CESI
ÉCOLE D'INGÉNIEURS

CESI-vents

Analyse du cahier des charges fonctionnel

Groupe n°2
Mars 2025

Table des matières

Introduction

Contexte

Le Bureau des Étudiants (BDE) du CESI souhaite développer une plateforme en ligne nommée CESI-vents.

L'application servira à promouvoir et gérer les événements organisés par les clubs étudiants et le BDE.

Cette plateforme devra être ergonomique, responsive, on favorise le mobile first et inclure un système de billetterie ainsi qu'une gamification pour inciter les étudiants à participer aux événements proposés par les différents clubs par le biais de bons, coupons et points de fidélité.

Objectifs

À l'issue de ce projet, il est essentiel de garantir au minimum l'ensemble des fonctionnalités souhaitées par le client, dont le Product Owner.

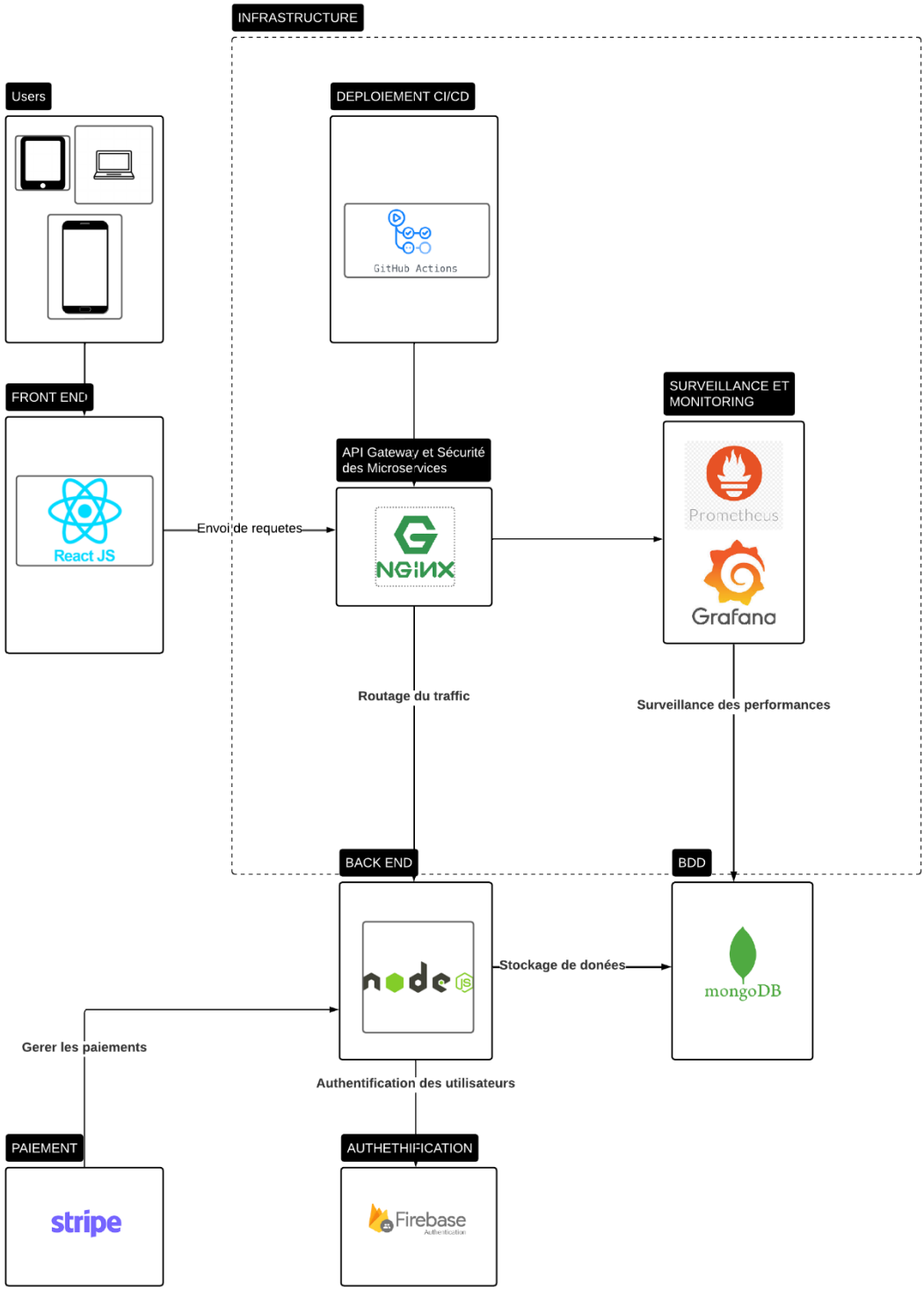
Les objectifs que le Bureau des Étudiants (BDE) du CESI sont les suivants :

1. **Promouvoir les activités des clubs** : permettre aux clubs du CESI de publier et promouvoir leurs événements (sportifs, caritatifs, e-sport, ludiques, etc.), consultables par tous les étudiants et éventuellement par des personnes extérieures.
2. **Proposer une billetterie en ligne** : permettre la réservation et le paiement des événements à capacité limitée ou payants.
3. Faciliter la gestion des événements :
    - Offrir aux clubs un espace d'administration pour gérer leurs événements et suivre les inscriptions.
    - Permettre au BDE d'avoir une vue d'ensemble des événements, des finances et des statistiques.
4. Assurer une expérience utilisateur fluide :
    - Interface responsive (mobile first).
    - Ergonomie et clarté des pages.
    - Processus d'inscription et de réservation simplifié.
5. Intégrer un système de gamification :
    - Attribution de points de fidélité pour encourager la participation aux événements.
    - Possibilité d'échanger ces points contre des récompenses (goodies, réductions, etc.).
    - Gestion des codes promos et réductions pour les adhérents BDE.
6. Assurer la sécurité et l'authentification :
    - Accès différencié selon les rôles (étudiants, adhérents, responsables de clubs, administrateurs BDE).
    - Sécurisation des transactions et des données personnelles.
7. Offrir un suivi et des statistiques avancées :
    - Vue d'ensemble pour le BDE avec suivi des événements, des inscriptions et des finances.
    - Outils pour analyser la fréquentation et optimiser les événements futurs.

L'application vise donc à centraliser et simplifier la gestion des événements étudiants tout en encourageant l'engagement de la communauté CESI.

Architecture

Architecture technique



La plateforme CESI-vents sera déployée sur plusieurs environnements : développement, recette et production.

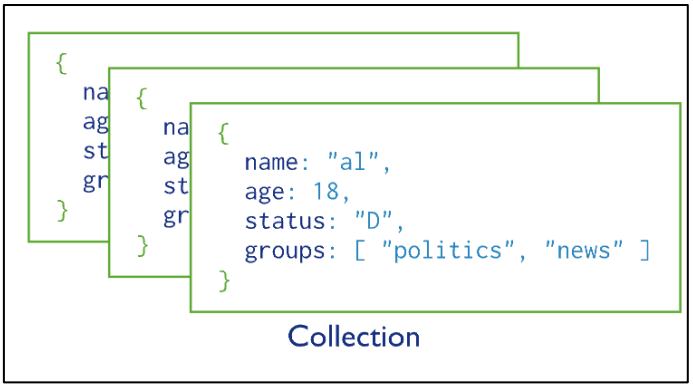
En matière de sécurité, la plateforme adopte plusieurs mesures, telles que l'utilisation de OAuth2 pour permettre une authentification sécurisée à nos utilisateurs.

De plus l'architecture la plus adaptée pour nos besoins est l'architecture microservice car elle respecte les conditions suivantes :

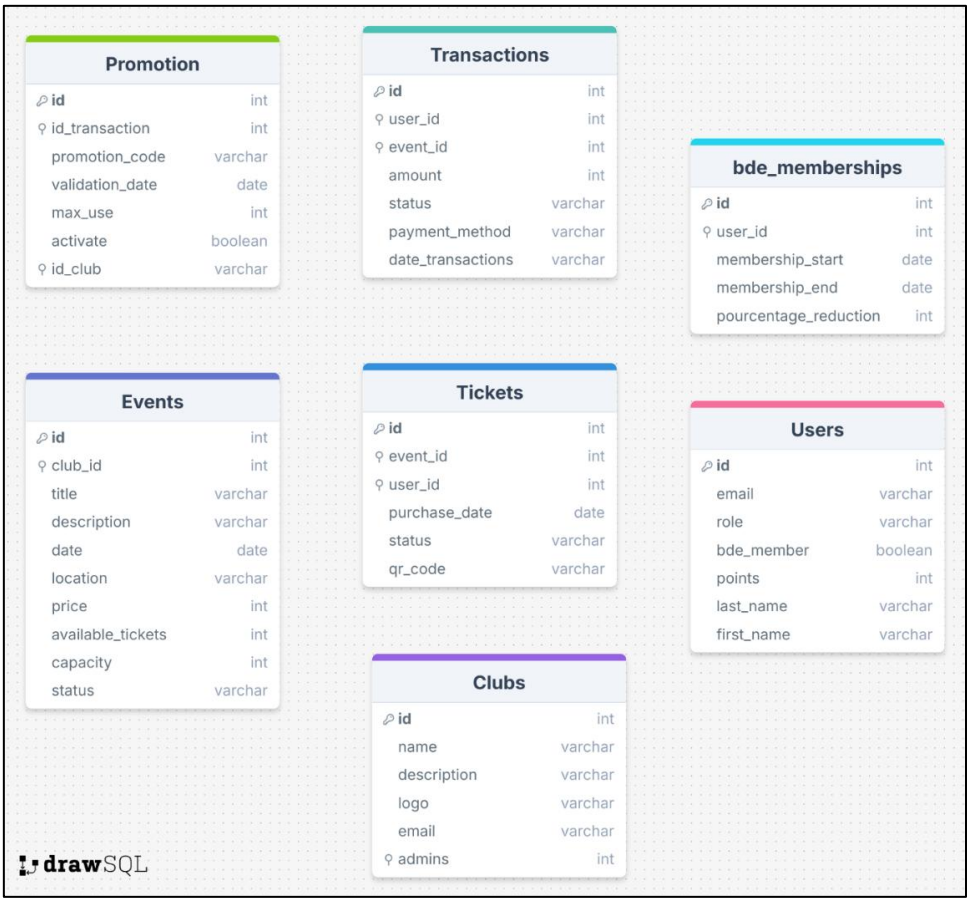
- Les services individuels sont simples à remplacer
- Les services sont conçus pour leur utilité spécifique
- L'architecture facilite le déploiement continu du code

Schéma BDD

Pour notre base de données l'utilisation de MongoDB est le plus adaptées à nos besoins, il faut savoir que MongoDB est une base de données dites SGBD NoSQL, contrairement à une base de données relationnelle, elle ne repose pas sur un schéma rigide. De plus en MongoDB on n'utilise pas des tables mais des collections, une collection un ensemble de document en JSON.



Cette base de données permet de gérer les clubs, les événements, les utilisateurs, les billets, les promotions, les transactions et les adhésions.



1. Users (Utilisateurs)

- id : Identifiant unique de l'utilisateur.
- e-mail : Adresse e-mail de l'utilisateur.
- rôle : Rôle de l'utilisateur.
- bde_member : Indique si l'utilisateur est membre du BDE.
- points : Points de fidélité accumulés par l'utilisateur.
- last_name : Nom de famille de l'utilisateur.
- first_name : Prénom de l'utilisateur.

2. Clubs (Clubs)

- id : Identifiant unique du club.
- name : Nom du club.
- description : Description du club.
- logo : Lien ou chemin vers le logo du club.
- email : Adresse e-mail de contact du club.
- admins : Identifiants des administrateurs du club.

3. Events (Événements)

- id : Identifiant unique de l'événement.
- club_id : Identifiant du club organisateur de l'événement.
- title : Titre de l'événement.
- description : Description de l'événement.
- date : Date de l'événement.
- location : Lieu de l'événement.
- price : Prix du billet pour l'événement.
- available_tickets : Nombre de billets disponibles.
- capacity : Capacité maximale de l'événement.
- status : Statut de l'événement.

4. Tickets (Billets)

- id : Identifiant unique du billet.
- event_id : Identifiant de l'événement associé au billet.
- user_id : Identifiant de l'utilisateur ayant acheté le billet.
- purchase_date : Date d'achat du billet.
- status : Statut du billet.
- qr_code : Code QR associé au billet pour la validation.

5. Transactions (Transactions)

- id : Identifiant unique de la transaction.
- user_id : Identifiant de l'utilisateur ayant effectué la transaction.
- event_id : Identifiant de l'événement associé à la transaction.
- amount : Montant de la transaction.
- status : Statut de la transaction (par exemple, réussie, échouée, en attente).
- payment_method : Méthode de paiement utilisée.
- date_transactions : Date de la transaction.

6. Promotion (Promotions)

- id : Identifiant unique de la promotion.
- id_transaction : Identifiant de la transaction associée à la promotion.
- promotion_code : Code de la promotion.
- validation_date : Date de validité de la promotion.
- max_use : Nombre maximal d'utilisations de la promotion.
- activate : Indique si la promotion est active (booléen).
- id_club : Identifiant du club associé à la promotion.

7. bde_memberships (Adhésions BDE)

- id : Identifiant unique de l'adhésion.
- user_id : Identifiant de l'utilisateur adhérent.
- membership_start : Date de début de l'adhésion.
- membership_end : Date de fin de l'adhésion.
- pourcentage_reduction : Pourcentage de réduction accordé aux adhérents.

Justification Technique

Utilisation du Framework React.js pour le front-end

React offre de nombreux avantages, notamment des composants réutilisables qui facilitent la maintenance et le développement rapide. Grâce au Virtual DOM, les performances sont optimisées en ne rendant que les parties modifiées. Son écosystème riche propose un large choix de bibliothèques comme **Redux**, **React Query** et **TailwindCSS**. De plus, React est compatible avec **Next.js**, permettant un meilleur référencement grâce au SEO et au SSR. Enfin, sa modularité permet de l'utiliser avec **React Native** pour développer des applications mobiles

Pourquoi ne pas choisir Angular ou Vue ?

Angular présente une courbe d'apprentissage plus élevée en raison de l'utilisation de TypeScript et de son approche structurée (services, modules, etc.).

Vue.js est plus léger et accessible, mais il est moins répandu en entreprise et dispose d'un écosystème plus limité.

React.js est le choix optimal pour une application scalable et modulaire, répondant aux besoins en performance et en SEO.

Conteneurisation

Docker assure que l'application fonctionne de la même manière en dev et en prod et simplifie le déploiement multi-environnement.

Nginx

Nginx est un choix robuste pour un reverse proxy performant et sécurisé, particulièrement adapté pour les requêtes HTTP statiques et la gestion avancée du caching. Bien qu'il nécessite une configuration manuelle plus poussée, il excelle dans la gestion du trafic web, notamment pour les fichiers statiques et la compression Gzip.

Traefik est idéal pour les architectures Kubernetes grâce à sa gestion automatique du service discovery et des certificats SSL. Cependant, en dehors de ce contexte, Nginx reste plus performant pour les besoins classiques de reverse proxy.

Kong est conçu pour gérer des API REST avec des fonctionnalités avancées de gestion des API, comme le monitoring, l'analytic et le throttling. Si l'application repose sur de nombreuses APIs et services, Kong peut être pertinent. Toutefois, pour une solution simple avec peu de surcharge administrative, Nginx est plus léger et efficace.

Nginx est le choix optimal pour un reverse proxy performant et sécurisé, sauf si l'architecture devient très orientée microservices, où Traefik ou Kong pourraient être envisagés.

Utilisation de node.js pour le backend

Node.js est idéal pour des applications en temps réel telles que les chats, les notifications et les paiements grâce à son modèle asynchrone et événementiel. Léger et performant, il utilise un seul thread et le modèle event-loop. De plus, il s'intègre facilement avec MongoDB via Mongoose et avec Redis pour le caching, offrant ainsi une solution complète et efficace pour le développement d'applications modernes.

Base de données

MongoDB est une solution flexible pour gérer des données non structurées comme les logs, l'historique ou les profils utilisateurs. Il assure une haute disponibilité grâce à la réplication automatique et offre d'excellentes performances en lecture, notamment avec la mise en cache via Redis.

En revanche, Firebase est une solution clé en main sans maintenance serveur, particulièrement adaptée aux applications mobiles et web nécessitant des mises à jour en temps réel et une intégration facile avec les services Google.

Le choix de MongoDB plutôt que Firebase se justifie par sa flexibilité dans la structuration des données et son efficacité pour l'analyse des informations.

Monitoring

Prometheus et Grafana forment une combinaison puissante pour surveiller les performances et détecter les anomalies. Prometheus collecte et stocke les métriques, tandis que Grafana permet de visualiser ces données de manière intuitive et interactive. Ensemble, ils offrent une solution complète pour le monitoring et l'analyse des systèmes, facilitant ainsi la détection proactive des problèmes et l'optimisation des performances.

Déploiement CI/CD avec Github Actions

L'adoption de CI/CD permet d'automatiser les tests, les builds et les déploiements, assurant ainsi une livraison continue et fiable des applications.

GitHub Actions est particulièrement avantageux pour son intégration fluide avec Docker. Ils permettent de créer des workflows automatisés pour les tests unitaires, le linting et les builds, simplifiant ainsi le processus de développement et de déploiement.

Diagramme de cas d’utilisation

Définition

Les diagrammes de cas d'utilisation décrivent les fonctions générales et la portée d'un système. Ces diagrammes identifient également les interactions entre le système et ses acteurs.

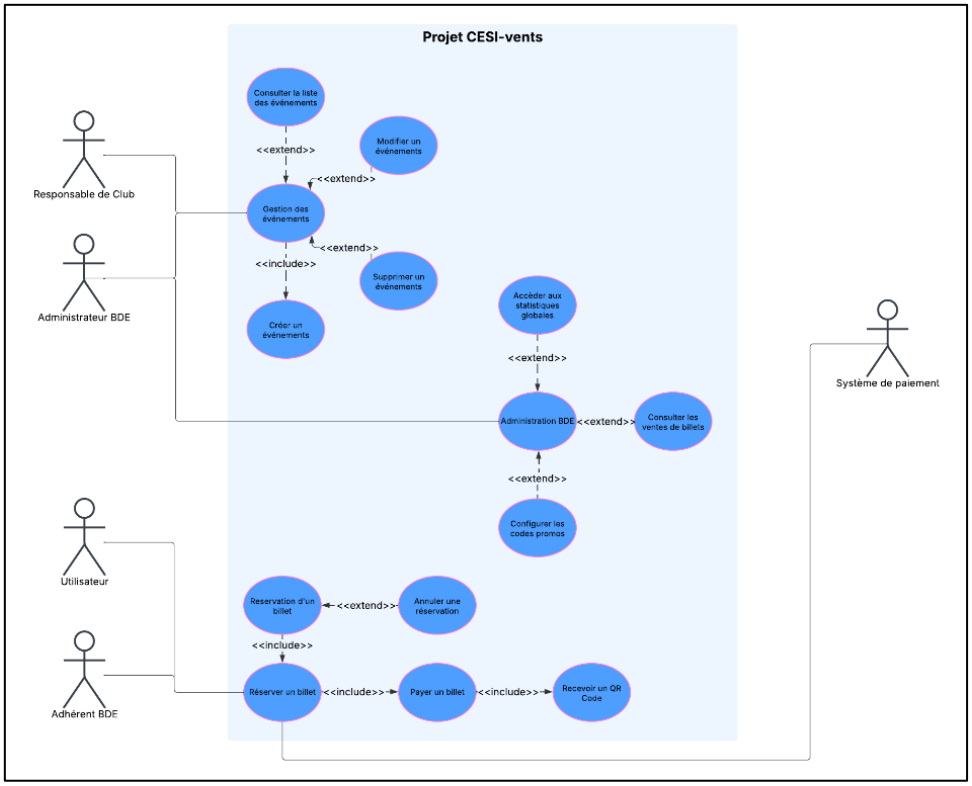


Diagramme de cas d’utilisation de CESI-vents

Diagramme d’activité

Définition

Un diagramme d'activité fournit une vue du comportement d'un système en décrivant la séquence d'actions d'un processus. Les diagrammes d'activité sont similaires aux organigrammes de traitement de l'information, car ils montrent les flux entre les actions dans une activité. Les diagrammes d'activité peuvent, cependant, aussi montrer les flux parallèles simultanés et les flux de remplacement.

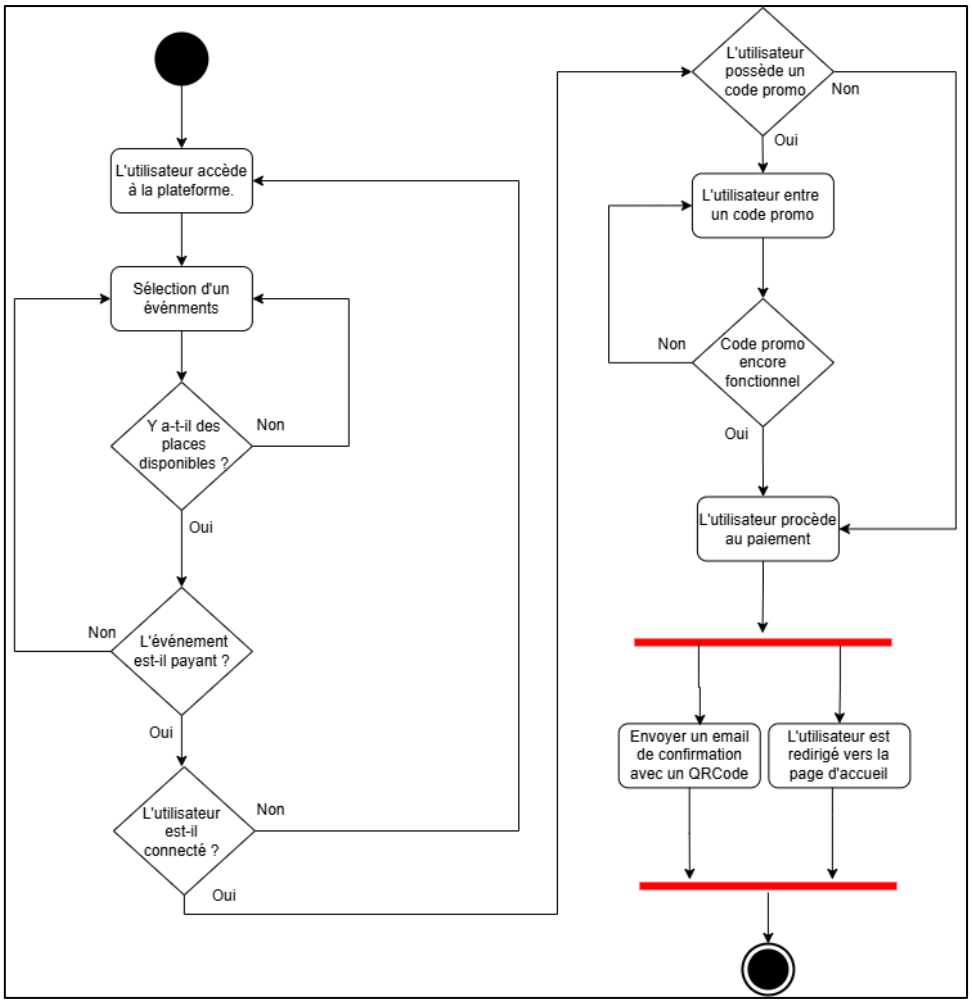


Diagramme d’activités CESI-vents

Intégration continue (CI)

Définition

Selon IBM, l'intégration continue est un processus de développement logiciel dans lequel les développeurs intègrent régulièrement du nouveau code dans la base de code, tout au long du cycle de développement.

Il est alors question de configuration unique, qui varie selon un projet, l'équipe qui le met en place ou encore l'entreprise.

Mise en place

Plusieurs solutions permettent de mettre en place ce type d'intégration telles que :

- GitLab CI/CD, outil intégré à GitLab permettant d'automatiser le build, les tests et le déploiement des applications. C'est un regroupement d'outils propre à GitLab, efficace lorsqu'un projet repose sur cet hébergeur.

- GitHub Actions, solution intégrée cette fois-ci à GitHub, fonctionnant sensiblement de la même manière que la précédente, en étant plus simple d'utilisation. Elle peut toutefois, se révéler plus limitée lorsque l'on souhaite gérer les rôles (accès distincts) et permissions dans les pipelines.

Dans notre cas, il s'agira d'utiliser GitHub Actions, étant donné que cette solution est directement intégrée à notre système de gestion de code. Ainsi, nous pourrons accélérer le développement logiciel car chaque intégration de code sera contrôlée par l'outil.

Plan d'intégration continue pour CESI-vents

Pour garantir une intégration continue fluide et efficace, nous avons opté pour GitHub Actions, qui est directement intégré à notre système de gestion de code sur GitHub. Ce choix nous permet d'automatiser les étapes essentielles du développement logiciel, tout en assurant un suivi rigoureux des modifications apportées à la base de code.

Structure des Branches

Nous adoptons une gestion des branches bien définie afin de garantir la stabilité du projet :

- **main** : Branche stable contenant le code validé et fonctionnel.
- **develop** : Branche principale pour le développement en cours.
- **Feature branches** : Créées pour chaque nouvelle fonctionnalité, puis fusionnées dans develop via des pulls requests.
- **Hotfix branches** : Pour les corrections urgentes de bugs.

Mise en place du Pipeline CI/CD et automatisation

GitHub Actions nous permettra d'automatiser notre pipeline de CI, se déclenchant à chaque nouvelle modification du code. Ce pipeline comprend plusieurs étapes essentielles.

Avant toute exécution des tests, la qualité du code est validée à l'aide d'outils d'analyse statique tels que :

- **ESLint** (pour JavaScript/React) afin de vérifier les bonnes pratiques et la conformité au style de code.
- **Prettier** pour le formatage automatique du code.
- **SonarCloud** pour l'analyse de la qualité du code et la détection des vulnérabilités.

Ces outils permettent de détecter les erreurs potentielles avant même d'exécuter les tests.

Dans un deuxième temps, la phase de build permet de compiler le code et de vérifier sa validité avant toute autre action. Une fois cette étape réussie, différents tests sont exécutés afin de s'assurer du bon fonctionnement du projet.

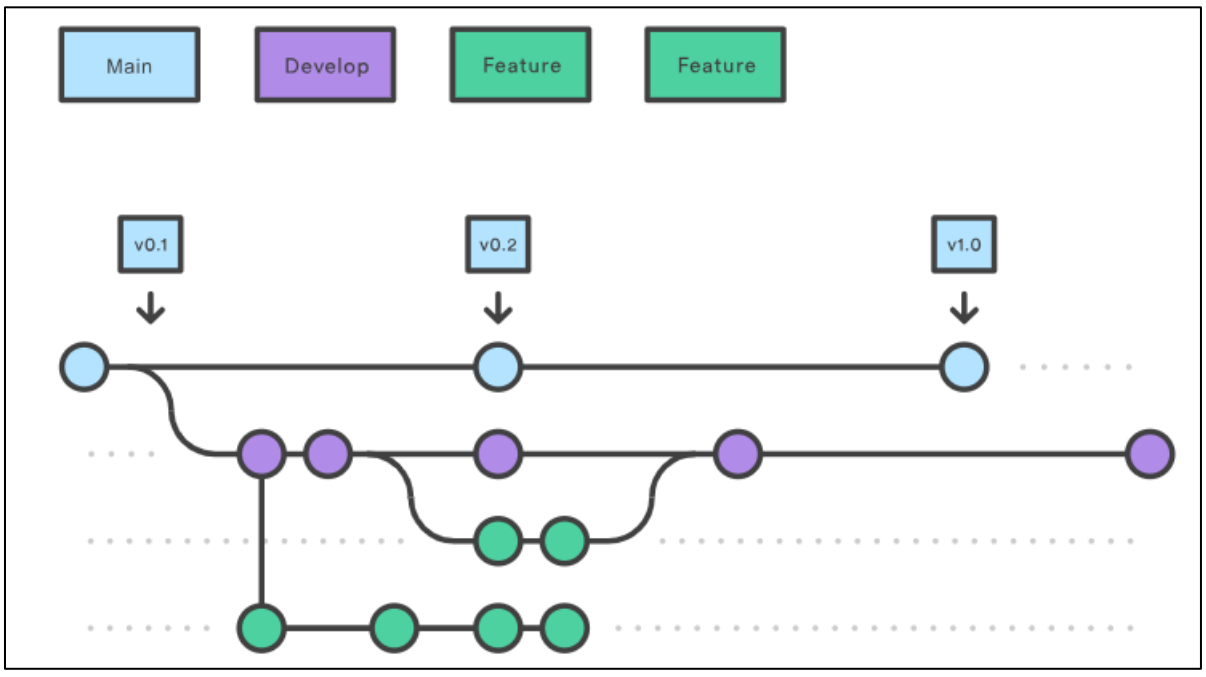
Ces tests jouent un rôle clé dans la validation du code avant son intégration. Nous pouvons mettre en place plusieurs types de tests :

- **Tests unitaires** : Vérifient le bon fonctionnement de chaque composant individuellement, en utilisant Jest et React Testing Library.
- **Tests d'intégration** : Assurent que les différentes parties du projet fonctionnent correctement ensemble.
- **Tests end-to-end (E2E)** : Simulent le parcours utilisateur avec Cypress pour valider l'expérience globale.

Si un test échoue, le pipeline est immédiatement interrompu et une alerte est envoyée aux développeurs.

Ces validations automatiques permettent ainsi de détecter et de corriger d'éventuelles erreurs avant l'intégration du code dans la branche principale. Ces étapes garantissent une validation systématique du code avant son intégration dans la branche principale.

Voici un exemple de Gitflow possible :



Automatisation et Déploiement

Grâce à GitHub Actions, chaque commit ou pull request déclenche automatiquement l'exécution du pipeline. En cas de succès des builds et tests, les modifications peuvent être fusionnées en toute confiance. Si un test échoue, les développeurs sont immédiatement alertés pour corriger les erreurs avant toute mise en production.

Déploiement continu (CD)

Définition

Selon IBM, le déploiement continu (CD) est une méthodologie de développement logiciel visant à automatiser la publication des mises à jour en production dès leur validation.

Il s'agit d'une approche reposant sur une suite de tests automatisés qui évaluent chaque modification en temps réel. Lorsque ces tests sont concluants, les nouvelles fonctionnalités ou corrections sont directement déployées sans intervention humaine.

Grâce à ce processus, le temps entre le développement et la mise en production est considérablement réduit, permettant aux utilisateurs de bénéficier rapidement des évolutions du produit tout en minimisant les risques liés aux erreurs manuelles.

Mise en place

Après avoir étudié deux solutions pour le déploiement continu, nous avons retenu GitHub Actions, conformément à notre choix initial. Cette solution, directement intégrée à notre gestion de code, nous permet d'automatiser efficacement la mise en production des mises à jour validées, garantissant ainsi une évolution continue de l'application tout en minimisant les interruptions.

Automatisation du Pipeline CD

L'automatisation du déploiement suivra plusieurs étapes essentielles afin d'assurer une mise en production fiable et maîtrisée.

Tout d'abord, une phase de préparation et validation permettra de vérifier l'intégrité du code et d'exécuter une dernière série de tests avant son déploiement. Ensuite, une version intermédiaire sera mise en ligne dans un environnement de staging, garantissant ainsi un contrôle approfondi du bon fonctionnement de l'application avant sa publication définitive.

Enfin, après validation, le déploiement en production s'effectuera automatiquement, rendant la nouvelle version immédiatement accessible aux utilisateurs. Cette approche assurera une transition fluide et sécurisée entre les différentes versions de l'application.

Gestion des Environnements

Pour garantir une transition fluide entre le développement et la mise en production, nous utilisons plusieurs environnements :

- **Développement** : Permet aux développeurs de tester leurs fonctionnalités en local.
- **Staging** : Espace intermédiaire pour valider les mises à jour avant leur déploiement final.
- **Production** : Version en ligne accessible aux utilisateurs finaux.

Sécurité et Fiabilité du Déploiement

Pour garantir un déploiement sécurisé et sans interruption, nous mettrons en place un déploiement progressif, permettant de surveiller les performances des nouvelles versions avant leur généralisation. En cas d'anomalie, un rollback automatisé assurera un retour rapide à une version stable. Un monitoring continu sera également prévu pour détecter et corriger les incidents rapidement.

Une planification prévisionnelle structurera le projet en plusieurs phases : conception et développement, suivis de tests et validations, avant un déploiement progressif en production. Enfin, une maintenance continue optimisera l'application en fonction des retours utilisateurs et des besoins émergents, assurant ainsi un service fiable et performant.

Planification prévisionnelle

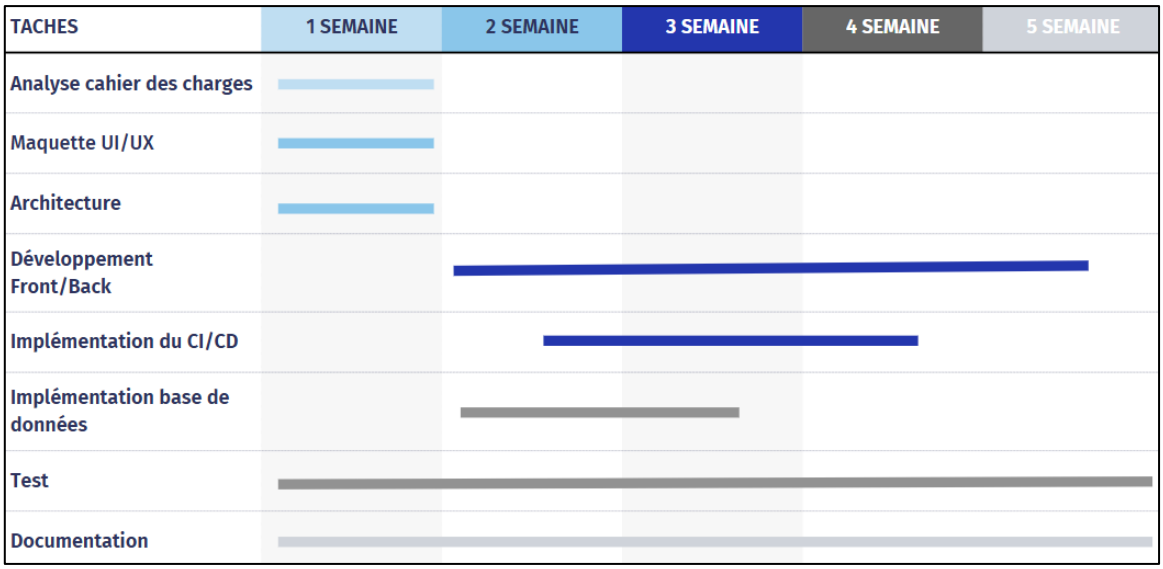
Méthodologie

Gestion du projet

Le projet sera mené selon une approche **Agile Scrum** avec :

- Des **sprints de 2,5 jours** pour une livraison rapide et itérative.
- Un **back log produit** géré dans **Notion** pour assurer un suivi structuré.
- Des **revues de sprint** avec le client et le Product Owner pour valider les avancées.
- Une **réunion quotidienne** pour synchroniser l'équipe et ajuster les priorités.
- Une **rétrospective à chaque fin de sprint** pour analyser les points d'amélioration.

Planning du projet



Phase d'analyse

- Définition détaillée du back log produit.
- Mise en place de l'infrastructure DevOps.
- Conception de l'interface utilisateur (UI/UX).

Phase de développement

- Création du squelette du frontend.
- Mise en place du backend et de l'authentification.
- Développement des modules clubs et événements.

Phase billetterie

- Implémentation du système de réservation.
- Gestion des places disponibles et verrouillage temporaire.
- Intégration du système de paiement.
- Génération des e-tickets et gestion des annulations.

Phase admin

- Développement du back-office pour les clubs et la gestion des événements.
- Suivi des participants et tableau de bord du BDE.
- Génération de rapports statistiques et export des données comptables.

Phase gamification

- Mise en place du système de points et des récompenses.
- Intégration des codes promo et avantages pour les adhérents.

Phase finalisation

- Réalisation des tests de charge.
- Corrections des bugs et optimisations des performances.

Pertinence des questions à débattre

Politique de remboursement et d'annulation ?

Le cahier ne stipule pas si une politique de remboursement ou d'annulation sera mise en place.

Proposition :

- **Remboursement total jusqu'à 7 jours avant l'évènement**
- **Plus de remboursement 24 heures avant l'évènement**

Quelle stratégie adopter pour promouvoir la plateforme auprès des étudiants et des clubs ?

Comment former les utilisateurs et les responsables de club à l'utilisation de la plateforme ?

Comment gérer les mises à jour et les évolutions futures de la plateforme ?

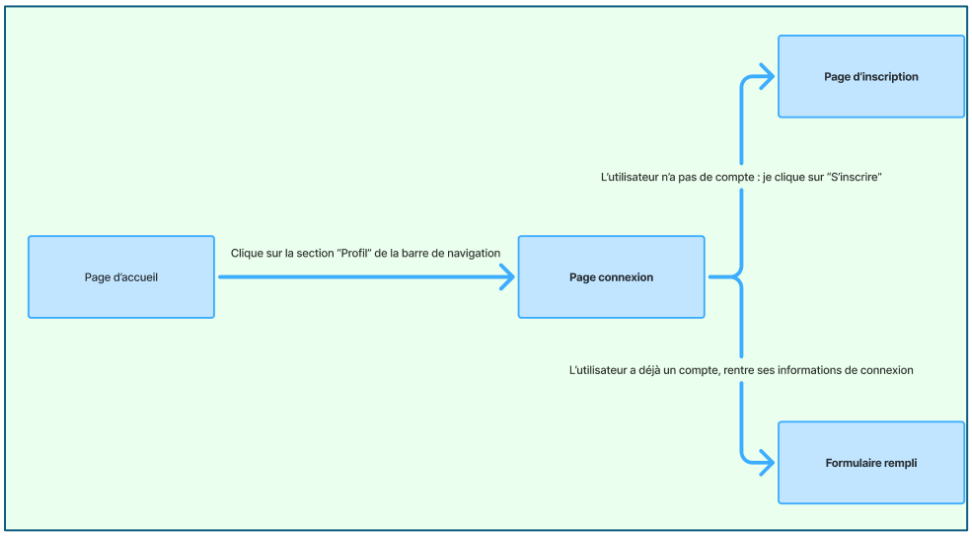
Maquette

Après avoir présenté les aspects fonctionnels et techniques du projet, nous abordons maintenant la dimension visuelle de notre plateforme. Cette section détaille les choix graphiques et l'expérience utilisateur conçus pour refléter l'identité du CESI tout en offrant une interface intuitive et engageante pour les utilisateurs.

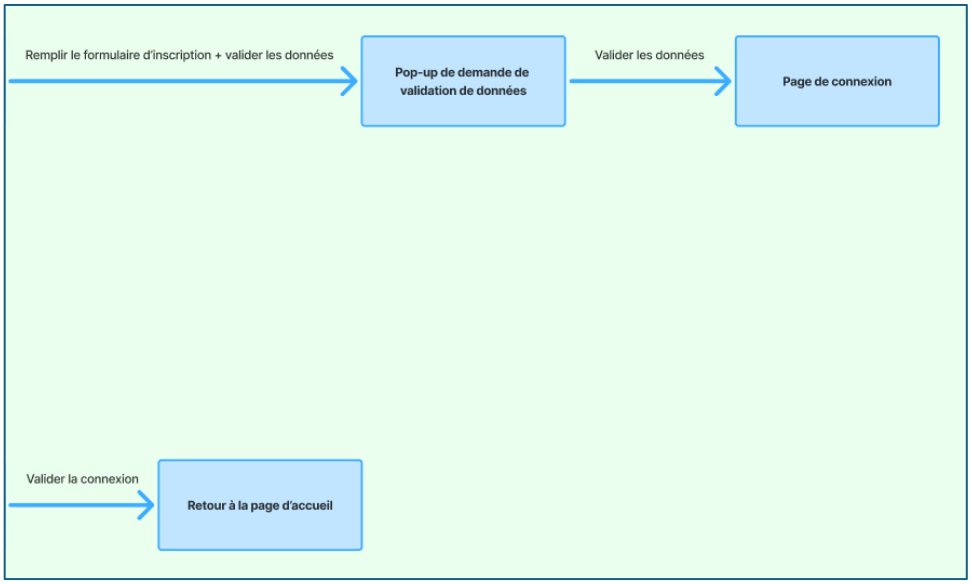
Analyse du besoin

Afin d'anticiper la conception de notre site web, nous avons élaboré des parcours utilisateurs. Il s'agit de diagrammes illustrant chacun des écrans que l'utilisateur traversera afin de réaliser une tâche bien précise. Ces parcours sont composés d'étapes (interfaces) et d'actions (processus entraînant le changement d'interface).

Parmi les parcours que nous avons établis, analysons de création d'utilisateur.



Parcours utilisateur – Création d'un utilisateur – Partie 1



Parcours utilisateur – Création d'un utilisateur – Partie 2

L'utilisateur commence son parcours sur la page d'accueil, où il sélectionne la section « Profil » de la barre de navigation. Cette action le redirige toujours vers la page de connexion.

S'il n'a aucun compte existant, il clique sur « S’inscrire », l’amenant à la Page d'inscription.

À cette étape, il remplit alors entièrement le formulaire d'inscription, puis valide toutes les données soumises. Une fenêtre pop-up de demande de validation de données apparaît toujours pour certifier toutes les infos entrées. Après que cette validation est entièrement bien effectuée, l'utilisateur est redirigé vers la Page de connexion.

S'il possède déjà bel et bien un compte, ou vient tout juste de finaliser son inscription, il saisit ses informations de connexion sur la Page de connexion et les valide. S'il y a exactitude dans chaque information, il accède vraiment à la Page d'accueil. Cela marque donc la fin du processus d'inscription ou bien de connexion.

Conception graphique

La conception graphique de la plateforme repose sur trois principes :

- **Cohérence avec l'identité CESI :** Le plateforme CESI-vents s'inscrira naturellement dans l'écosystème numérique de l'école en reprenant ses codes visuels tout en développant sa propre personnalité.
- **Clarté et accessibilité :** L'interface privilégie la lisibilité et la facilité d'utilisation pour tous les utilisateurs, y compris ceux disposant de contraintes visuelles.
- **Modernité et dynamisme :** le design reflète le caractère innovant de l'école et l'énergie de sa vie évènementielle à travers des choix graphiques modernes et contemporains.

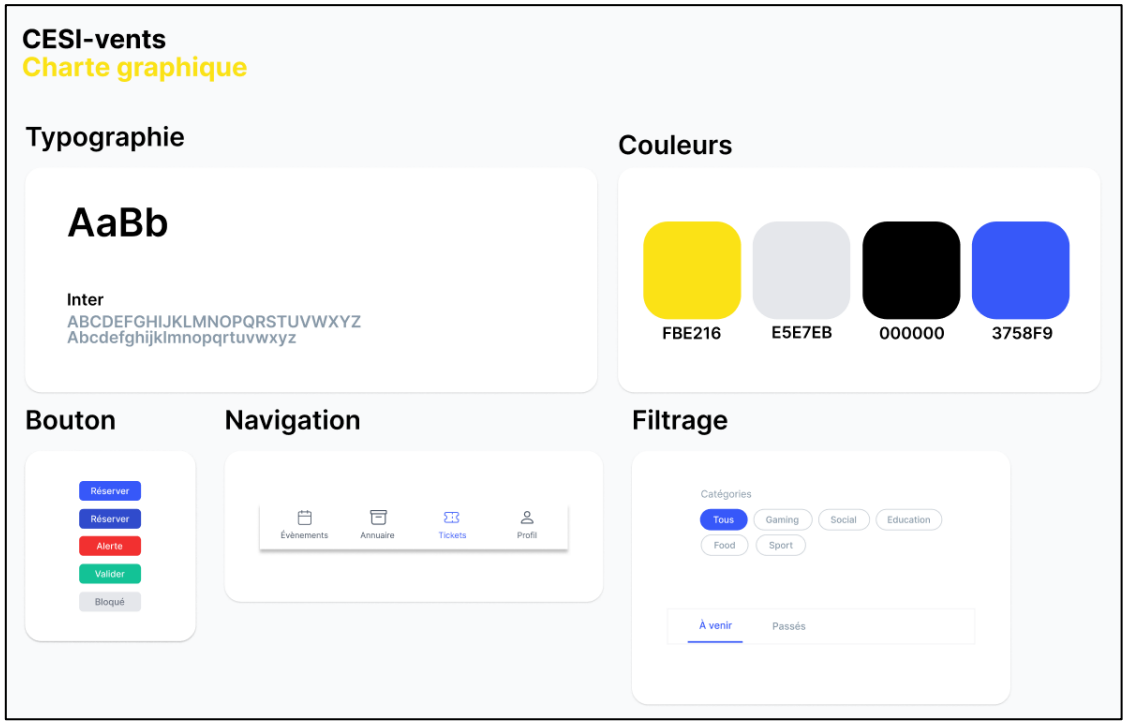
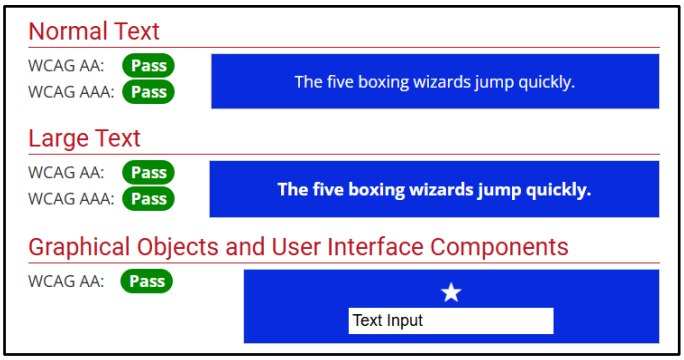


Illustration de la charte graphique de CESI-events

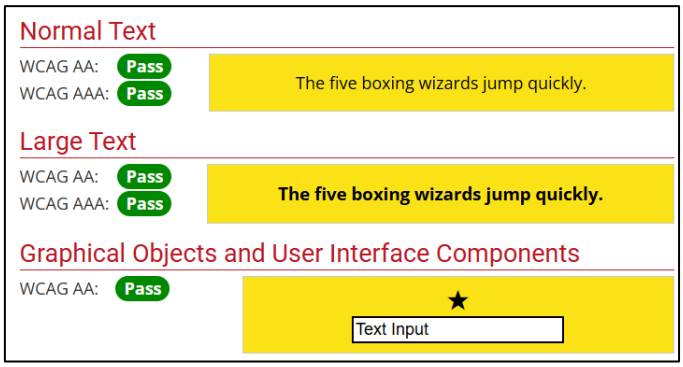
Couleurs

Les couleurs principales de la plateforme s'inspirent de la charte graphique du CESI tout en introduisant des teintes propres à CESI-vents pour démarquer l'application. De plus, un contraste a été rendu possible en utilisant l'outil « webaim », permettant d'aider les concepteurs de site en ligne à gérer efficacement les « jeux de couleurs ».

Ci-dessous le processus de test de contraste de couleurs :



Test de contraste de notre couleur bleu

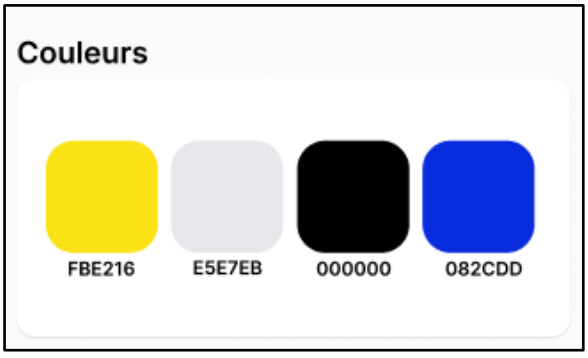


Test de contraste de notre couleur jaune

Sur les illustrations ci-dessus, nous remarquons que les tests de contrastes sont tous réussis, valorisant ainsi notre choix de couleurs.

Notre charte sera principalement composée de quatre couleurs.

Ci-dessous la palette de couleurs que nous utiliserons



Typographie

Le système typographique de CESI-vents utilise la famille de polices Inter, une police sans-serif moderne conçue pour offrir une excellence lisibilité sur les écrans :

Nous utilisons la police « Inter » avec différentes épaisseurs de texte pour tous les éléments de l'interface :

- Titres : Inter Bold
- Sous-titres : Inter Semi-Bold
- Corps de texte : Inter Regular
- Textes secondaires : Inter Light

Composants d'interface

Pour faciliter de développement et éviter de créer trop de style différent, nous allons définir un kit UI.

Les boutons

Notre système de boutons utilise un code couleur cohérent pour indiquer différentes actions :

Bouton principal (bleu) :

- Actions principales comme "Réserver" ou toute action positive qui fait avancer l'utilisateur dans son parcours.

Bouton d'alerte (rouge) :

- Actions destructives ou qui nécessitent une attention particulière comme "Annuler" ou "Supprimer".

Bouton de validation (vert) :

- Confirmation d'actions ou états positifs comme "Validé" ou "Disponible".

Bouton bloqué (gris) :

- Bouton démontrant que l'action ou la fonctionnalité n'est pas disponible ou accessible.

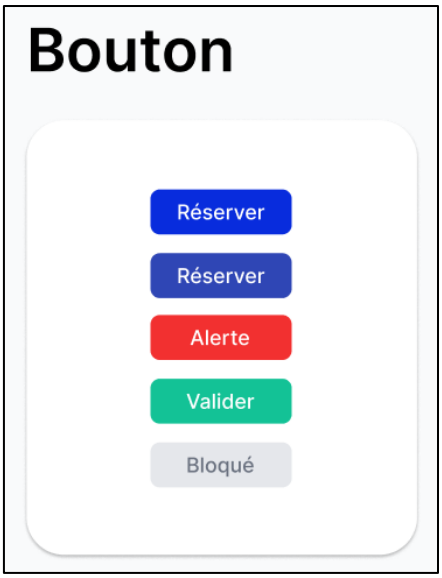


Illustration des différents boutons

Tous les boutons présentent des coins légèrement arrondis pour une apparence moderne et accueillante.

Les boutons cliquables se distingueront grâce à un changement de couleur au passage de la souris dessus.

Navigation

Le système de navigation utilise une approche minimaliste qui passera aussi bien sur mobile que sur ordinateur. Avec des icônes claires et un texte d'accompagnement.

L'élément actif sera mis en évidence par une couleur bleue qui contraste avec les icônes inactives en gris, permettent à l'utilisateur de se repérer facilement.

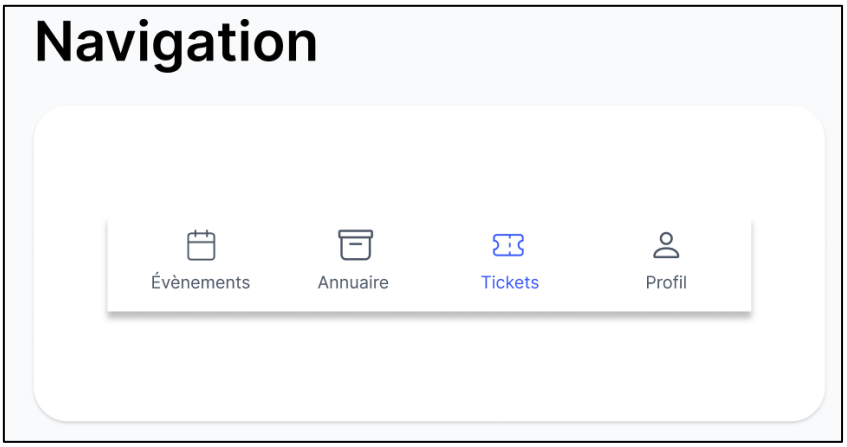


Illustration du menu de navigation

Système de filtre

Le système de filtrage se distingue par sa flexibilité et sa facilité d'utilisation :

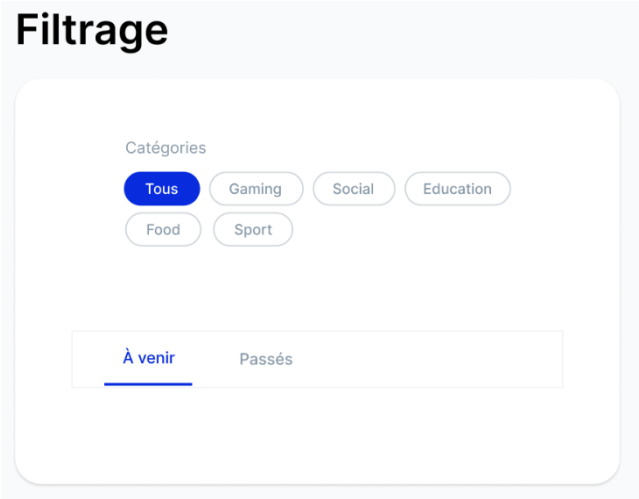


Illustration du système de filtrage

Ce système permet aux utilisateurs de trouver rapidement les évènements qui correspondent à leurs centres d'intérêt sans surcharger l'interface.

Premier visuel du site

Pour donner une première idée de l'interface utilisateur, on a ajouté un aperçu du site directement dans ce document. Cela aide les lecteurs à saisir rapidement le design et l'expérience utilisateurs projetés, sans avoir à se connecter à un outil extérieur immédiatement. Cette façon de faire rend la compréhension du projet plus accessible à tous les intervenants, en leur donnant une image claire et cohérente dès qu'ils plongent dans le livrable.

Afin de consulter l'ensemble des maquettes interactives et obtenir plus de détails sur la conception graphique, les visuels complets sont disponibles sur Figma à l'adresse suivante :

https://www.figma.com/design/dqjLgaUt1rsNkxvRbu9AUZ/CESI-vents?node-id=60-20&t=px4ZnXscSlxjmGZM-1

Page d'accueil

La page d'accueil constitue le point d'entrée principal de la plateforme CESI-vents. Elle est conçue pour offrir une vue d'ensemble claire des évènements à venir tout en facilitant l'exploration.

Pour l'instant, nous retrouvons sur la page d'accueil :

- Recherche d'évènement et de club
- Prochain ticket (possédé par l'utilisateur)
- Trois prochains évènements
- Petite liste des clubs
- Barre de navigation

Page évènement

La page évènement regroupe tous les évènements organisés par les clubs, avec un système de filtre, par catégorie et par club. Elle sera consultable par tout le monde.

Page Ticket

La page ticket regroupe tous les tickets de l'utilisateur. Une proposition que nous présentons consiste à faire en sorte que si aucun utilisateur n'a de ticket, alors une suggestion d'évènements est proposée. Ainsi, l'utilisateur est redirigé vers une section lui permettant de ne pas rester bloqué sur la page ticket.

Conclusion

Le projet CESI-vents représente une solution innovante et centralisée pour la gestion des événements étudiants au sein du CESI. Grâce à une architecture technique robuste, une intégration continue optimisée et une interface utilisateur ergonomique, cette plateforme répond aux besoins des clubs et du Bureau des Étudiants tout en favorisant l'engagement des étudiants.

Toutefois, des axes d'amélioration restent à explorer pour maximiser son impact. Il pourrait être pertinent d'enrichir le système de gamification avec des défis collaboratifs ou des classements dynamiques, d'intégrer des outils analytiques plus poussés pour optimiser les événements en fonction des tendances, ou encore d'élargir l'accessibilité de la plateforme aux partenaires extérieurs du CESI. Enfin, une stratégie de communication efficace et un accompagnement des utilisateurs assureront l'adoption et le succès à long terme de la plateforme.

Avec ces perspectives, CESI-vents s'inscrit dans une démarche évolutive et pérenne, prête à répondre aux besoins futurs de la communauté étudiante.

Annexe

PLAN D'INTEGRATION CONTINU – IBM

https://www.ibm.com/fr-fr/topics/continuous-integration

PLAN DE DEPLOIEMENT CONTINU – IBM

https://www.ibm.com/fr-fr/topics/continuous-deployment

DIAGRAMME DE CAS D'UTILISATION – IBM

https://www.ibm.com/docs/fr/dmrt/9.5?topic=diagrams-use-case

DIAGRAMME D'ACTIVITE – IBM

https://www.ibm.com/docs/fr/dmrt/9.5?topic=diagrams-activity

MAQUETTE UI/UX

https://www.figma.com/design/dqjLgaUt1rsNkxvRbu9AUZ/CESI-vents?node-id=60-20&t=px4ZnXscSlxjmGZM-1

BASE DE DONNEES

https://www.mongodb.com/docs/manual/core/databases-and-collections/

https://www.ionos.fr/digitalguide/serveur/know-how/mongodb-vs-firebase/

https://firebase.google.com/docs/database?hl=fr

ARCHITECTURE

https://fr.wikipedia.org/wiki/Microservices

https://webaim.org/resources/contrastchecker/