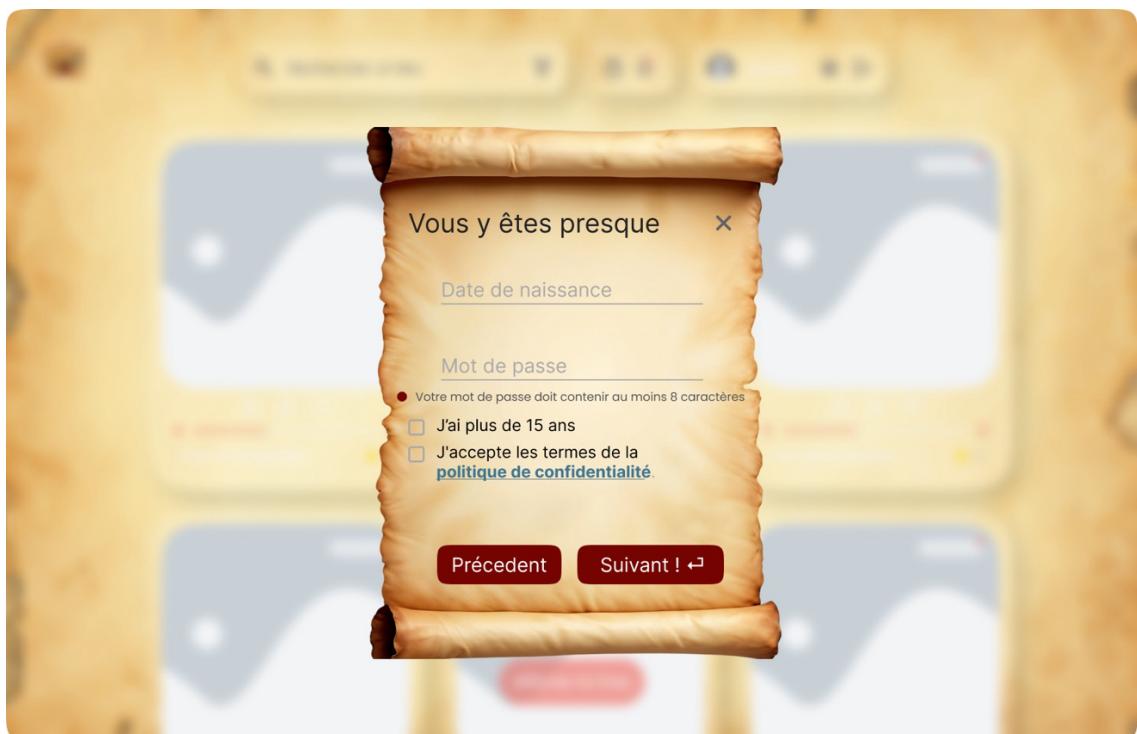
CHARGE DE TRAVAIL ET CONTROLE EXPLICITE

L'interface est conçue pour minimiser l'effort utilisateur, dans un site où il peut y avoir beaucoup de saisi (notamment pour la création d'événements avec plusieurs formulaires), il est important de minimiser la charge de travail et de permettre un contrôle explicite à l'utilisateur :

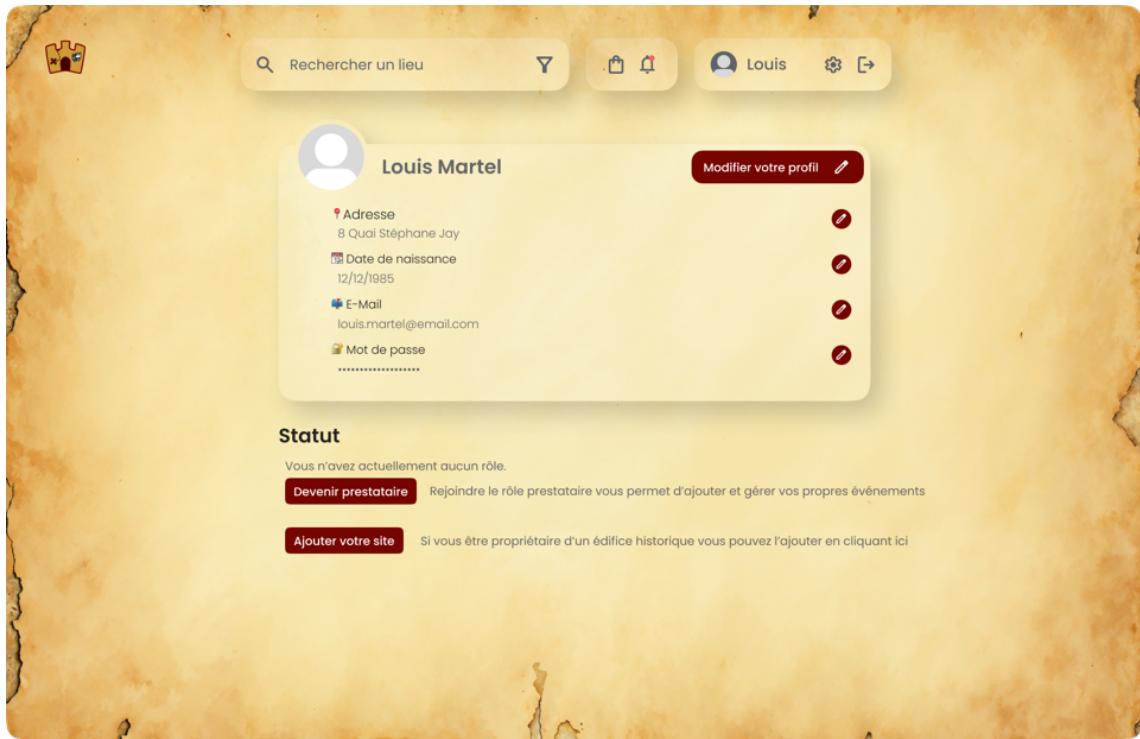
- Densité informationnelle
 - *Maquette 4* : les informations sont réparties sur plusieurs pages pour éviter la surcharge pour ne pas démotiver nos utilisateurs à remplir les formulaires.
- Contrôle utilisateur
 - *Maquette 4* : l'utilisateur peut modifier ses actions à tout moment. Il peut par exemple interrompre un processus en cours (appuyés sur la croix pour annuler l'action de devenir prestataire)
 - L'utilisateur peut aussi revenir à la page d'accueil depuis n'importe quel endroit dans le site.



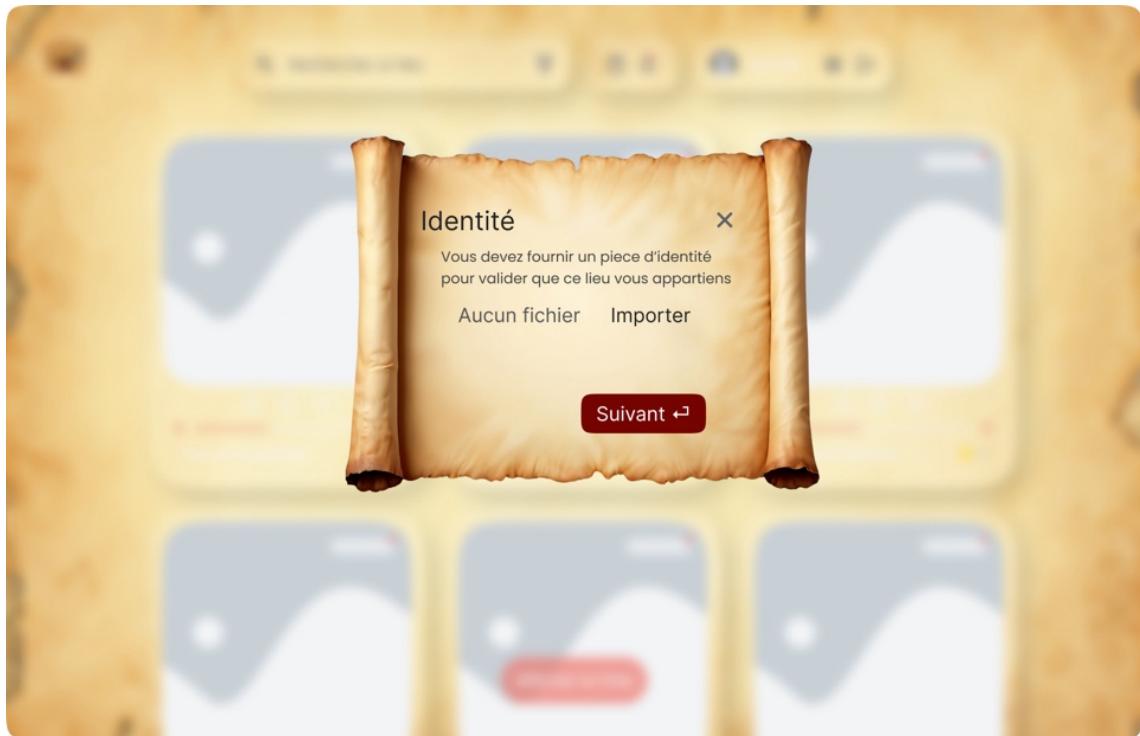
Maquette 1 - Page d'accueil avec les différents événements/visites



Maquette 2 - Page de création compte (dernière étape)



Maquette 3 - Page paramètre d'un utilisateur par défaut



Maquette 4 - Vérification d'identité

STRUCTURATION DE LA BASE DE DONNEES

Afin de définir l'architecture future de la base de données de Mona, nous avons élaboré un modèle de données relationnel qui permet de visualiser les relations entre les différentes entités et leurs interactions. La table Utilisateur contient les informations des utilisateurs et gère la répartition des rôles, chacun ayant des accès spécifiques selon son statut. Le propriétaire de monument est associé à un ou plusieurs monuments dont il détient la gestion. Tous les utilisateurs peuvent effectuer des réservations pour visiter des monuments ou participer à des événements spécifiques. Le prestataire d'événement, quant à lui, est rattaché à une entreprise, en tant que responsable ou employé. Cette entreprise peut organiser des événements dans un monument, à condition que le propriétaire de celui-ci donne son approbation. Enfin, chaque utilisateur peut réserver une place pour un événement particulier, en fonction de sa disponibilité et des événements proposés. Une représentation relationnelle offrant une représentation des tables de la base de données est disponible en [Annexe VI](#).

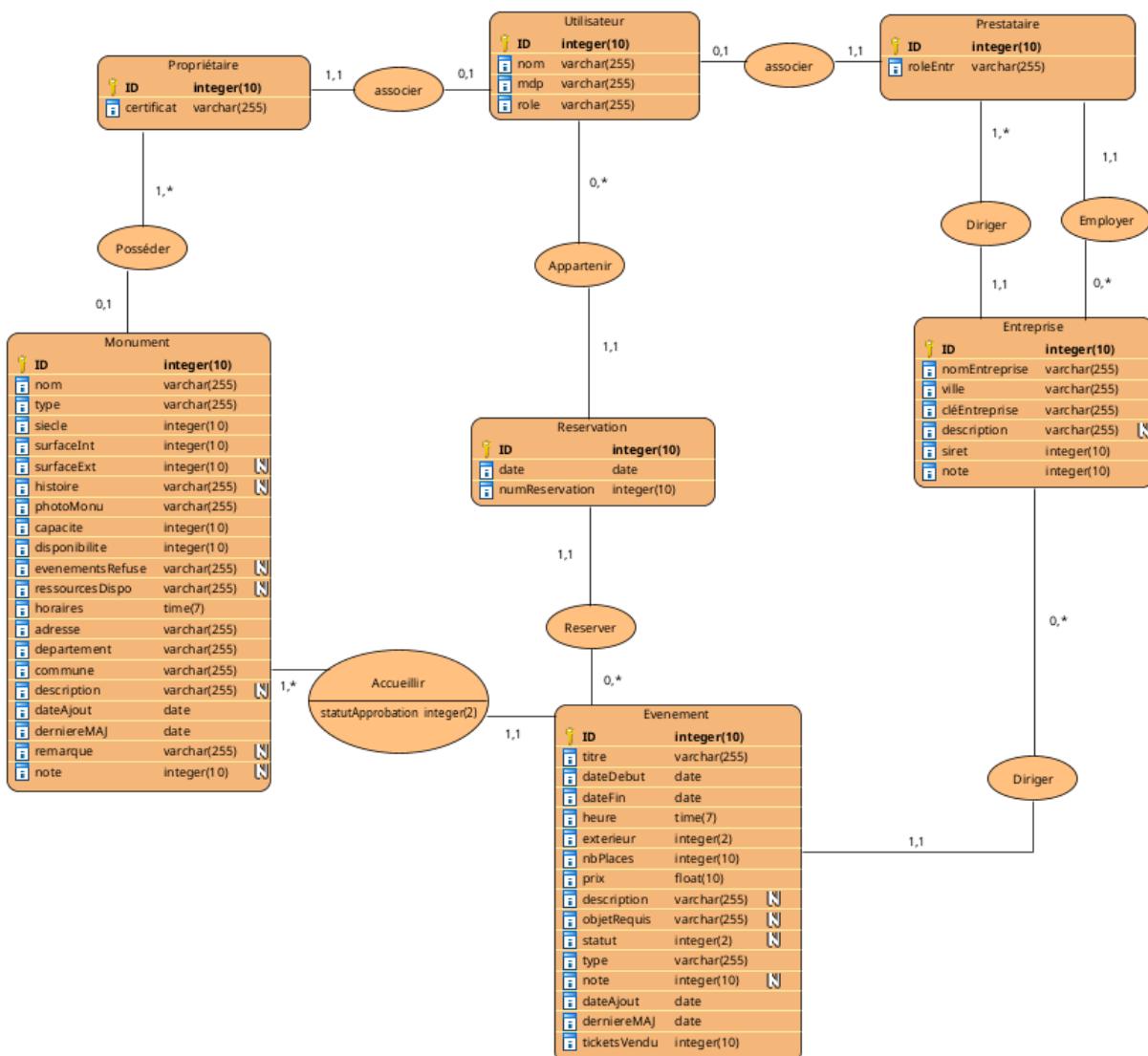


Figure 1 – Schéma base de données

STRUCTURATION DU CODE

Le diagramme de classe ci-dessous représente l'organisation des principales entités du système de gestion de Mona. Il s'appuie sur une hiérarchie claire pour modéliser les interactions et les responsabilités des différents acteurs.

Les classes Propriétaire et Prestataire sont des spécialisations de la classe parente Utilisateur qui centralise la gestion des utilisateurs, qu'ils soient propriétaires de monuments, simples visiteurs, ou prestataires d'événements. Chaque Prestataire est associé à une Entreprise, dans laquelle il occupe le rôle de Responsable ou d'Employé. Les Entreprises jouent un rôle clé en tant qu'organisatrices d'événements, choisissant des monuments en accord avec leurs propriétaires selon les disponibilités et horaires des lieux pour accueillir leurs activités.

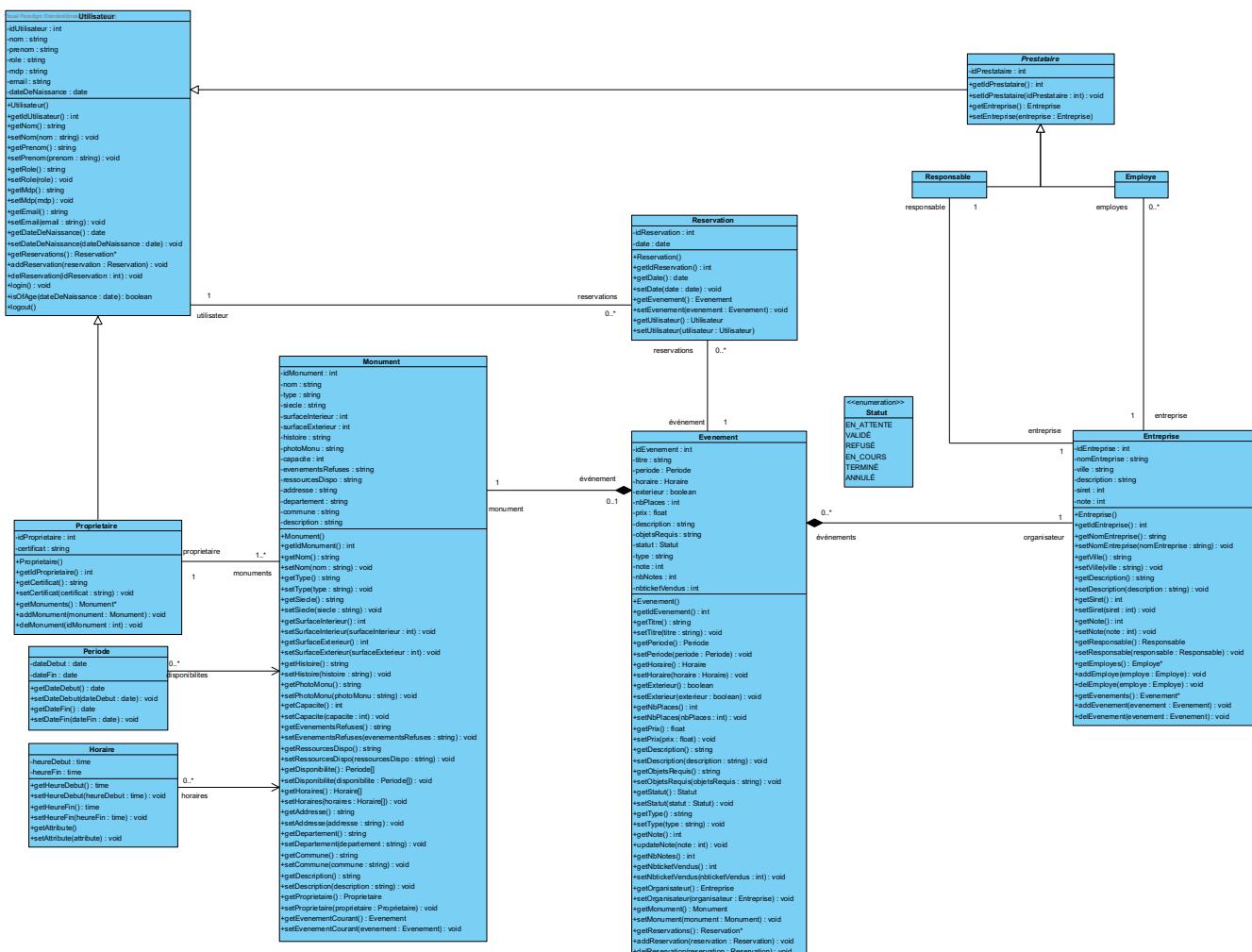


Figure 2 -Diagramme de classes

PLAN DU SITE

Après avoir réalisé nos maquettes, nous avons dû les organiser en réalisant un plan de site pour représenter la structure depuis la page d'accueil afin d'avoir un aperçu final du site pour un visiteur (Annexe III).

Le visiteur a plein d'options à sa disposition depuis la page 'Accueil'. Le 'Calendrier' pour les évènements à venir et la 'Liste des monuments' qui permettent d'accéder à la page 'Aperçu d'un monument' en cliquant dessus. Les pages 'Profil', 'Paramètres', 'A propos' et 'Panier' (pour la billetterie) sont également accessibles depuis la page 'Accueil'. Une page pour effectuer les paiements et recevoir son ticket pour participer aux évènements est également accessible depuis la page du 'Panier' avec un bouton pour confirmer les articles choisis. La page 'Paramètres' dans certaines catégories est également accessible depuis la page 'Profil'. Certaines pages ne sont accessibles que pour certains utilisateurs comme 'Gestion Monuments' qui n'est accessible que par les propriétaires de monuments ou alors 'Communication pour Collaboration' qui est accessible à la fois par les propriétaires de monuments et les prestataires mais pas par les visiteurs du site.

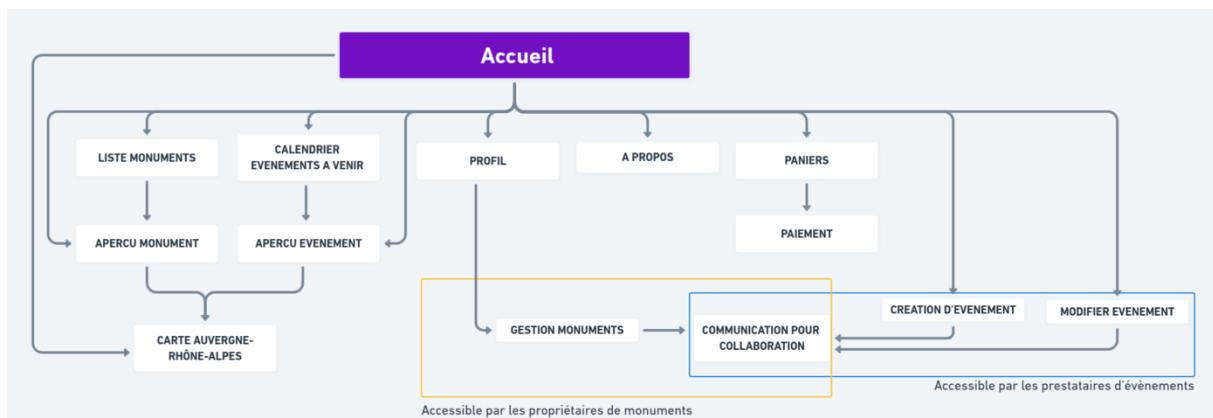


Schéma 6 - Plan du site simplifié

REALISATION DU PROJET

ÉTUDE DE FAISABILITE

Pour réaliser l'étude de faisabilité de notre projet, nous avons, tout d'abord, listé tous les langages/technologies que nous pouvions utiliser et décidé lesquels seraient conservés pour le développement (Annexe II). Puis, nous avons défini le modèle de base de données, le système associé et les langages pour l'implémentation dans l'application web.

Notre application **Mona** repose sur une architecture *client/serveur* classique en utilisant des machines virtuelles (Annexe II) afin d'assurer une séparation claire des responsabilités entre la gestion des données (backend) et l'interface utilisateur (frontend).

- **Côté client (Frontend)** : Le frontend est développé en **ReactJS**, une bibliothèque Javascript moderne, afin de garantir une interface utilisateur dynamique. Les pages sont structurées avec **HTML/CSS**, et les interactions asynchrones (comme les appels aux API) sont gérées via TypeScript. Le rôle principal du client est de présenter les données de manière claire, d'interagir avec l'utilisateur et d'envoyer les requêtes nécessaires au serveur.
- **Côté serveur (Backend)** : Le backend est responsable du traitement des données et des requêtes utilisateurs via une API REST. Il utilise **PHP** comme langage principal pour nos connaissances déjà acquises, sa robustesse et sa compatibilité avec le projet. Les données sont stockées dans une base relationnelle **PostgreSQL**, choisie pour ses performances dans la gestion de données et surtout pour notre affinité avec l'outil.
- **Communication client/serveur** : Les échanges entre le client et le serveur se font via des API REST sécurisées, utilisant le protocole **HTTPS** pour garantir la confidentialité et la sécurité des données transmises.

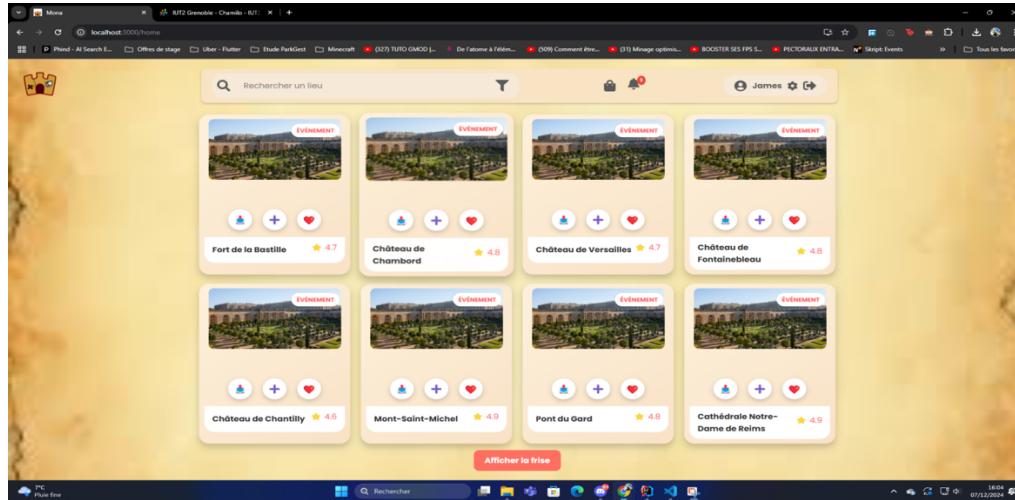
Dans le cadre du développement de l'application **Mona**, certaines technologies ont été écartées en raison de leur inadéquation avec les besoins du projet ou du manque d'expertise de l'équipe. Par exemple, MongoDB et les bases de données NoSQL ont été jugées inadaptées pour gérer les relations complexes entre les données, comme celles entre les monuments et les événements. De même, Node.js, bien qu'offrant des solutions modernes et scalables, n'a pas été retenu en raison d'une maîtrise insuffisante au sein de l'équipe. Ces choix ont été faits pour privilégier la stabilité, réduire les risques et s'appuyer sur des technologies maîtrisées, garantissant ainsi la meilleure mise en œuvre possible pour le projet.

Ainsi, ReactJS est la seule technologie pour laquelle nous avons un membre de l'équipe avec une expérience un peu plus avancée, ce qui nous permet de minimiser les éventuelles difficultés techniques liées à son utilisation. Les autres choix technologiques reposent sur des langages et outils bien maîtrisés par l'ensemble de l'équipe. Par ailleurs, dans le cadre de la SAE, les membres de l'équipe développent progressivement leurs compétences en ReactJS afin d'assurer une mise en œuvre efficace en temps et en heure.

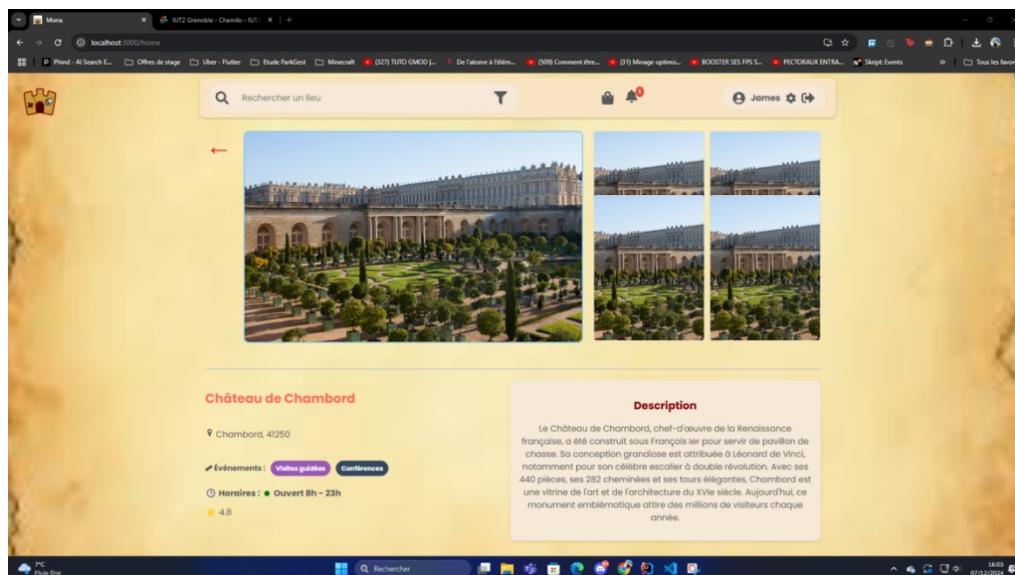
TEST DE L'UTILISATION DE REACTJS

Pour vérifier la faisabilité de notre application et de son architecture, nous avons demandé à deux membres de l'équipe de réaliser un prototype en ReactJS. Leur mission consistait à reproduire quelques pages et composants de notre application : l'affichage listé des monuments et des activités, l'affichage détaillé d'un monument et enfin implémenter une fonction de recherche fonctionnelle avec la base de données. Cet exercice avait un double objectif :

- 1. Tester la faisabilité de reproduction des maquettes que nous avions conçues.**
 - **Résultat :** L'équipe a réussi à reproduire l'interface souhaitée, y compris la fonctionnalité de recherche par nom de monument. Cela confirme que nos maquettes sont réalisables avec les outils choisis.
- 2. Évaluer si l'utilisation de ReactJS est forcément nécessaire.**
 - **Résultat :** oui, ReactJS sera très utile pour la création de nos interfaces dynamiques avec des composants réutilisables et une infrastructure de code moderne, stable et évolutive.



Capture 1 - Page d'accueil



Capture 2 - Page d'un monument

Cependant, le développement de l'application ReactJS a rapidement saturé le quota disponible après la création de seulement quelques fichiers “**.tsx**” de bases sur les stations de l'IUT. Pour surmonter cette contrainte, deux solutions ont été identifiées :

- Effectuer le développement directement sur les machines virtuelles avec un espace de stockage plus important, en utilisant une connexion SSH et un éditeur comme Visual Studio Code.
- Développer sur des machines personnelles, puis transférer les fichiers via GitLab pour centraliser le code.

Grâce à cette expérimentation, on a pu identifier un nouveau risque mais cela a quand même permis de valider nos choix technologiques tout en identifiant un point d'attention majeur pour la gestion du stockage lors du développement futur.

TEST DE L'UTILISATION DE GITLAB

Dans le cadre du projet **Mona**, GitLab a été imposé comme plateforme de gestion des versions et de collaboration. Bien que cet outil offre de nombreuses fonctionnalités avancées pour les équipes de développement, son utilisation a présenté quelques défis.

Les tests ont révélé que GitLab peut être efficacement utilisé en combinaison avec des environnements de développement modernes tels que **Visual Studio Code**, qui intègre une interface graphique pour la gestion des dépôts Git. Cette solution permet de simplifier les opérations de base (commits, pushes, pulls, etc.) via une interface, réduisant ainsi la nécessité d'utiliser exclusivement la ligne de commande. Cependant, dans des environnements restreints comme les salles informatiques de l'IUT, l'installation et la configuration de ces outils peuvent être limitées, obligeant parfois à recourir à Git en ligne de commande.

Malgré les défis initiaux liés à l'utilisation de GitLab, les solutions mises en place, notamment l'utilisation de VS Code et la formation interne, ont permis de maximiser son efficacité tout en répondant aux contraintes de l'environnement imposé.

STRUCTURE MATERIELLE ET TECHNOLOGIQUE DE NOTRE SITE

Pour mettre en œuvre notre application web, nous avons le choix entre deux architectures réseau principales, qui déterminent les liens entre les différents outils et technologies utilisés. Ce choix impacte directement la configuration des serveurs et le déploiement.

POSSIBILITE 1 : SERVEUR ASSR DE L'IUT2

Avantages	Inconvénients
Séparation claire des parties de l'architecture (frontend/backend et base de données)	Non possibilité d'avoir un réel certificat sécurisé.
Amélioration de la scalabilité et de la sécurité (la base de données n'est pas exposée au public)	Obligé de maintenir le serveur à jour et dépendant de l'administrateur réseau (M. Bonnaud)
On peut s'entraîner en appliquant nos connaissances des exercices sur VM.	On ne peut pas y ajouter un nom de domaine et y accéder facilement en dehors de l'IUT2

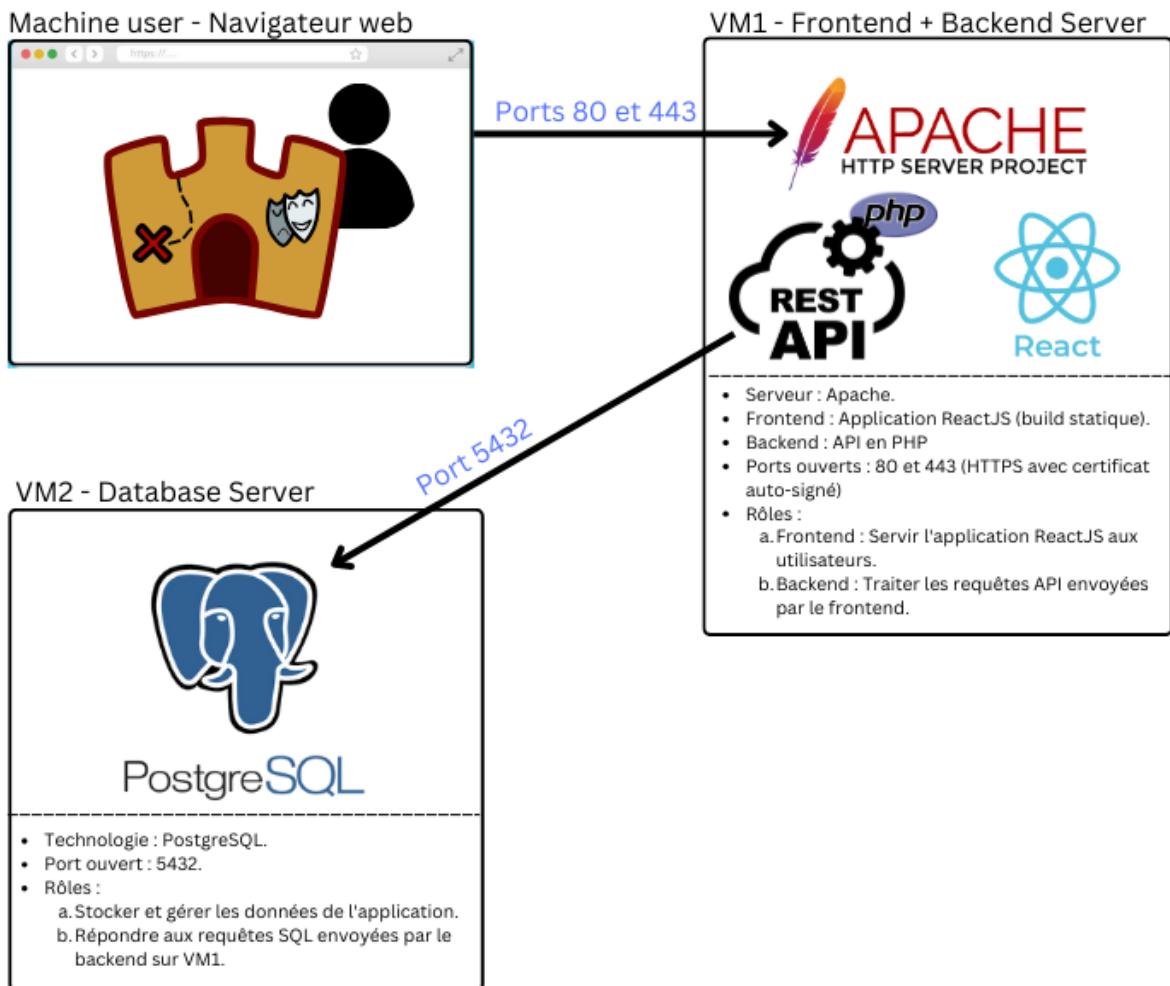


Schéma 7 - Schéma d'implémentation 1

POSSIBILITE 2 : SERVEUR DANS UN CLOUD IONOS

Avantages	Inconvénients
Possibilité d'avoir un vrai certificat signé SSL/TLS	C'est un service payant
Accès public depuis n'importe où (utile pour ajouter aux portfolios) et ajouter un nom de domaine	Performance du serveur faible pour les premiers prix (1Go de RAM)
	Pas de sécurité déjà mis en place

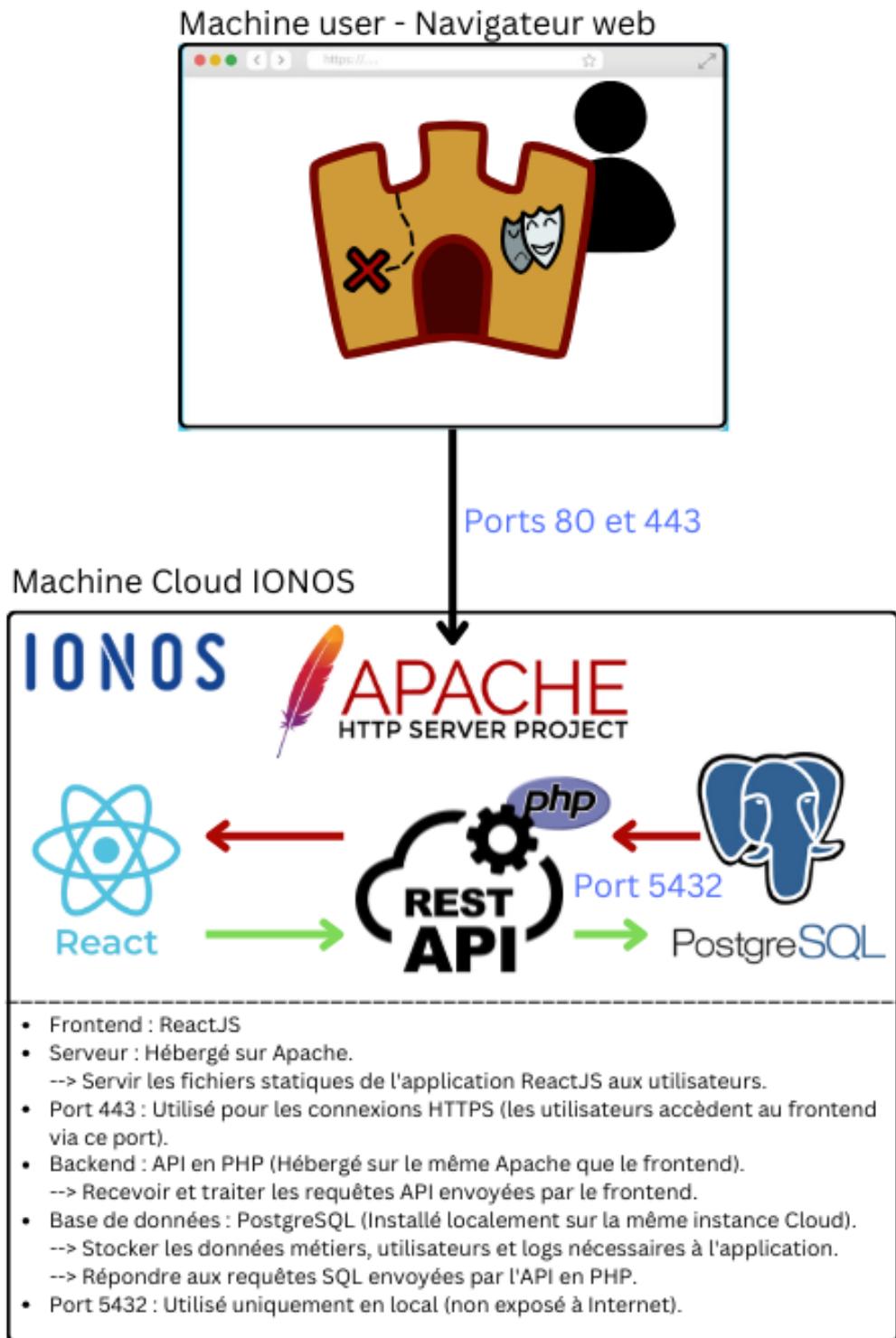


Schéma 8 - Schéma d'implémentation 2

Après réflexion avec l'équipe, nous allons utiliser les serveurs de l'IUT. En effet, nous avons 2 personnes dans notre équipe en parcours B qui ont déjà bien configuré les deux machines virtuelles (test de faisabilité effectué avec ASSR). De plus, nous ne souhaitons pas prendre le risque d'un manque de puissance sur les serveurs IONOS (tableau contraintes et risques). Le manque de puissance et le prix est un réel problème avec les serveurs de IONOS.

EFFECTUE ET RESTANT A FAIRE

Pour garantir une gestion efficace du projet, la planification joue un rôle central. Cette phase implique de suivre en permanence l'état d'avancement des tâches tout en évaluant ce qu'il reste à accomplir.

SUIVI DE L'EFFECTUE (PHASE 1 & 2)

Au cours des phases 1 et 2, plusieurs livrables importants ont été réalisés, constituant les fondations nécessaires pour la suite du projet :

- Étude des besoins utilisateurs : recueil des attentes et analyse des cas d'utilisation.
- Spécifications fonctionnelles et techniques : définition précise des fonctionnalités attendues et des contraintes techniques.
- Conception du modèle de données : réalisation du diagramme de classes et des schémas entité-association pour structurer la base de données.
- Validation des critères ergonomiques : élaboration des maquettes des interfaces utilisateur en tenant compte des principes de Bastien et Scapin.
- Finalisation de la documentation sur les spécifications du projet.
- Analyse des risques : identification des points critiques et définition des stratégies d'atténuation.
- Planification des étapes de développement : création du planning global avec l'identification des jalons majeurs.
- Validation des dossiers techniques avec les parties prenantes.

ANALYSE DU RESTANT A FAIRE (PHASE 3)

La prochaine étape du projet est centrée sur la réalisation complète de l'application web, accompagnée d'une extraction efficace des données. Nous avons 3 semaines. Voici les grandes étapes qui restent à accomplir :

Semaine 1 (6 au 10 janv)	Semaine 2 (13 au 17 janv)	Semaine 3 (20 au 24 janv)
Extraction des données Chef : Loan Avec : Raphaël	Développement Chef : Yani Avec : Joshua & Nils & Mathias	Déploiement Chef : Nils Avec : Yani
Tests & Supervision Chef : Raphaël Avec : Vladimir & Loan		Nettoyer GIT Chef : Mathias Avec : Raphaël

- Développement : réalisation des interfaces et création des fonctionnalités principales.
- Extraction des données : extraction des données et mise en place dans PostgreSQL.
- Tests : validation des fonctionnalités et corrections. Travail parallèle avec l'équipe de développement.
- Supervision : donne des conseils sur le rendu effectué par les développeurs.
- Déploiement : mise en ligne et vérifier maintenabilité de l'application.
- Nettoyer GitLab : mettre au propre GitLab avec les bons README et architecture.

ANNEXES

CONTRAINTES ET RISQUES

Type de Contrainte	Contrainte	Risque	Mitigation
Budgétaire	Extraction à partir d'Open Data	Informations inexactes ou manquantes	Identifier des sources alternatives fiables
Réglementaire	Respecter le RGPD	Sanctions administratives (amendes, perte de données)	Suivi strict du RGPD, formation continue et audits réguliers
Technique	Utilisation de GitLab	Non maîtrise de l'outil	Backup régulier, organisation des noms et branches
		Surcharge de l'application web	Utiliser PostgreSQL pour les transactions. Ou répartition des tâches sur plusieurs serveurs
		Utilisation de ReactJS et non maîtrise	Faire des formations en ligne pour maîtriser l'outil
		Mauvaise communication avec les visiteurs	Effectuer des tests utilisateurs, recueillir régulièrement des feedbacks
		Failles de sécurité dans l'application web	Appliquer les différentes techniques vues en TP de système réseau
		Hébergement chez IONOS pour avoir un accès à distance	Utiliser finalement les serveurs de l'IUT

Annexe I - Tableau de contraintes et risques

ETUDE DE FAISABILITE

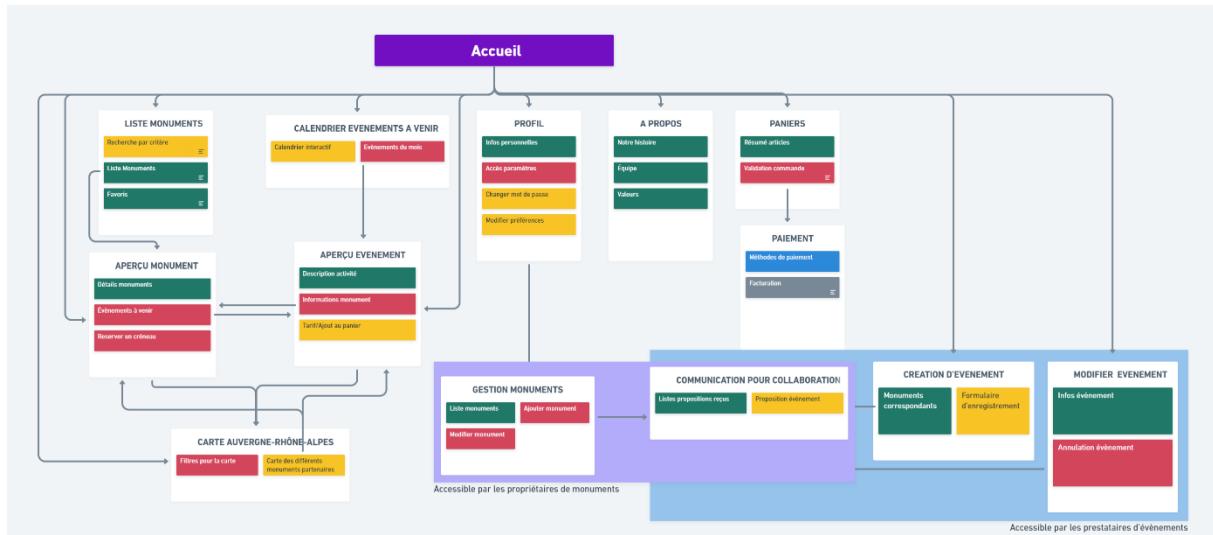
Langage/Bas e de données	Rôle	Description/Particularités	Avantages/Points forts	Inconvénient s	Retenu?
HTML	Front-end	Le langage indispensable pour coder une application web.	Toute notre équipe est formée dessus.	Aucun	OUI, il est indispensable pour créer une application web.
CSS/SCSS	Front-end	Le langage basique pour styliser une page web.	Toute notre équipe est formée dessus.	Aucun	OUI, il n'y a pas d'équivalent viable pour nous. SCSS nous permettra de pousser encore plus loin mais nécessitera une légère formation.
Javascript/Typescript	Front-end	C'est un langage moderne permettant de rendre interactives des pages et dont sont basés la plupart des frameworks et librairies populaires.	La majorité de notre équipe a des bases dessus. C'est un langage asynchrone (plusieurs tâches en simultané exécutables) ce qui le rend très rapide à l'exécution.	Aucun	OUI, Javascript sera utilisé pour rendre nos pages et boutons interactifs. De plus, nous utiliserons ReactJS est un framework JS.
ReactJS	Back-end /Gestion bases de données	ReactJS est un framework Javascript permettant de créer des interfaces utilisateurs et de les automatiser bien plus facilement qu'avec simplement HTML.	Framework moderne qui permettra de simplifier le code HTML en créant des fonctions qui placeront du code HTML en fonction de certains paramètres.	La majorité de notre équipe n'a jamais utilisé cette librairie.	OUI, notre équipe étant formé dessus, nous pourrons éviter des risques en utilisant un autre modèle de base de données comme NoSQL.
PHP	Back-end /Gestion bases de données	PHP est un langage de programmation web servant à faire des interfaces dynamiques côté serveur.	Toute l'équipe est formée dessus.	Langage un peu daté et synchrone (une seule tâche en même temps) ce qui le rend très lent par	OUI, l'API sera réalisée en PHP mais le reste sera réalisé principalement avec ReactJS.

				rapport à Javascript.	
PostgreSQL	Back-end/Gestion bases de données	PostgreSQL est un langage qui permet de gérer des bases de données relationnels.	Toute l'équipe est formée dessus et a une bonne compréhension des schémas	Aucun	OUI, notre équipe étant formé dessus, nous pourrons éviter des risques en utilisant un autre modèle de base de données comme NoSQL.
MongoDB (NoSQL) 8.4	Back-end/Gestion bases de données	NoSQL est un modèle de bases de données non relationnelles.	L'approche NoSQL est plus adaptée au développement d'application web.	Aucun membre de l'équipe a utilisé une approche NoSQL ou concernant des bases de données non relationnelles .	NON, notre équipe n'a jamais travaillé avec ce type de bases de données, nous utiliserons plutôt PostgreSQL pour éviter d'engendrer des risques.
Node.js	Back-end	Node.js est un environnement d'exécution qui permet d'exécuter du Javascript côté serveur.	Node.js permet de créer des applications web qui puissent s'élargir facilement en cas d'augmentations des fréquentations et de l'utilisation de notre site. Node.js permet aussi de gérer les bases de données NoSQL comme MongoDB.	Aucun	NON, notre équipe n'a jamais travaillé avec ce type de bases de données, nous utiliserons plutôt PostgreSQL pour éviter d'engendrer des risques.

Annexe II - Tableau avec les différents langages

PLAN DU SITE

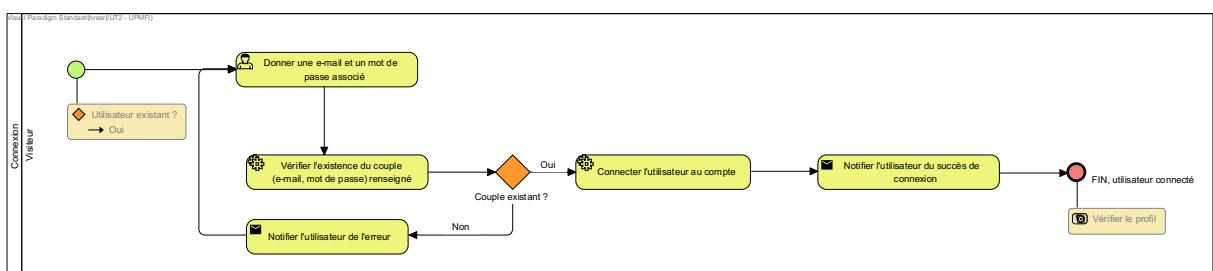
Pour permettre une compréhension des composants des pages nous avons réalisé un code couleur : **Bouton->Rouge** **Texte->Vert** **ActionUtilisateur/Formulaire->Jaune** **ActionBackEnd->Gris**



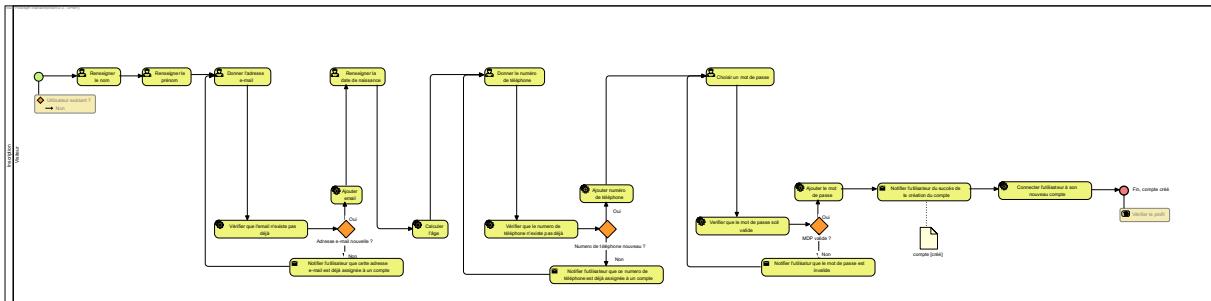
Annexe III - Plan du site détaillé

Comme le montre notre plan du site, la page d'Accueil permettra d'accéder à la majorité des pages hormis le paiement. Les pages 'Calendrier événements à venir' et 'Liste monuments' permettent à l'utilisateur de parcourir nos différents partenaires et activités proposés pour faire son choix. La page 'Aperçu d'évènement' permet à l'utilisateur d'en apprendre plus sur un évènement et de pouvoir l'ajouter au panier pour réserver sa place. Certaines pages ne sont accessibles que pour certains types d'utilisateurs comme les pages 'Création d'évènement' et 'Modifier évènement' qui ne sont pourront être accessibles que par les prestataires d'évènements, ou encore la page 'Communication pour Collaboration' qui ne sont pas consultables par les visiteurs du site car elles permettront à nos partenaires propriétaires de monuments ou prestataires d'évènements de s'organiser pour la création d'évènements en collaboration.

BPMN



Annexe IV - Connexion



Annexe V – Inscription

SCHEMA RELATIONNEL

Utilisateur(idUtilisateur, nom, mdp, role)

Propriétaire(idProprietaire, certificat, #idUtilisateur)

Prestataire(idPrestataire, roleEntr, #idUtilisateur, #idEntreprise)

Reservation(idReservation, date, numReservation, #idEvenement, #idUtilisateur)

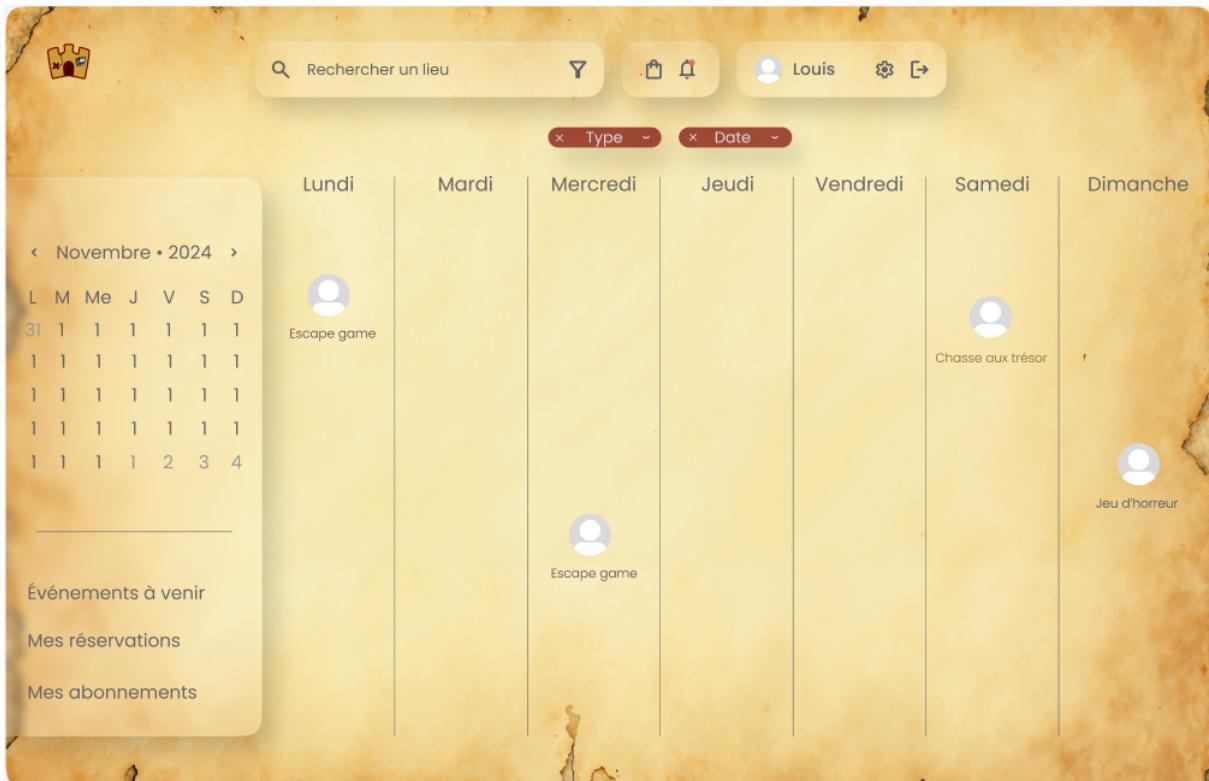
Entreprise(idEntreprise, nomEntreprise, ville, cleEntreprise, description, siret, note, #idResponsable)

Monument(idMonument, nom, type, siecle, surfaceInterieur, surfaceExterieur, histoire, photoMonu, capacite, disponibilite, evenementsRefuses, ressourcesDispo, horaires, adresse, departement, commune, description, dateAjout, derniereMAJ, remarque, note)

Possede(#idMonument, #idProprietaire)

Annexe VI - Schéma relationnel

MAQUETTES



Annexe VII - Maquette du calendrier

Deux écrans de la plateforme de réservation montrant le processus de création d'un événement. L'écran de gauche montre les champs de saisie pour le titre, la date, la description, le nombre de personnes et le type d'événement, avec un bouton "Envoyer". L'écran de droite montre une boîte d'erreur "Oups..." indiquant que le propriétaire n'accepte que 8 personnes au maximum, avec un champ "Message" et un bouton "Faire une proposition".

Annexe VIII - Maquette ajouter événement

Trois écrans de la plateforme montrant les paramètres du profil. Chaque écran affiche les informations de base (nom, statut), les options de réservation et un bouton "Modifier ce profil". Les écrans sont étiquetés "Compte", "Compte + prestataire" et "Compte + prestataire".

Annexe IX - Maquette paramètre du profil

The screenshot shows a travel application interface. At the top, there is a search bar with the placeholder "Rechercher un lieu" and a user profile icon for "Louis". Below the search bar is a large, stylized profile picture of a monument, possibly the Eiffel Tower, with a yellow vertical bar running through it. To the right of the profile picture is a "Description" section containing placeholder text in French.

Fort de la Bastille

8 Quai Stéphane Jay

Événements Escape game Chasse aux trésors Jeu d'horreur

4,91

Horaires Ouvert 8h - 23h

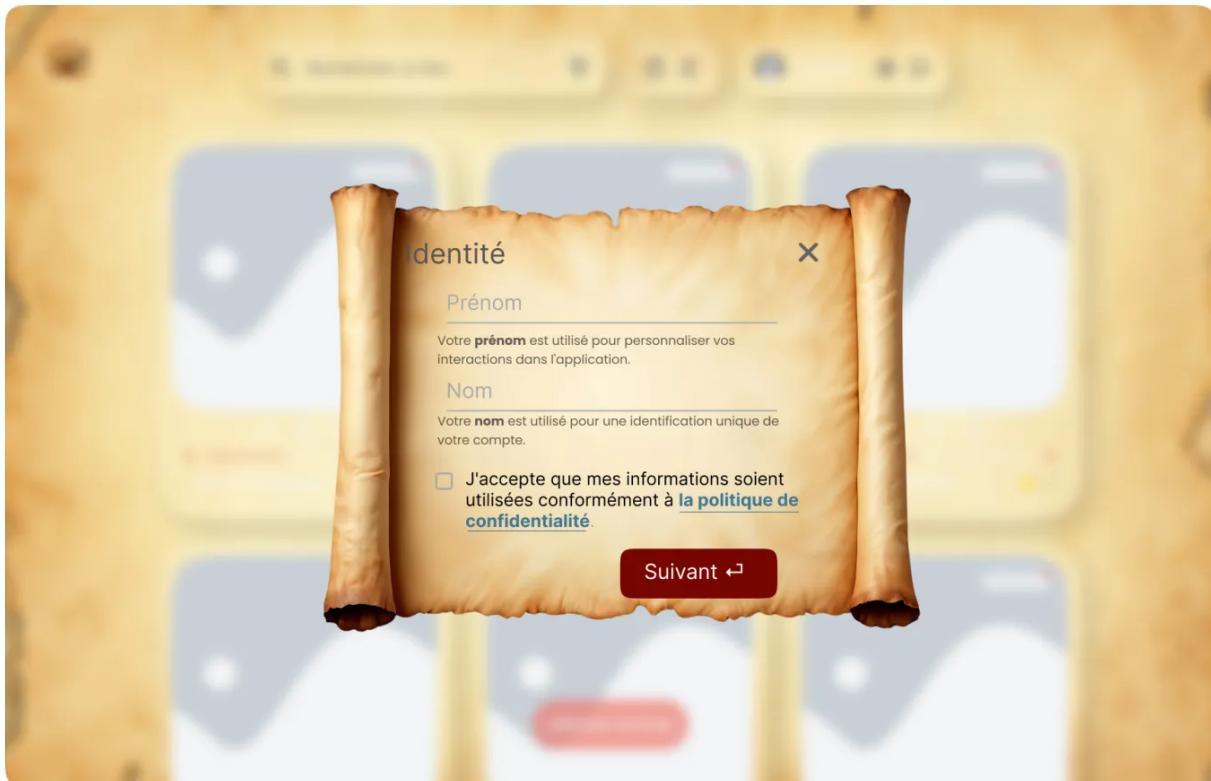
Voir les événements Reserver une visite

Afficher sur la frise

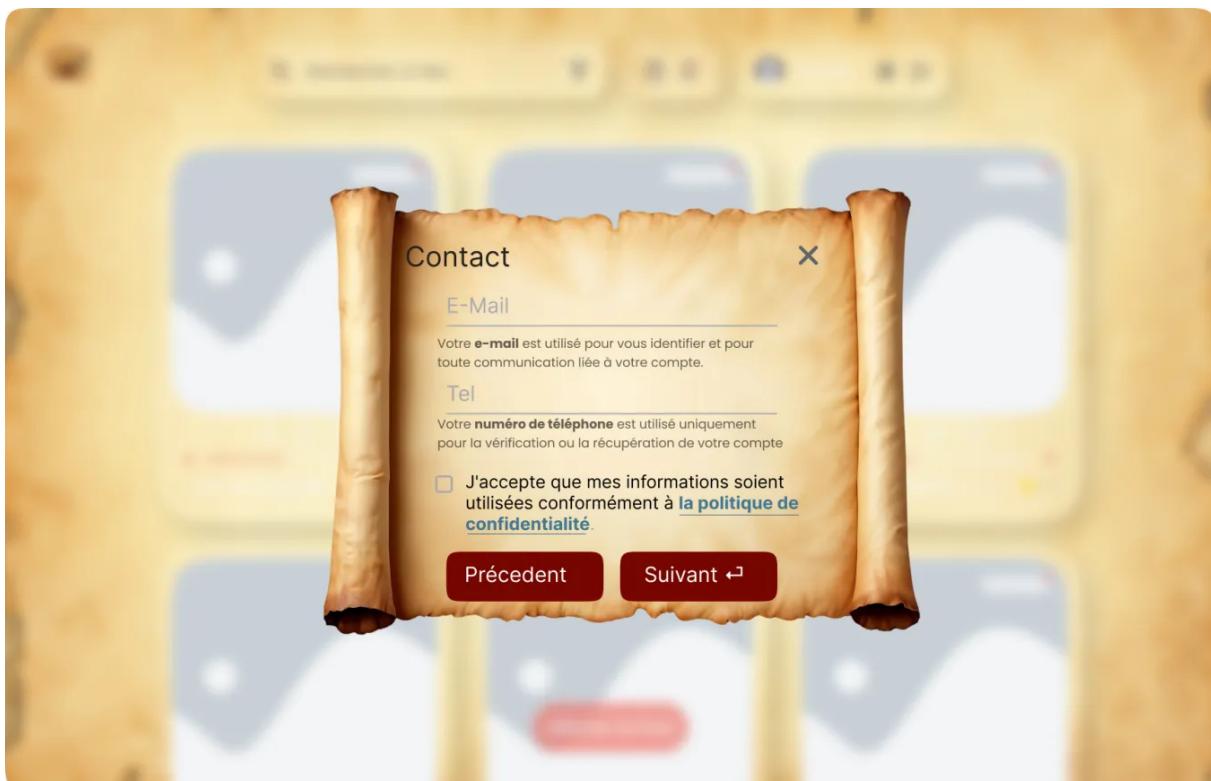
Annexe X - Maquette affichage monument

The screenshot shows two versions of a shopping cart interface. The left version is titled "Panier • vide" and displays four placeholder cards for monuments. The right version is also titled "Panier" and shows two items: "Visite simple • 19,89€" and "Chasse au trésor • 39,89€", each with a quantity selector and a delete icon.

Annexe XI - Maquette panier



Annexe XII - Maquette identité



Annexe XIII - Maquette pour contact

Politique de Confidentialité

Date d'entrée en vigueur : 01/12/2024
La gestion des données personnelles constitue un enjeu central pour notre organisation. À travers cette politique de confidentialité, nous détaillons les mécanismes de collecte, d'utilisation et de protection des informations sensibles, en conformité avec les standards, notamment le Règlement Général sur la Protection des Données.

1-Données collectées et finalités associées :

Dans le cadre de l'inscription et de l'utilisation de notre application, les informations suivantes peuvent être collectées :

- Identité (nom et prénom) : Permet une identification unique et nécessaire à la personnalisation des services.
- Adresse électronique : Sert d'identifiant principal et assure la communication des notifications liées à l'utilisation de l'application.
- Date de naissance : Confirme l'éligibilité de l'utilisateur selon les critères d'âge requis et contribue à l'adaptation des fonctionnalités.
- Mot de passe : Assure la sécurisation des accès aux données personnelles stockées.

2-Modalités de traitement des données :

Les traitements appliqués visent les objectifs suivants :

- Authentification et gestion des comptes utilisateurs.
- Sécurisation de l'accès aux services via des mécanismes de chiffrement et d'identification.
- Personnalisation des interactions pour mieux répondre aux attentes des utilisateurs.
- Communication proactive concernant les mises à jour ou tout changement relatif aux services.

3-Droits des utilisateurs :

En tant que détenteur des données, vous bénéficiez de droits spécifiques :

- Accès : Obtenir une copie des informations stockées vous concernant.
- Rectification : Corriger les données inexacts ou incomplètes.
- Effacement : Demander la suppression des informations personnelles dans des conditions définies.
- Portabilité : Recevoir vos données dans un format structuré, couramment utilisé et lisible par une machine.

Pour exercer vos droits, veuillez adresser vos demandes à : support-data@monadev.fr.

Les données personnelles seront conservées uniquement pendant la période nécessaire à la réalisation des objectifs précités ou conformément aux obligations légales en vigueur. Une fois la suppression du compte initiée par l'utilisateur, les données seront effacées dans un délai raisonnable.

Annexe XIV - Maquette politique de confidentialité