

Rapport PFE : Ingénierie des systèmes d'informations

Ghrai Marwa

2021

Dédicace

Je dédie ce travail à tous ceux qui me sont chers :

A ma mère et mon père qui m'ont arrosée de tendresse et des espoirs, leur amour ont fait de moi ce que je suis aujourd'hui.

A mes sœurs (Najla , Nadra, Rimel) et Sabrine Marzougui qui m'ont toujours soutenu et encouragé durant ces années d'études.

A toute ma famille et mes amis.

Remerciements

C'est avec un grand plaisir que je réserve ces lignes en signe de profonde reconnaissance et de gratitude à Monsieur **Bel-hassen Mazigh** pour l'encadrement et les recommandations précieuses, sa disponibilité et ses encouragements tout au long de cette période, je le remercie vivement pour m'avoir fait confiance.

Je remercie mon encadrant professionnel M. **Housseem Angoud** et toute l'équipe de **Itgate-Goup** pour m'avoir accueilli chaleureusement, me faire partager leur grande expériences, et être disponible malgré les responsabilités contribuant au bon déroulement de ce stage et à l'approfondissement de mes connaissances.

J'adresse également mes remerciements à tous les enseignants du Département Informatique pour leur contribution à ma formation.

Pour finir, je tiens à remercier mes parents, ma famille, mes amis et toute autres personnes ayant contribué ou non de loin ou de près à ce projet.

Table des matières

| | |
|--|----|
| Dédicace..... | 1 |
| Remerciements..... | 2 |
| Introduction générale | 7 |
| Introduction..... | 8 |
| 1. Contexte du stage..... | 8 |
| 1.1. Cadre du projet | 8 |
| 1.2. Présentation de l'établissement | 8 |
| 1.3. Les services de ItGate..... | 9 |
| 2. Présentation du sujet :..... | 9 |
| 2.1. Problématique et motivations :..... | 9 |
| 3. Expression des besoins | 10 |
| 3.1. Etude des applications existantes | 10 |
| 3.2. Solution proposée..... | 11 |
| Conclusion | 12 |
| Chapitre 2 : Spécification des besoins, Analyse et conception..... | 13 |
| Introduction..... | 13 |
| 1. Méthodologie de modélisation (UML)..... | 13 |
| 2. Spécification des besoins | 14 |
| 2.1. Identification des acteurs..... | 14 |
| 2.2. Besoins Fonctionnels..... | 15 |
| 2.3. Besoins non Fonctionnels..... | 16 |
| 3. Les diagrammes de cas d'utilisation..... | 16 |
| 3.1. Diagramme de cas d'utilisation global..... | 17 |
| 3.2. Diagramme de cas d'utilisation de l'acteur « Client » | 18 |

| | | |
|------|---|----|
| 3.3. | Diagramme de cas d'utilisation de l'acteur « Organisateur » | 20 |
| 3.4. | Diagramme de cas d'utilisation de l'acteur « Administrateur »..... | 23 |
| 4. | Les diagrammes de classes statiques | 27 |
| 4.1. | Dictionnaire des propriétés..... | 27 |
| 4.2. | Diagramme de classe statiques de notre application | 31 |
| 5. | Diagramme de séquence objet | 33 |
| 5.1. | Diagramme de séquence objet du cas d'utilisation « S'authentifier » | 33 |
| 5.2. | Diagramme de séquence objet du cas d'utilisation « S'inscrire »..... | 35 |
| 5.3. | Diagramme de séquence objet du cas d'utilisation « Créer un événement » | 36 |
| | Conclusion | 38 |
| | Chapitre 3 : Réalisation de l'application..... | 39 |
| | Introduction..... | 39 |
| 1. | Environnement de développement | 39 |
| 1.1. | Environnement matériel | 39 |
| 1.2. | Environnement logiciel | 40 |
| 2. | Architecture Système : | 42 |
| 2.1. | MVC-MVVM..... | 42 |
| 2.2. | Vue d'ensemble de l'application..... | 42 |
| 3. | Quelques Interfaces de l'application « Boleto » | 43 |
| 3.1. | Interface Principale..... | 44 |
| 3.2. | Interface Authentification..... | 44 |
| 3.3. | Interface Contact | 45 |
| 3.4. | Interface Liste des catégories | 45 |
| 3.5. | Interface Création d'événement | 46 |
| 3.6. | Interface Liste des avis | 47 |

3.7. Dashboard administrateur..... 48

Conclusion 48

Conclusion et perspectives..... 49

Introduction générale

La réservation et l'achat des tickets d'entrée pour un événement (une conférence, un festival, un match ou de l'achat d'un bon de participation à une formation. . .), ne sont pas toujours avantageux physiquement, ce qui nous motive à donner plus d'importance à la vente électronique.

Les billetteries numériques sont des outils rapides, faciles à utiliser et efficaces qui remplacent les billets au format papier. Ils sont utilisés par les compagnies aériennes ainsi que dans les transports en commun. Ils sont également employés pour l'achat ou la réservation d'une place de spectacle comme les tickets de cinéma. Bénéficiant de ces atouts, beaucoup d'organismes d'événements s'orientent encore plus vers l'utilisation de la billetterie électronique notamment avec les conditions sanitaires nationales et internationales favorisant la transformation digitale de tous les services.

Notre travail s'inscrit dans le cadre du projet de fin d'étude pour le diplôme du master professionnel en ingénierie des systèmes d'informations ayant comme objectif "La conception et la réalisation d'une application web de billetterie électronique" pour le compte de ItGate-Group, une société d'ingénierie informatique, spécialisée en développement des applications web, mobiles, la création des sites, la conception graphique et la gestion de communauté.

Ce rapport se compose de trois chapitres :

Nous débuterons le premier chapitre par la présentation générale de la société d'accueil, la problématique, la solution proposée ainsi que la démarche de travail.

Par la suite, nous allons introduire la partie de l'étude conceptuelle de l'application.

Enfin, nous décrirons l'architecture et la phase de la réalisation du projet.

Chapitre 1 : Etude préalable

Introduction

Dans ce chapitre, nous allons décrire et définir en premier lieu le cadre de ce projet et en deuxième lieu nous allons présenter l'organisme qui a accueilli ce stage de fin d'année. Dans une troisième étape, nous présenterons les motivations, la problématique, l'analyse de l'existant ainsi que la solution proposée qui a permis à ce stage d'aboutir à ses fins.

1. Contexte du stage

1.1. Cadre du projet

Le projet entre dans le cadre universitaire académique au sein de la faculté des sciences de Monastir(FSM) pour valider le projet de fin de 2ème année Master Professionnel spécialité Ingénierie des systèmes d'informations. Ce projet a été effectué au sein de la société ItGate-Group pour une durée de 6 mois.

1.2. Présentation de l'établissement

ItGate Group est une société créée en 2015 et spécialisée en développement et services informatiques : l'expérience utilisateur, la conception, le développement et le Webdesign, Elle déclare avoir les valeurs comme : organisation, collaboration, confiance et responsabilité.



Figure 1.1 :logo de ItGate group

1.3. Les services de ItGate

Dans le cadre de ses activités de son administration quotidienne, ItGate s'occupe notamment de :

- mettre en place des solutions informatiques fiable, innovante et moderne;
- développer des applications web modernes;
- développer des applications mobiles pour les plateformes Android et IOS;
- développer des supports et des interfaces graphiques pour ses clients;
- promouvoir et de mettre en œuvre les stratégies de référencement pour une meilleure visibilité sur Internet;

ItGate est divisée en deux organismes : centre de formation professionnelle, spécialisé dans plusieurs secteurs (informatique, marketing, management, etc.), et boîte de développement Web, Mobile et Embarqué où j'ai effectué mon projet de fin de Master Professionnel.

2. Présentation du sujet :

Le billet électronique (aussi appelé e-ticket) est un billet dématérialisé qui remplace un support d'information matériel par des informations numériques. [1]

Etant faciles à utiliser et efficaces en remplaçant les billets au format papier, certaines compagnies aériennes ainsi que dans les transports en commun, les événements comme les matchs dans les stades, les cinémas, les concerts utilisent les billets électroniques.

Pour effectuer la vente de ces billets à leurs clients, ces organisations vont utiliser certaines plateformes de ventes de billets électroniques en ligne. C'est dans ce contexte que s'inscrit le sujet de notre projet de fin d'année qui consiste à développer une application web de billetterie électronique, qui facilite et accélère le marketing en ligne des événements. D'une part, Les organisateurs présentent leurs événements, et d'autre part les clients visualisent et achètent les billets.

2.1. Problématique et motivations :

Vu les circonstances sanitaires à l'échelle national et international due à la pandémie de Coronavirus et avec toutes les mesures prises notamment de distanciation sociale, notre sujet trouve tout son intérêt.

Voyant le grand nombre des évènements organisés chaque jour, la promotion, la visibilité et le choix devient de plus en plus complexe; de plus, on se retrouve face à d'énormes difficultés pour le client aussi que pour l'organisateur.

En effet :

- La nécessité d'une recherche quotidienne pour pouvoir visualiser et choisir l'évènement;
- C'est indispensable de se déplacer, faire la queue et gaspiller du temps, tout en ayant le risque que les tickets soient épuisés;
- Risque de perdre les billets ou les oublier le jour de l'évènement;
- Les coûts d'impression des tickets en papier qui augmentent de temps en temps;
- La désignation d'agents de guichets de confiance et la recherche des points de ventes accessibles par un maximum de public cible;

Face à ces problèmes, la réalisation d'une application de billetterie électronique est une solution adaptée à la gestion de tous les types d'évènements. On trouvera les outils nécessaires pour promouvoir les événements, mettre en place une billetterie en ligne, suivre les ventes en un coup d'œil.

3. Expression des besoins

L'expression des besoins est le processus permettant de définir les besoins et les attentes dans notre projet. Pour ce faire, il est indispensable d'adopter une démarche basée sur l'étude de l'existant et une interprétation comparative qui permet de positionner notre projet par rapport aux applications similaires.

3.1. Etude des applications existantes

Le tableau ci-dessous illustre le résultat de l'étude comparative que nous avons menés afin de détecter et d'évaluer les applications dites similaires à la nôtre.

| Application web | Avantages | Inconvénients |
|-----------------|-----------|---------------|
|-----------------|-----------|---------------|

| | | |
|--|--|--|
| <p>Teskerti (Tunisienne)</p>  | <ul style="list-style-type: none"> -L'application est facile à utiliser. -L'application est liée à Google Agenda. -Possibilité d'appliquer un code promo. | <ul style="list-style-type: none"> -Seul l'administrateur peut créer l'événement. -La carte géographique n'est pas bien précise. |
| <p>Tikashop (Tunisienne)</p>  | <ul style="list-style-type: none"> -Une alerte pour informer des prochains événements. -Choisir une date pour se présenter aux guichets de vente spécifiés. | <ul style="list-style-type: none"> -Problème de chargement de Google Maps. |
| <p>Billetweb (Française)</p>  | <ul style="list-style-type: none"> -L'organisateur peut créer et gérer son propre événement. -Récupération des billets perdus. -Existence d'un service technique. | <ul style="list-style-type: none"> -Pas de possibilité de laisser un feedback pour les organisateurs. |

Tableau 1.1 : Tableau de comparaison entre les applications existantes

3.2. Solution proposée

Malgré qu'elles répondent à certaines attentes, l'étude du marché des applications existantes nous a permis de dégager plusieurs limites. Pour se positionner par rapport à ces derniers, notre solution propose de développer une application web baptisée "BILLETO" qui permet non seulement de faciliter les processus de vente et d'achat des billets électroniques mais qui représente également un outil aux organisateurs pour promouvoir leurs événements.

Conclusion

Dans ce premier chapitre nous avons mis le sujet dans son cadre général et nous avons effectué une étude analytique et comparative entre des exemples d'applications de billetterie en ligne en déduisant leurs avantages et leurs défaillances que nous allons mettre en considération dans notre projet. Dans le deuxième chapitre, nous allons aborder l'étude conceptuelle de notre application, mettant en œuvre les tâches qui nous ont été confié au sein de cette société durant la période de stage.

Chapitre 2 : Spécification des besoins, Analyse et conception

Introduction

La phase d'analyse et de conception est la première phase du processus de développement que nous avons adopté. En effet, elle permet de formaliser les étapes préliminaires et la démarche du développement d'un système afin de la rendre plus fidèle aux besoins attendus.

Tout au long de ce chapitre, nous détaillerons les différents diagrammes de la méthodologie UML utilisés dans la phase d'analyse. Dans la suite, nous allons définir les différents cas d'utilisations de notre système. Puis, nous aborderons la partie conceptuelle en présentant les diagrammes des classes statiques et quelques diagrammes de séquence. Afin de rendre la logique du métier plus compréhensible, tous les diagrammes seront décrits en détails avec une petite conclusion à la fin du chapitre.

1. Méthodologie de modélisation (UML)

UML est un langage de modélisation qui est devenu un standard de l'industrie de l'informatique. Il est conçu pour être utilisé pour l'analyse et la conception des systèmes orientés objets. L'intérêt d'UML est de disposer de vues de haut niveau d'abstraction pour favoriser et faciliter la communication entre utilisateurs, experts métiers et informaticiens.

UML offre plusieurs diagrammes qui sont subdivisées

- En vues **statique** représentant physiquement le système à modéliser (diagramme de cas d'utilisation, diagramme de classes, diagramme de composants, diagramme de déploiement).
- En vues **dynamiques** modélisant le fonctionnement du système (diagramme des séquences, diagramme d'état de transitions, diagramme de collaboration et diagramme d'activité)



Figure 02.1 : Langage de Modélisation Unifié

2. Spécification des besoins

La spécification des besoins s'intéresse à faire le recensement des besoins qui devraient être satisfaits dans notre application. Afin de comprendre le projet, nous commençons à mentionner les acteurs actifs dans la réalisation de l'application. Ensuite, nous dégagons les besoins fonctionnels de chaque acteur et nous présentons les besoins non fonctionnels de notre application.

2.1. Identification des acteurs

Dans notre application nous avons distingué les acteurs suivants :

- **Administrateur** : C'est l'acteur qui possède un accès privé de notre système avec tous les privilèges son rôle est d'administrer l'application.
- **Visiteur ou internaute** : C'est la personne qui visite le site pour recherche des événements sans inscription.
- **Client** : C'est l'acteur qui doit s'inscrire pour bénéficier des événements disponibles, Il peut effectuer plusieurs opérations qui seront détaillées dans ce qui suit.
- **Organisateur** : C'est l'acteur qui s'inscrit pour fournir des événements aux clients.



Figure 2.2 : Identification des acteurs de l'application

2.2. Besoins Fonctionnels

Pour chaque acteur nous avons identifié un ensemble des besoins :

❖ Administrateur :

- **Gestion du profile** : l'administrateur peut s'authentifier, modifier son compte et consulter sa messagerie.
- **Gestion des utilisateurs** : l'administrateur peut répondre aux demandes d'inscription des organisateurs, désactiver un organisateur déterminé, consulter liste des clients et des organisateurs.
- **Gestion des évènements** : l'administrateur peut accepter ou refuser les évènements ajouté par les organisateurs avant qu'ils sont affichés aux clients.il a la possibilité de consulter et de supprimer les évènements.
- **Evaluation** : l'administrateur peut consulter l'évaluation des événements.
- **Consulter Statistiques** : il peut consulter des statistiques qui lui sont propres

❖ Internaute :

- **Inscription** : l'internaute peut s'inscrire en tant que client ou organisateur.
- **Accès aux catégories** : l'internaute peut consulter la liste des catégories.
- **Accès aux événements** : l'internaute peut consulter la liste des événements.

❖ Client :

- **Gestion du profile** : le client peut s'authentifier, modifier ou désactiver son compte.
- **Réserver et acheter en ligne** : C'est une phase très importante, pour cela elle exige un haut niveau de sécurité. Pour terminer la procédure de paiement avec succès, le client doit remplir le formulaire et indiquer le numéro de sa carte et sa valeur de vérification dite CVV.
- **Recevoir le ticket** : La page finale représente le billet électronique accompagné des détails à savoir : (nom de l'évènement, la date, le temps...) ainsi que QR Code. Le client aura la permission de télécharger le billet.
- **Evaluation** : le client peut évaluer un événement.

❖ **Organisateur :**

- **Gestion du profile :** l'organisateur peut s'authentifier, modifier ou désactiver son compte et contacter l'administrateur.
- **Gestions des évènements :** L'organisateur crée son évènement dans l'application en mettant toutes les informations qui y sont relatives. Une validation est indispensable pour que l'évènement soit ajouté à la liste des évènements. il peut aussi consulter la liste, apporter des modifications, désactiver, activer et supprimer les évènements.
- **Evaluation :** un organisateur peut consulter l'évaluation des services.
- **Consulter statistiques :** comme pour l'administrateur, un organisateur pourra consulter quelques statistiques concernant l'évènement

2.3. Besoins non Fonctionnels

Notre application est soumise à un ensemble de contraintes techniques et ergonomiques à respecter pour la réalisation et le bon fonctionnement. Les principaux besoins non fonctionnels sont :

- **Fiabilité :** L'application doit toujours s'exécuter sans erreur et doit être satisfaisante ;
- **Gestion des erreurs :** L'ambiguïté doit être indiquée par des messages d'erreur bien claires pour guider correctement l'utilisateur et le familiariser avec notre site Web ;
- **Ergonomie :** En terme de navigation entre les différentes pages, les couleurs utilisées et les mises en page doivent être adaptée à l'expérience de l'utilisateur, sans effort (claire et simple à utiliser) ;
- **Sécurité :** Notre solution doit d'abord respecter la confidentialité des données personnelles de l'utilisateur, qui reste l'une des restrictions les plus importantes de l'application ;
- **Aptitude à la maintenance et à la réutilisation :** Le système doit être conforme aux normes et à une architecture bien structurée afin de pouvoir être entretenu et réutilisé ;

3. Les diagrammes de cas d'utilisation

Le diagramme de cas d'utilisation (en anglais « **use case** ») a pour objectif de décrire sous forme d'actions et de réactions le comportement d'un système du point de vue d'un utilisateur.

3.1. Diagramme de cas d'utilisation global

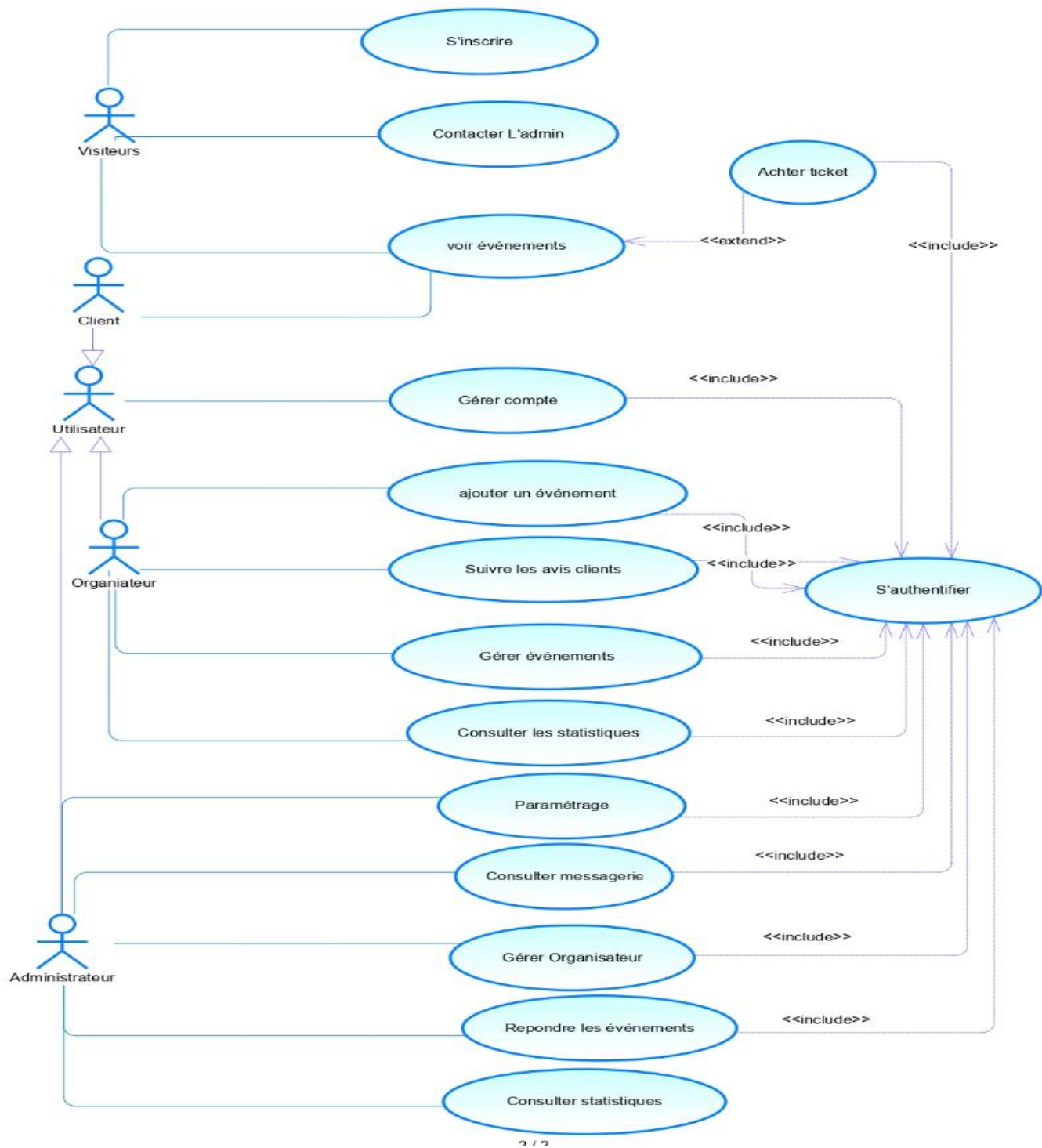


Figure 2.3 : Diagramme de cas d'utilisation global

Le diagramme représenté par la figure ci-dessus décrit le diagramme de cas d'utilisation global et les différentes actions réalisées par l'internaute, l'utilisateur, le client, l'organisateur et l'administrateur.

3.2. Diagramme de cas d'utilisation de l'acteur « Client »

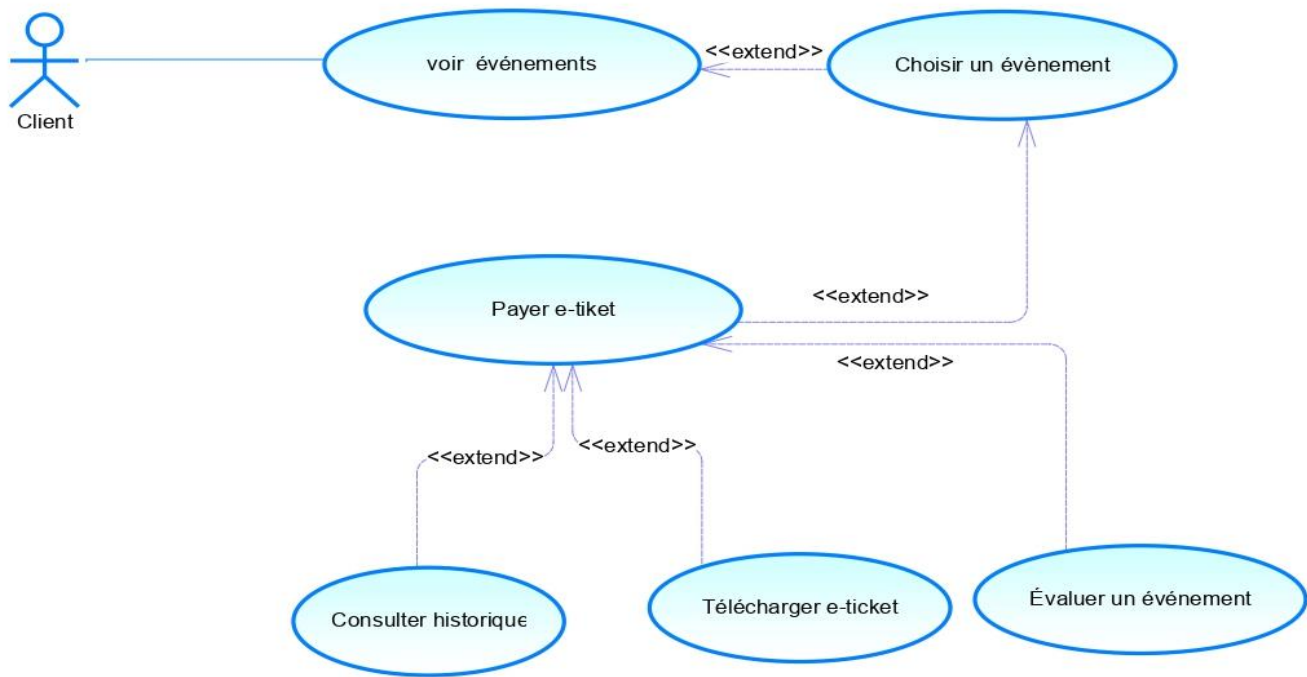


Figure 2.4 : Diagramme de cas d'utilisation de l'acteur « Client »

Le tableau suivant présente la description du cas d'utilisation « Consulter événement »

| | Voir événements |
|-------------------|-----------------|
| Cas d'utilisation | Voir événements |

| | |
|---------------------|---|
| Acteur | Client |
| Objectif | Consulter événement pour voir les nouveautés et réserver |
| Précondition | S'authentifier |
| Post-conditions | Déconnecter ou faire des autres tâches. |
| Scénario nominal | <ol style="list-style-type: none"> 1. Le client demande la liste des évènements 2. le système affiche la liste des évènements |
| Scénario alternatif | <ol style="list-style-type: none"> 2. le système affiche un message qui indique qu'il n'y a pas d'évènements dans la base de données ou erreurs. |

Tableau 2.1 : Description cas d'utilisation "Consulter événement"

Le tableau suivant présente la description du cas d'utilisation «Payer e-ticket»

| | Payer e-ticket |
|-------------------|--|
| Cas d'utilisation | Payer e-ticket |
| Acteur | Client |
| Objectif | Payer le ticket pour avoir le billet avec le QR Code |
| Précondition | S'authentifier |
| Post-conditions | Déconnecter ou faire des autres tâches. |

| | |
|---------------------|--|
| Scénario nominal | <ol style="list-style-type: none"> 1. Choisit un évènement 2. Le système affiche l'évènement sélectionné. 3. Le client fait une demande d'achat. 4. Le système affiche un formulaire de paiement. 5. Le client remplit les informations puis valide. 6. Le système affiche un message de succès après une vérification du formulaire |
| Scénario alternatif | <ol style="list-style-type: none"> 2. le système affiche un message qui indique si l'évènement n'existe pas dans la base de données ou erreurs. 6. système affiche un message qui indique que le formulaire n'est pas valide. |

Tableau 2.2 : Description cas d'utilisation "Payer e-ticket"

3.3. Diagramme de cas d'utilisation de l'acteur « Organisateur »

C'est un raffinement des cas d'utilisation d'organisateur présenté dans le diagramme de cas d'utilisation général.

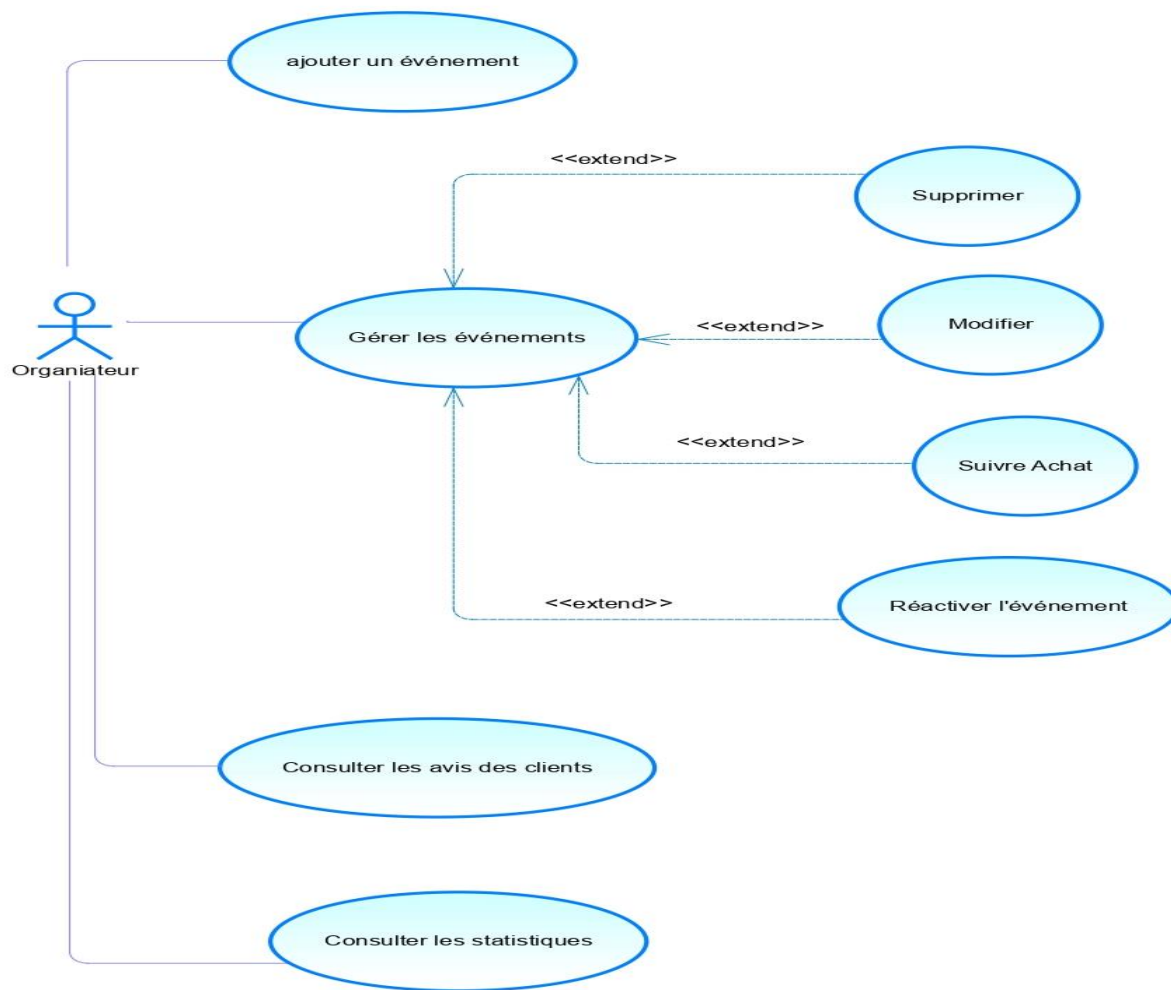


Figure 2.5 : Diagramme de cas d'utilisation détaillé de l'acteur organisateur

Le tableau suivant présente la description du cas d'utilisation «Ajouter un évènement»

| Ajouter un évènement | |
|----------------------|----------------------|
| Cas d'utilisation | Ajouter un évènement |
| Acteur | Organisateur |

| | |
|---------------------|--|
| Objectif | Ajouter un événement. |
| Précondition | S'authentifier |
| Post-conditions | Déconnecter ou faire des autres tâches. |
| Scénario nominal | <ol style="list-style-type: none"> 1. Le client demande la liste des sous catégories. 2. le système affiche la liste des sous catégories. 3. Le client choisit une sous-catégorie. 4. Le client fait une demande d'ajout. 5. Le système affiche un formulaire d'ajout. 6. Le client remplit les informations puis valide. 7. Le système affiche un message de succès après une vérification du formulaire |
| Scénario alternatif | 7. système affiche un message qui indique que le formulaire n'est pas valide. |

Tableau 2.3 : Description du cas d'utilisation «Ajouter un évènement»

Le tableau suivant présente la description du cas d'utilisation «Consulter les évènements»

| consulter les évènements | |
|--------------------------|---|
| Cas d'utilisation | consulter les évènements |
| Acteur | Organisateur |
| Objectif | Modifier un événement. |
| Précondition | S'authentifier |
| Post-conditions | Déconnecter ou faire des autres tâches. |

| | |
|---------------------|---|
| Scénario nominal | <ol style="list-style-type: none"> 1. Modifier un événement <ol style="list-style-type: none"> 1.1. L'organisateur choisit un événement. 1.2. L'organisateur fait une demande de modification. 1.3. Le système affiche les différents champs de la catégorie à modifier. 1.4. L'organisateur apporte les différentes modifications de l'évènement et valide. 1.5. Le système enregistre et affiche un message de succès, le tout après une étape de vérification des données. 2. Supprimer un événement <ol style="list-style-type: none"> 2.1. L'organisateur choisit un événement. 2.2. L'organisateur fait une demande de suppression. 2.3. Le système affiche un message de confirmation. 2.4. L'organisateur confirme. 2.5. Le système affiche un message de réussite après la suppression de l'évènement. |
| Scénario alternatif | <p>(2.1) le système affiche un message qui indique si l'évènement n'existe pas dans la base de données ou erreurs.</p> <p>(1.5, 32.5) système affiche un message qui indique que le formulaire n'est pas valide.</p> |

Tableau 2.4: Description du cas d'utilisation «Consulter les évènements»

3.4. Diagramme de cas d'utilisation de l'acteur « Administrateur »

C'est un raffinement des cas d'utilisation d'administrateur présenté dans le diagramme de cas d'utilisation général.

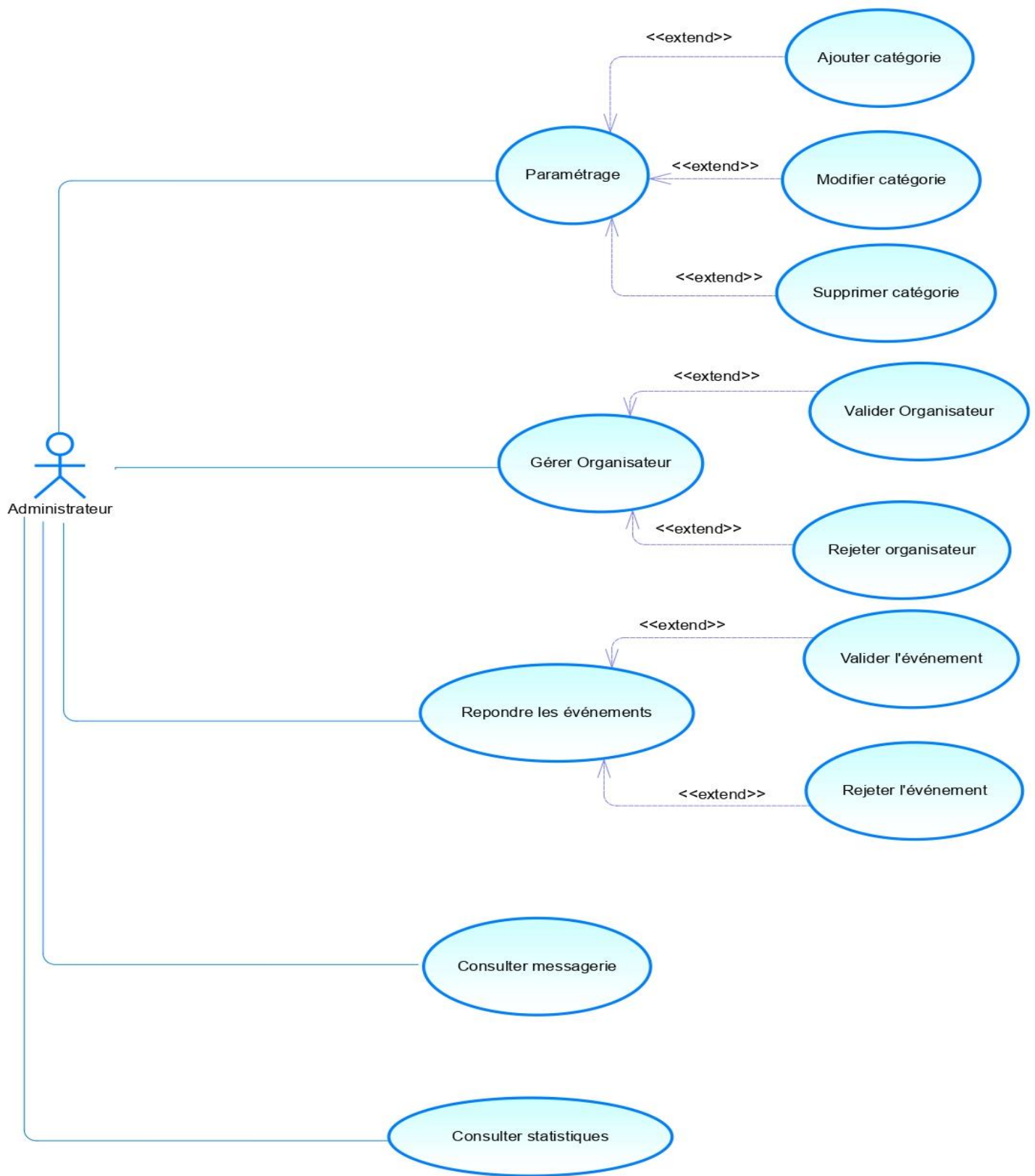


Figure 2.6 : Diagramme de cas d'utilisation détaillé de l'acteur administrateur

Le tableau suivant présente la description du cas d'utilisation «Gérer catégorie»

| Titre | Paramétrage |
|---------------------|--|
| Description | Ce diagramme illustre le cas d'utilisation Paramétrage |
| Acteurs | Administrateur |
| Préconditions | L'administrateur doit s'authentifier pour avoir le privilège de gérer les services et le sous services |
| Post-conditions | Déconnecter ou faire des autres tâches. |
| Scénario alternatif | <ol style="list-style-type: none"> 1. Ajouter catégorie <ol style="list-style-type: none"> 1.1. L'administrateur fait une demande d'ajout. 1.2. Le système renvoi le formulaire pour l'ajout. 1.3. L'administrateur rempli le formulaire et valide. 1.4. Le système affiche un message de succès après vérification des champs. 2. Modifier une catégorie <ol style="list-style-type: none"> 2.1. L'administrateur choisit une catégorie. 2.2. L'administrateur fait une demande de modification. 2.3. Le système affiche les différents champs de la catégorie à modifier. 2.4. L'administrateur apporte les différentes modifications de la catégorie et valide. 2.5. Le système enregistre et affiche un message de succès, le tout après une étape de vérification des données. 3. Supprimer une catégorie <ol style="list-style-type: none"> 3.1. L'administrateur choisit une catégorie. 3.2. L'administrateur fait une demande de suppression. 3.3. Le système affiche un message de confirmation. 3.4. L'administrateur confirme. 3.5. Le système affiche un message de réussite après la suppression de la catégorie. |

| | |
|---------------------|--|
| Scénario alternatif | (1.4 ,2.5, 3.5) système affiche un message qui indique que le formulaire n'est pas valide. |
|---------------------|--|

Tableau 2.5 : La description du cas d'utilisation «Gérer catégorie»

Le tableau suivant présente la description du cas d'utilisation « Consulter organisateur » :

| Titre Consulter Organisateur | |
|------------------------------|--|
| Description | Ce diagramme illustre le cas d'utilisation Consulter organisateur |
| Acteurs | Administrateur |
| Préconditions | L'administrateur est connecté à son compte. |
| Post-conditions | Déconnecter ou faire des autres tâches. |
| Scénario alternatif | <ol style="list-style-type: none"> 1. Valider l'organisateur <ol style="list-style-type: none"> 1.1. L'administrateur demande la liste des invitations. 1.2. Le système affiche la liste des invitations. 1.3. L'administrateur fait une demande d'acceptation. 1.4. Le système valide la demande d'acceptation. 2. Rejeter l'organisateur <ol style="list-style-type: none"> 2.1. L'administrateur demande la liste des notifications. 2.2. Le système affiche la liste des notifications. 2.3. L'administrateur fait une demande de refus de l'invitation. 2.4. Le système valide la demande de refus. |
| Scénario alternatif | (1.4, 2.4) le système affiche un message d'erreurs. |

Tableau 0.1: La description du cas d'utilisation « Consulter organisateur »

4. Les diagrammes de classes statiques

Les diagrammes de classes expriment de manière générale la structure statique d'un système en termes de classes et de relations entre ces classes. Une classe permet de décrire un ensemble d'objets (attributs et comportements), tandis qu'une relation permet de faire apparaître des liens entre ces objets.

4.1. Dictionnaire des propriétés

Le dictionnaire des propriétés de notre application « Billeto » comporte l'ensemble des informations, leurs propriétés et leurs désignations.

| Attribut | Type | Description |
|--------------------|-----------|---|
| Classe Compte | | |
| IdClient | Entier | Identifiant du compte |
| Email | Chaine | Email de l'utilisateur |
| MotDePasse | Chaine | Mot de passe de l'utilisateur |
| Statut | Booléen | Statut de l'utilisateur vrai si compte est actif faux sinon |
| Rôle | Caractère | Indique si l'utilisateur client, administrateur ou organisateur |
| Classe Utilisateur | | |
| IdUtilisateur | Entier | Identifiant d'utilisateur |

| | | |
|----------------------------|--------|-----------------------------------|
| Nom | Chaine | Nom d'utilisateur |
| Prénom | Chaine | Prénom d'utilisateur |
| Téléphone | Chaine | Numéro de téléphone d'utilisateur |
| Photo | Objet | Photo d'utilisateur |
| Classe Client | | |
| Adresse | Chaine | L'adresse de client |
| Ville | Chaine | Ville du client |
| CodePostal | Entier | Code Postal du client |
| Classe Organisateur | | |
| Adresse | Chaine | L'adresse du l'organisateur |
| CodePostal | Entier | Code Postal du l'organisateur |
| Ville | Chaine | Ville du l'organisateur |
| Description | Chaine | description du l'organisateur |
| CV | objet | CV du l'organisateur |
| Classe Société | | |
| IdSociété | Entier | Identifiant de la société |
| Nom | chaine | Nom de la société |

| | | |
|-------------------------|---------|--|
| Email | Chaine | Email de la société |
| Téléphone | Chaine | Numéro de téléphone de la société |
| Adresse | Chaine | Adresse de la société |
| Classe événement | | |
| Idévénement | Entier | Identifiant de l'événement |
| Titre | Chaine | Titre de l'événement |
| Date_Début | Date | Date de début de l'événement |
| Heure_Début | Chaine | Heure de début de l'événement |
| Heure_Fin | Chaine | Heure de fin de l'événement |
| Image | Objet | Image de l'événement |
| Vidéo | Objet | Vidéo de l'événement |
| Taux_remplissage | Entier | Taux de remplissage de l'emplacement de l'événement |
| Ville | Chaine | Ville de l'événement |
| Emplacement | Chaine | Emplacement de l'événement |
| Description | Chaine | Description de l'événement |
| Statut | Booléen | Statut de l'événement 1 si l'événement actif 0 sinon |

| | | |
|------------------------------|---------|--|
| Validate | Booléen | 1 si l'événement validé 0 sinon |
| Classe classe | | |
| IdClasse | Entier | Identifiant de la classe d'événement |
| Nom | Chaine | Nom de la classe d'événement |
| Prix | Réel | Prix de la classe d'événement |
| Classe Commanditaires | | |
| IdCommanditaires | Entier | Identifiant du commanditaire d'événement |
| Nom | Chaine | Nom du commanditaire d'événement |
| Type | Chaine | Type du commanditaire d'événement |
| Logo | Objet | Logo du commanditaire d'événement |
| Classe Membre_Equipe | | |
| IdEquipe | Entier | Identifiant du Membre d'équipe d'événement |
| Nom | Chaine | Nom du Membre d'équipe d'événement |
| Prénom | Chaine | Prénom du Membre d'équipe d'événement |
| Photo | Objet | Photo du Membre d'équipe d'événement |
| Classe Billet | | |

| | | |
|------------------------------|--------|-------------------------------|
| IdBillet | Entier | Identifiant de billet |
| QrCode | Chaine | Le QR Code de billet |
| Classe Paiement | | |
| IdPaiement | Entier | Identifiant de paiement |
| Classe Catégorie | | |
| IdCatégorie | Entier | Identifiant du Catégorie |
| Nom | string | nom du Catégorie |
| Photo | String | Photo du Catégorie |
| Classe Sous-catégorie | | |
| IdSousCatégorie | Entier | Identifiant du Sous-catégorie |
| Nom | string | nom du Sous-catégorie |
| Photo | String | Photo du Sous_catégorie |

Tableau 2.7 : Dictionnaires des propriétés

4.2. Diagramme de classe statiques de notre application

En se basant sur ce qui précède, une solution conforme aux besoins exprimés et aux objectifs déjà se résume dans un diagramme composé des classes suivantes :

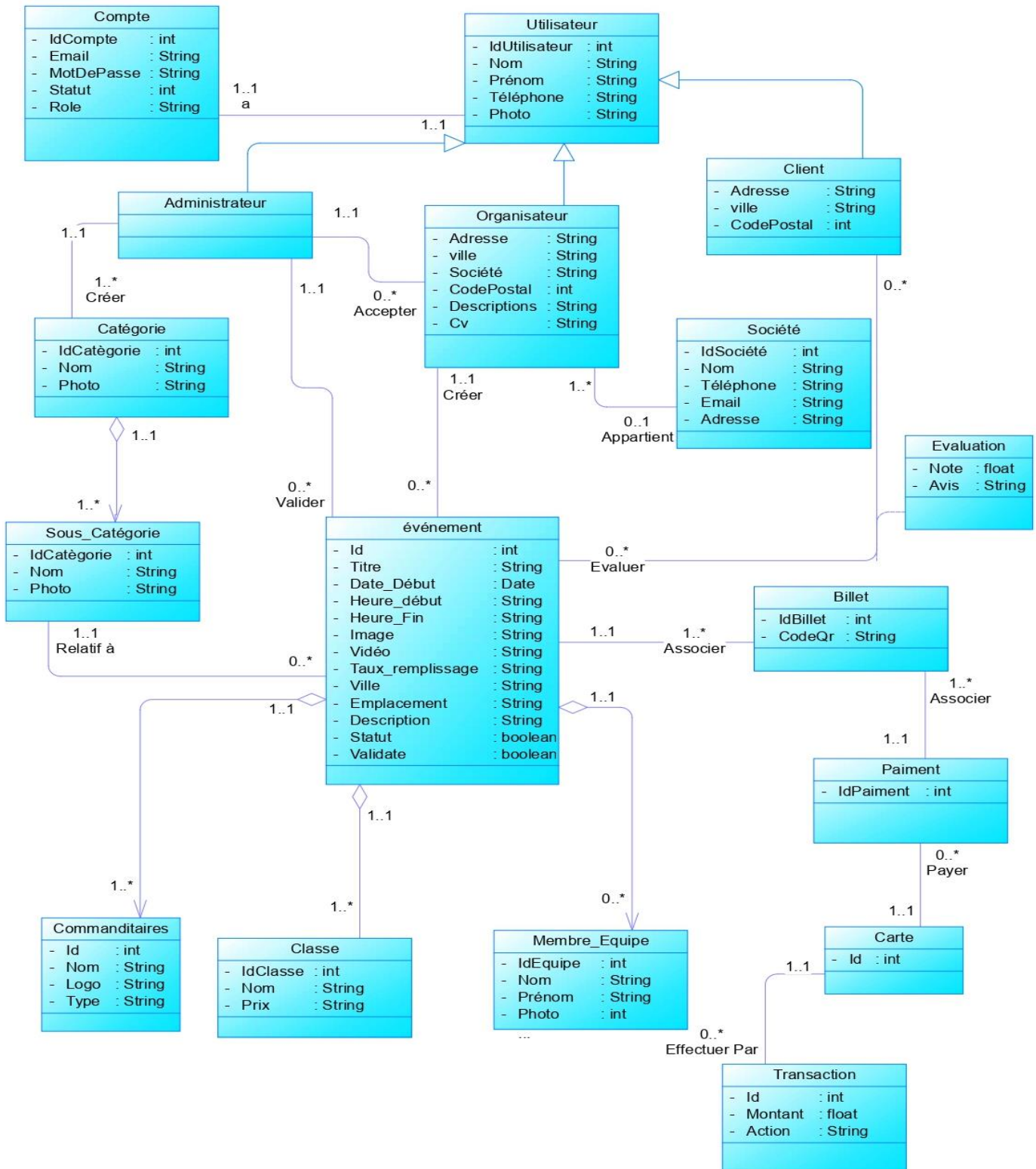


Figure 2.7 : Diagramme de classe statique

5. Diagramme de séquence objet

Le diagramme de séquence permet de décrire les scénarios de chaque cas d'utilisation en interaction avec les objets tout en mettant l'accent sur la chronologie des opérations.

5.1. Diagramme de séquence objet du cas d'utilisation « S'authentifier »

La figure ci-dessus, est le diagramme de séquence objet du cas d'utilisation «Authentification »

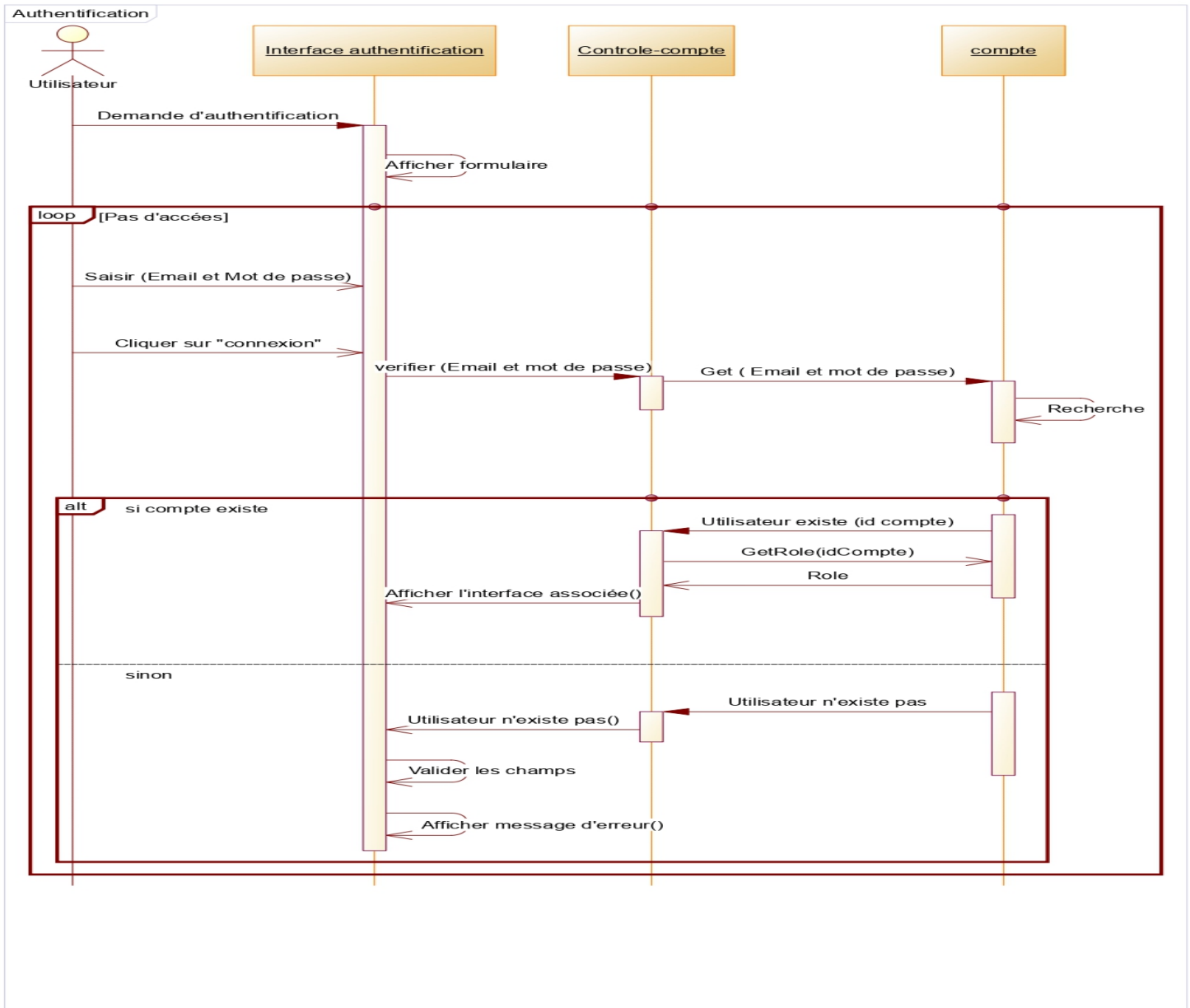


Figure 2.8 : Diagramme de séquence objet du cas d'utilisation «Authentification »

Chaque utilisateur doit s'authentifier, en fournissant son identifiant et son mot de passe afin d'accéder à son espace. Si les données saisies sont erronées, le système affiche un message d'erreur. Lorsque l'utilisateur demande de s'authentifier, un formulaire d'authentification sera affiché. Une fois la saisie terminée, l'utilisateur demande de se connecter, le contrôleur va vérifier si l'utilisateur existe par la méthode get (identifiant, mot de passe) S'il existe le contrôleur va vérifier son rôle et afficher le compte associé à cette personne. Sinon un message d'erreur sera affiché.

5.2. Diagramme de séquence objet du cas d'utilisation « S'inscrire »

Cette figure illustre l'inscription d'un Organisateur

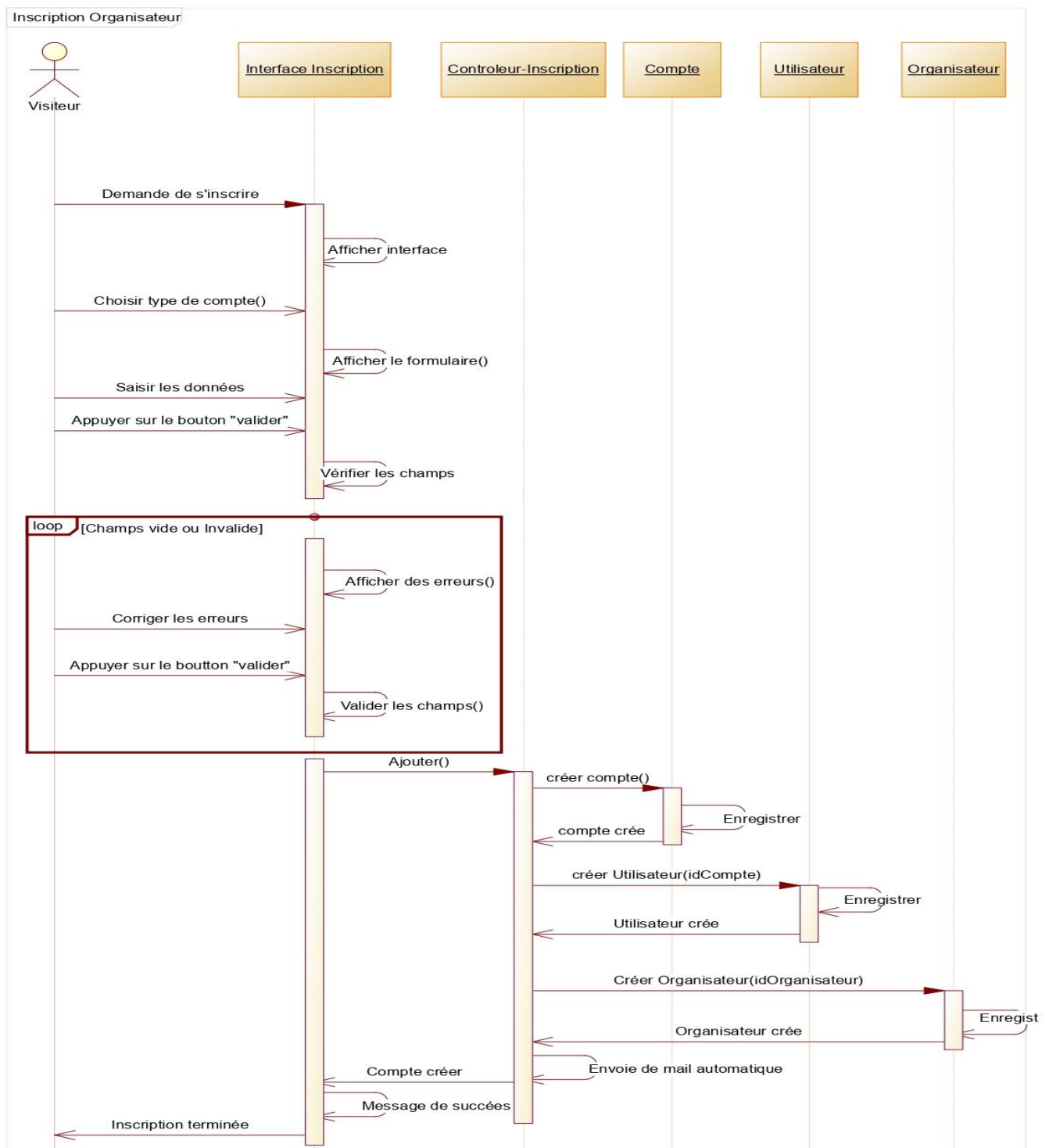


Figure 2.9 : Diagramme de séquence objet du cas d'utilisation «Inscription Organisateur »

❖ Description :

Lorsque l'internaute demande de s'inscrire, un message sera affiché pour lui demander de choisir le type du compte. Une fois que l'internaute confirme son choix (dans ce cas un organisateur d'événement) un formulaire d'inscription associé, sera affiché.

Après avoir rempli le formulaire l'internaute valide l'inscription. Les champs seront par la suite vérifiés :

- Si un ou plusieurs champs sont invalides ou vides, des messages d'erreurs seront affichés au-dessous de ces champs. L'internaute pourra ainsi les corriger et valider l'inscription de nouveau.
- Sinon si les champs sont valides, l'interface va demander au contrôleur d'ajouter le prestataire dans la base de données. Le contrôleur va créer des nouveaux objets (compte, utilisateur et organisateur), ensuite un mail sera automatiquement envoyé à l'organisateur, et un message de succès de l'opération sera affiché.

L'objet (compte) va contenir un attribut "statut" dont l'état est "0" en attente de la réponse de l'administrateur. Si l'administrateur accepte le prestataire, son statut devient "1", sinon le compte sera supprimé.

- Pour l'inscription d'un client, on procède de la même manière sauf que le statut du client est automatiquement en état "1". Il n'a pas besoin de la confirmation de l'administrateur.

5.3. Diagramme de séquence objet du cas d'utilisation « Créer un événement »

L'organisateur demande l'accès à la liste des catégories, l'interface envoie la demande, le contrôleur récupère la liste et l'affiche. L'organisateur choisit une catégorie, appuie sur 'détails', le contrôleur récupère les détails de catégorie. Après l'affichage des détails, il appuie 'créer un événement', un formulaire s'affiche puis saisit les données et clique sur 'Poster', les champs seront vérifiés.

S'il y a au moins un qui est incorrect un message d'erreur s'affiche sinon l'interface envoie la demande de dépôt d'événement au contrôleur, ce dernier va créer un nouvel objet événement. Ensuite le contrôleur récupère automatiquement la liste des événements personnels de l'organisateur et l'envoie à l'interface pour l'afficher

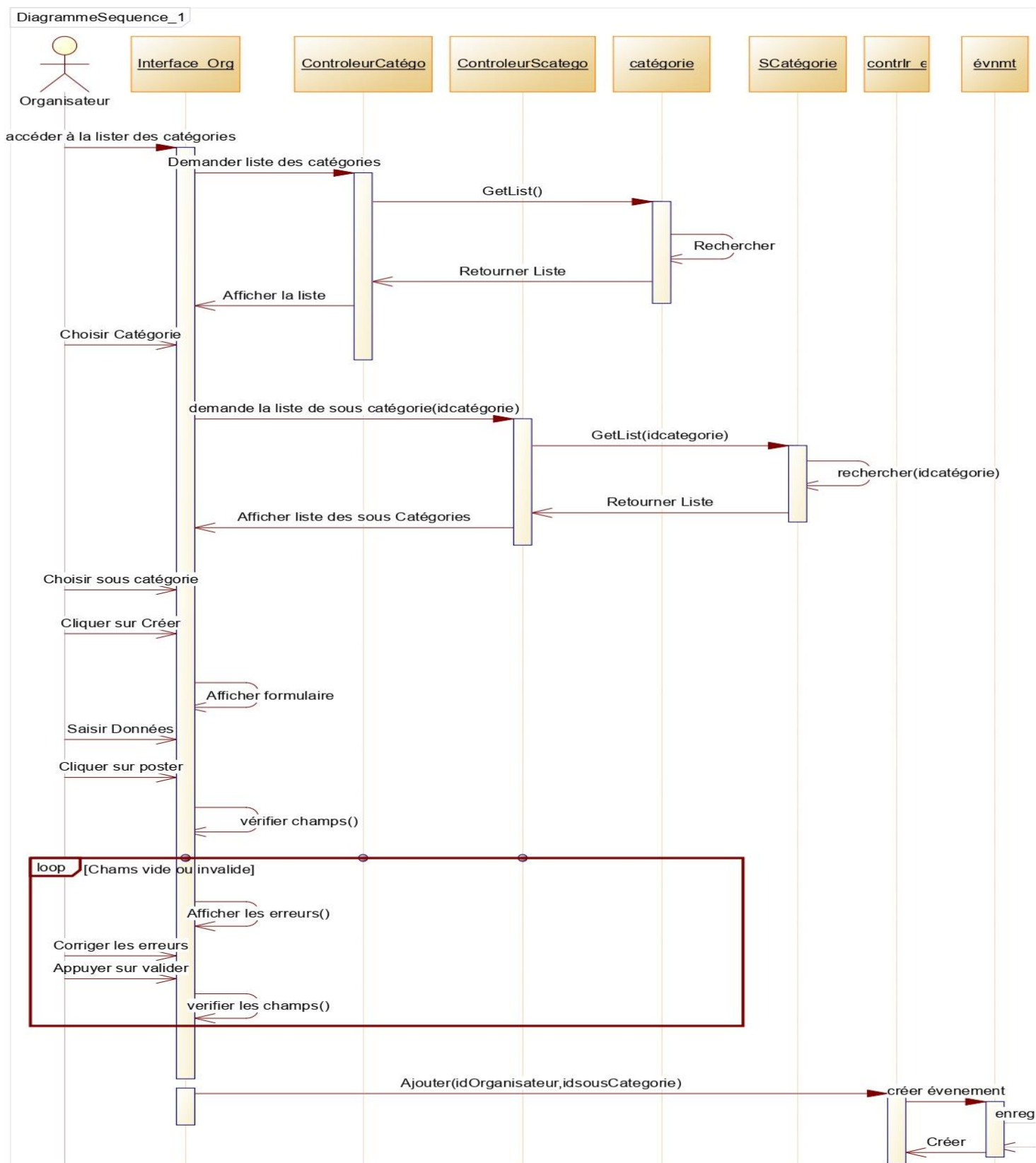


Figure 2.10 : Diagramme de séquence objet du cas d'utilisation « Créer un événement »

Conclusion

Dans ce chapitre, nous avons commencé par la spécification des besoins fonctionnels et non fonctionnels de notre application. Puis, nous avons présenté les diagrammes de cas d'utilisations de notre application « Boleto ».

Ensuite, nous avons entamé la phase de conception qui sert à identifier les différents objets contribuant à assurer les fonctionnalités souhaitées et les types des relations qui existent entre eux en donnant une description complète et détaillée de ces objets dans les diagrammes de classes statiques. Enfin, nous avons détaillé quelques diagrammes de séquence de certains cas d'utilisations.

Dans le chapitre suivant nous allons présenter la réalisation de notre application.

Chapitre 3 : Réalisation de l'application

Introduction

Après avoir élaboré la conception de notre application, nous abordons dans ce chapitre le dernier volet de ce rapport, qui a pour objectif d'exposer la phase de réalisation.

La phase de réalisation est considérée comme étant la concrétisation finale de toute la méthode de conception.

Nous menons tout d'abord, une étude technique où nous décrivons les ressources logicielles utilisées dans le développement de notre projet. Nous présentons en premier lieu notre choix de l'environnement de travail, où nous spécifions l'environnement matériel et logiciel qu'on a utilisé pour réaliser notre application puis nous détaillons l'architecture. En deuxième lieu, nous présentons quelques interfaces réalisées pour illustrer le fonctionnement de l'application.

1. Environnement de développement

Pour la réalisation de notre application, nous avons eu recours à plusieurs moyens matériels et logiciels

1.1. Environnement matériel

Le développement de l'application est réalisé via un ordinateur portable ayant les caractéristiques suivantes :

| Caractéristique | ASUS |
|-----------------|---|
| Marque | Asus |
| Processeur | Intel® Core™ i5-4210U CPU @ 1.70 GHz 2.40 GHz |
| RAM | 8GO |
| Disque dur | 500GO |

| | |
|-------------------------------|---------------------|
| Système d'exploitation | Microsoft Windows 8 |
|-------------------------------|---------------------|

Tableau 3.1 : Matériel de base

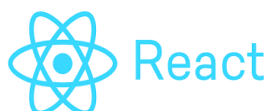
1.2. Environnement logiciel



MongoDB est un système de gestion de base de données orienté documents, répartitionnable sur un nombre quelconque d'ordinateurs et ne nécessitant pas de schéma prédéfini des données. Il est écrit en C++. Le serveur et les outils sont distribués sous licence SSPL, les pilotes sous licence Apache et la documentation sous licence Creative Commons4. Il fait partie de la mouvance NoSQL.



ExpressJs est un framework Node.js qui permet la mise en place rapide d'un serveur de requêtes http. Il récupère ainsi les requêtes Ajax du client et les ajoute à la pile de requêtes qui doivent être traitées par Node.js.



ReactJS est une bibliothèque JavaScript utilisée pour créer des composants d'interface utilisateur réutilisables



Node.js est un environnement bas niveau permettant l'exécution de JavaScript côté serveur, il utilise la machine virtuelle V8, la librairie libuv pour sa boucle d'événements, et implémente sous licence MIT les spécifications CommonJS.

Parmi les modules natifs de Node.js, on retrouve http qui permet le développement de serveur HTTP. Il est donc possible de se passer de serveurs web tels que Nginx ou Apache lors du déploiement de sites et d'applications web développés avec Node.js.

J W T

JSON Web Token (JWT) est un standard ouvert. Il permet l'échange sécurisé de jetons (tokens) entre plusieurs parties. Cette sécurité de l'échange se traduit par la vérification de l'intégrité des données à l'aide d'une signature numérique.



Visual Studio Code

Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS. Les fonctionnalités incluent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la refactorisation du code et Git intégré.



Postman

Postman est un logiciel qui permet de construire et d'exécuter des requêtes HTTP, de les stocker dans un historique afin de pouvoir les rejouer, mais surtout de les organiser en Collections. Cette classification permet notamment de regrouper des requêtes de façon « fonctionnelle » (par exemple enchaînement de création d'un événement).

Postman assure également la gestion des environnements, qui permet de contextualiser des variables et d'exécuter des requêtes ou des séries de requêtes dans différentes configurations.



mongodb compass

MongoDB Compass est un client graphique pour MongoDB. Compass se connecte au cluster MongoDB et présente deux onglets - Databases et Performance

2. Architecture Système :

Dans notre travail, toutes les données sont centralisées sur un seul serveur qui joue le rôle de l'administrateur or les utilisateurs utilisant simplement un client léger. Pour cela, il est bien clair que notre projet est basé sur l'architecture client-serveur.

Un **client** : est d'abord actif, il envoie des requêtes au serveur, il attend et reçoit les réponses du serveur.

Un **serveur** : est initialement passif, il attend, il est à l'écoute, prêt à répondre aux requêtes envoyées par des clients. Dès qu'une requête lui parvient, il la traite et envoie une réponse.

2.1. MVC-MVVM

MVC et MVVM sont des modèles de design pattern de structure.

MVC signifie Model-View-Controller et MVVM signifie Model-View-View-Model.

- Model : code qui s'occupe de la façon dont les données sont stockées;
- View : code qui s'occupe de la façon dont les données sont affichées;
- Controller : code qui s'occupe de la façon dont les données sont créées / mises à jour / supprimées;
- ViewModel : code qui s'occupe à la fois de la façon dont les données sont stockées et de la façon dont elles sont affichées;

Ces modèles aident à développer des applications qui sont librement couplées, faciles à tester et à entretenir.

Notre travail regroupe ces deux modèles, en effet : on a adopté MVC au niveau du backend (Node js) et le MVVM dans le frontend (React js).

2.2. Vue d'ensemble de l'application

Les composants de notre système se réagissent entre eux pour répondre à nos besoins. La figure suivante (figure 2.7) donne une vue d'ensemble de notre application et illustre les différents échanges entre les modules.

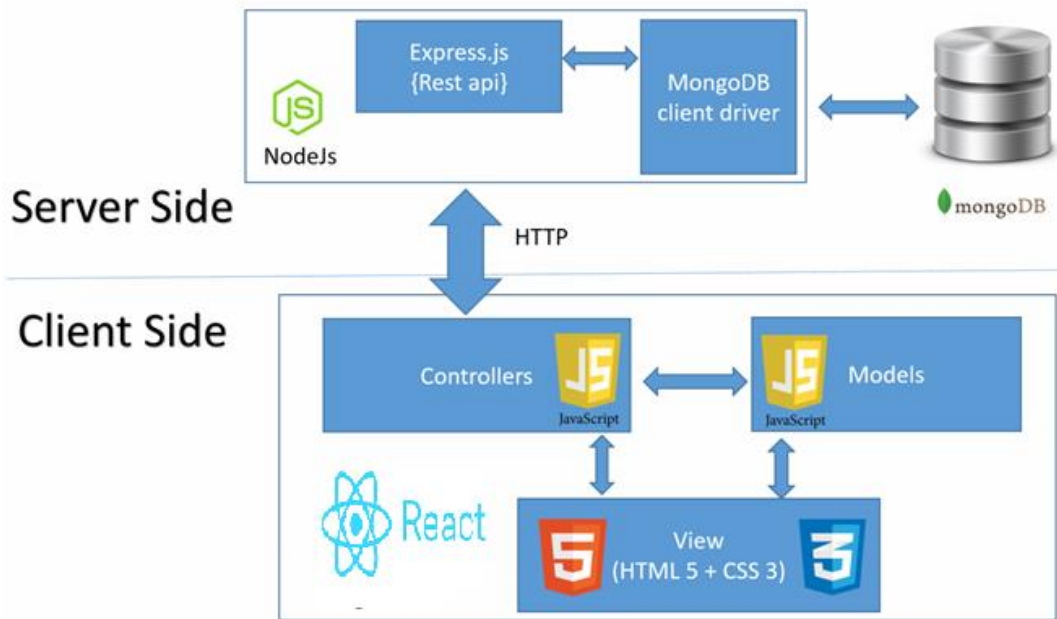


Figure 3.1 : Vue d'ensemble de l'application

Comme le montre l'image ci-dessus, nous avons ReactJs comme langage côté client qui traite la demande d'un client.

- Chaque fois qu'un utilisateur fait une demande (authentification, création d'événement, achat d'un billet,...), elle est d'abord traitée par ReactJs.
- Ensuite, la requête entre dans la deuxième étape, où nous avons Node.js comme langage côté serveur et ExpressJS comme framework web backend.
- Node.js gère les demandes client/serveur et ExpressJS fait une demande à la base de données.
- Dans la dernière étape, MongoDB (base de données) récupère les données et envoie la réponse à ExpressJS.
- ExpressJS renvoie la réponse à Nodejs et à son tour à ReactJs, puis affiche la réponse à l'utilisateur.

3. Quelques Interfaces de l'application « Boleto »

Après avoir explicité l'environnement de développement, nous exposons quelques interfaces réalisées de notre projet. Ces derniers sont les résultats de la consommation des apis dans la partie front end.

3.1. Interface Principale

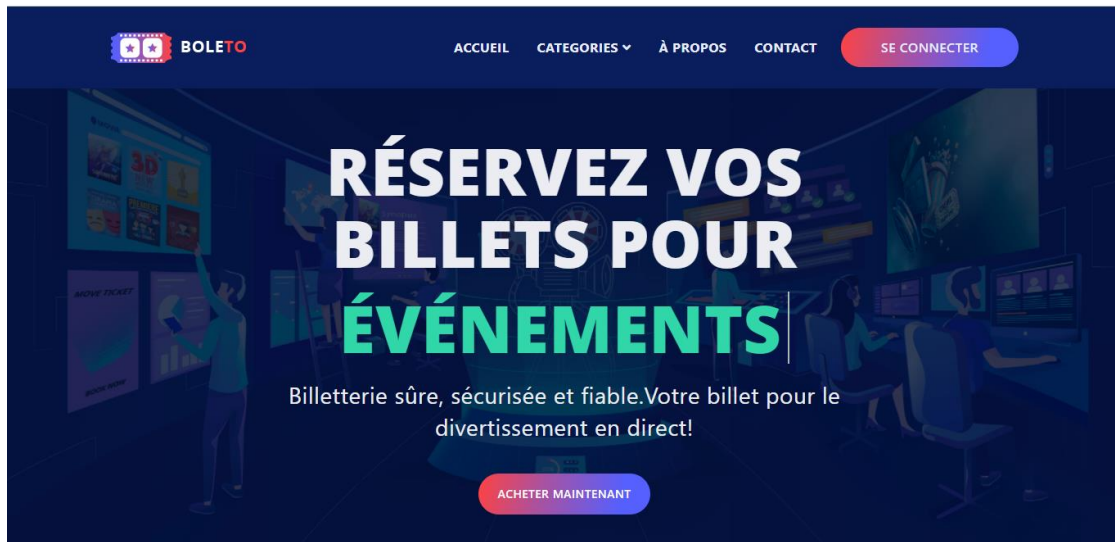


Figure 3.2 : Interface Principale

Cette interface représente l'interface principale, l'utilisateur peut visualiser l'application sans authentification.

3.2. Interface Authentification



Figure 3.3 : Interface Authentification

Pour que l'utilisateur puisse bénéficier des fonctionnalités de l'application, il doit s'authentifier.

3.3. Interface Contact

CONTACTER NOUS

ENTRER EN CONTACT

Nous aimerions parler de la façon dont nous pouvons travailler ensemble. Envoyez-nous un message ci-dessous et nous vous répondrons dans les plus brefs délais.

Nom Complet *

Enter your name

E-mail *

Enter your Email


Sujet *

Enter your subject

Message *

Enter your message

ENVOYER LE MESSAGE



TÉLÉPHONE

+1254 56789

E-MAIL

info@soleto.com

Figure 3.5 : Interface Contact










Dans cette interface il y a tous les contacts de l'administrateur, l'internaute peut aussi envoyer un message anonyme.

3.4. Interface Liste des catégories

LISTE DES CATEGORIES

+

Recherche:

| ID | PHOTO | TITRE | ACTION |
|----|---|------------|--|
| 0 |  | SPORT |   |
| 1 |  | ÉVÉNEMENTS |   |
| 2 |  | FILMS |   |
| ID | Photo | Titre | Action |

Préc

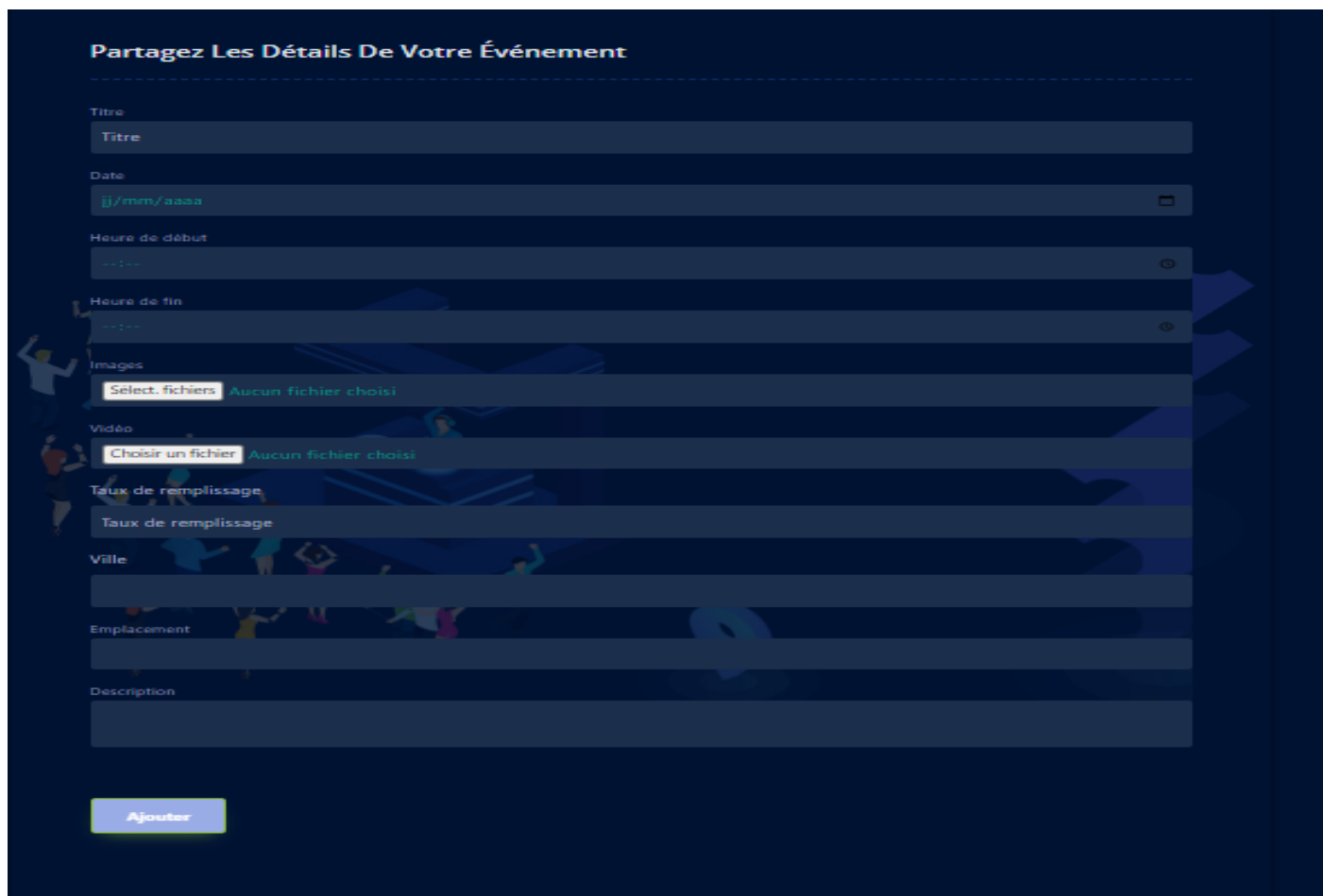
1

Suiv

Figure 3.6 :Interface Liste des catégories

En navigant vers cette page, on trouve la liste de tous les catégories. On peut faire la recherche d'une catégorie particulière à l'aide de la barre de recherche en tapant le nom.

3.5. Interface Création d'événement



The screenshot shows a web form titled "Partagez Les Détails De Votre Événement" (Share the details of your event) on a dark blue background. The form contains several input fields and buttons:

- Titre**: A text input field with the placeholder "Titre".
- Date**: A date input field with a placeholder "jj/mm/aaaa" and a calendar icon.
- Heure de début**: A time input field with a placeholder "hh:mm" and a clock icon.
- Heure de fin**: A time input field with a placeholder "hh:mm" and a clock icon.
- Images**: A file upload section with a button "Select. fichiers" and the text "Aucun fichier choisi".
- Vidéo**: A file upload section with a button "Choisir un fichier" and the text "Aucun fichier choisi".
- Taux de remplissage**: A text input field with the placeholder "Taux de remplissage".
- Ville**: A text input field.
- Emplacement**: A text input field.
- Description**: A large text area for the event description.
- Ajouter**: A green button at the bottom left to submit the form.

Figure 3.7 : Interface Création d'événement

En remplissant le formulaire, l'organisateur peut ajouter un événement.

3.6. Interface Liste des avis

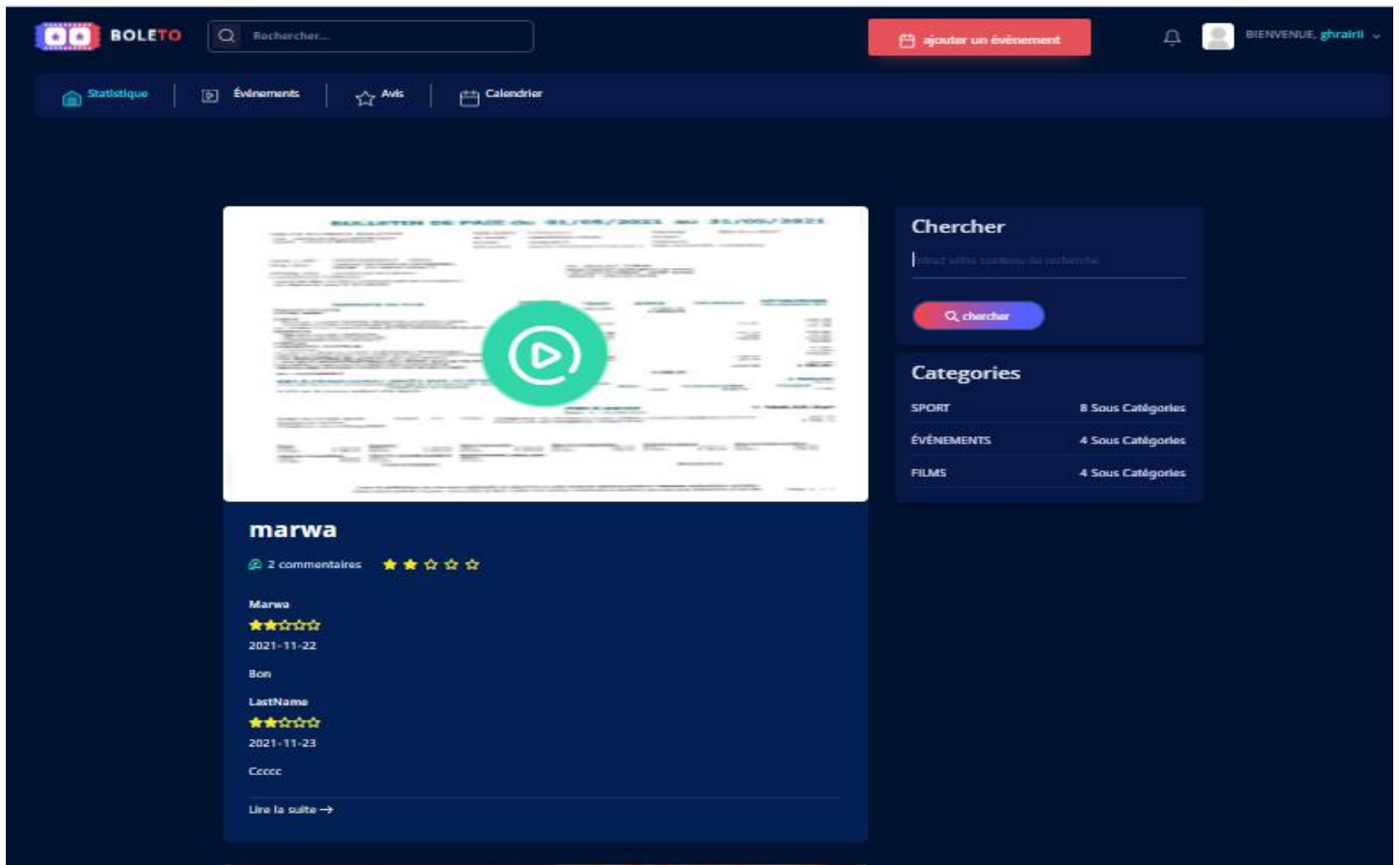


Figure 3.8 : Interface Liste des avis

En navigant vers cette page, on trouve la liste de tous les avis sur un événement.

3.7. Dashboard administrateur

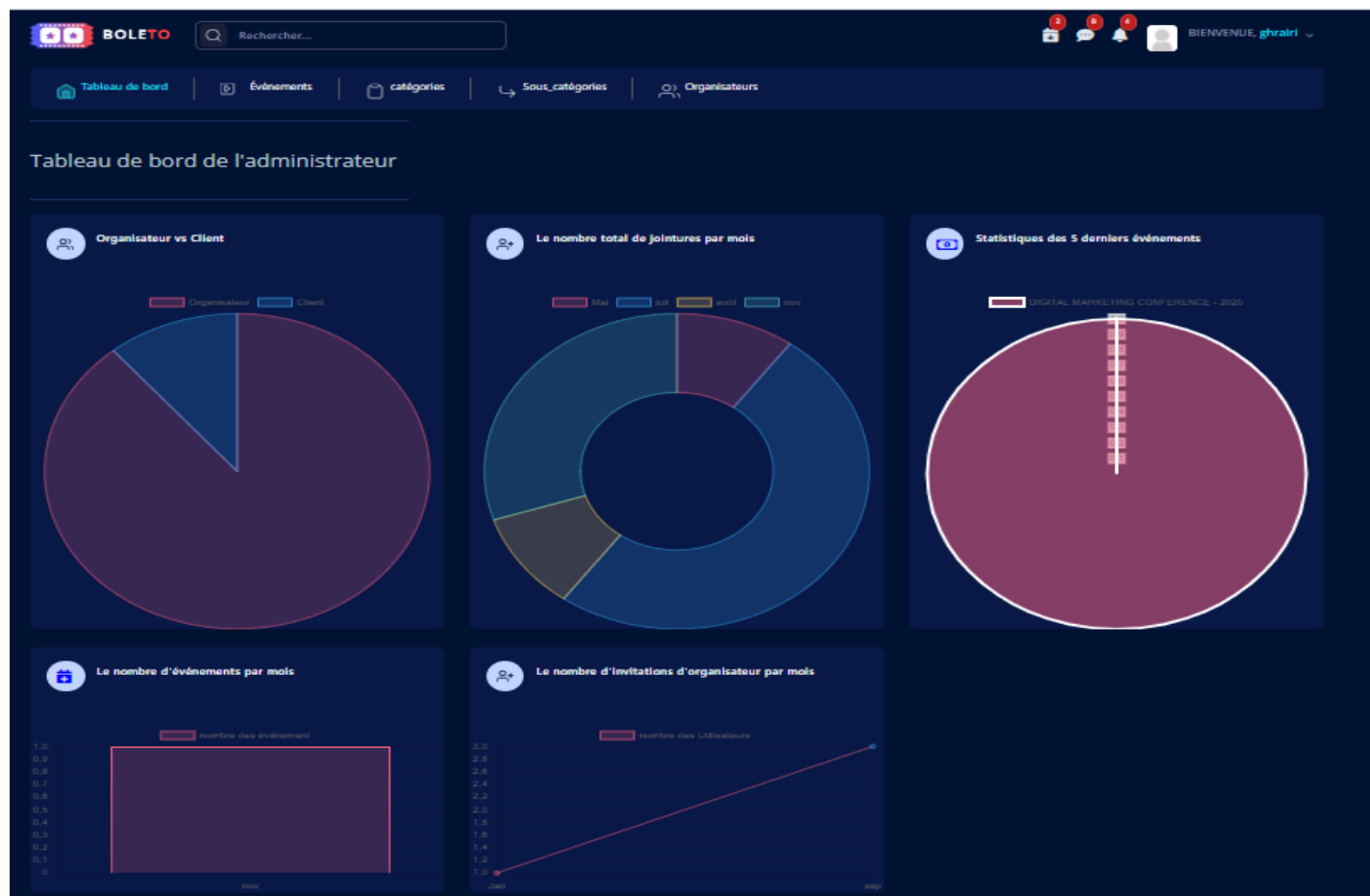


Figure 3.9 : Dashboard administrateur

Cette partie est consacrée pour l'administrateur, via cette interface il peut visualiser et gérer les utilisateurs (clients et organisateurs), les événements et les messages.

Conclusion

Au cours de ce chapitre, nous avons défini l'environnement du travail matériel et logiciel avec lequel nous avons pu implémenter cette application. Nous avons essayé par la suite d'expliquer le fonctionnement de notre système en présentant quelques imprimés écrans de quelque interface.

En achevant ce chapitre, nous nous approchons de la fin de ce mémoire qui sera clôturée par une conclusion générale.

Conclusion et perspectives

Ce rapport présente le travail que nous avons réalisé au sein de la société de développement informatique ItGate dans le cadre de notre projet de fin d'année.

L'étude conceptuelle approfondie guidée par les besoins détectés nous a permis de fonder une vue d'ensemble de projet où les composants MERN .JS (MongoDB, Express, React, Node .JS) interagissent et faisant appel à certains modèles comme MVC, MVVM. Nous avons mis en œuvre cette étude dans les différentes interfaces de notre application qui ont répondu aux besoins spécifiés.

Tout au long de la période de stage nous avons pu solliciter les compétences dans le cadre professionnel à savoir : gestion du temps et de stress, développement best practice, l'esprit d'analyse et de critique, C'était également une opportunité de mettre en œuvre les notions techniques et académiques acquises tout au long de notre formation.

Ce stage nous a donc permis de confronter les problématiques que pose le passage de la théorie à la pratique.

Dans ce terme, nous avons fixé comme perspectives l'ajout d'une partie mobile qui permet aux organisateurs de contrôler l'accès des clients le jour de l'évènement l'accès des clients.