



République Tunisienne

Ministère de l'Enseignement Supérieur

et de la Recherche Scientifique

École Supérieur Privée d'ingénierie et de
technologie

TEK-UP

Sofiatech
A Onetech company

RAPPORT DE PROJET DE FIN D'ÉTUDES

Présenté en vue de l'obtention du

Diplôme National d'Ingénieur en Sciences Appliquées et Technologiques
Spécialité : Filière

Réalisé par

Neyssen Ben Romdhane

Conception et développement d'une application de billetterie en ligne sécurisée

Encadrant professionnel : **M. Hassen BEN AMIRA**

Ingénieur Informatique

M. Abdelhalim BEN JMILA

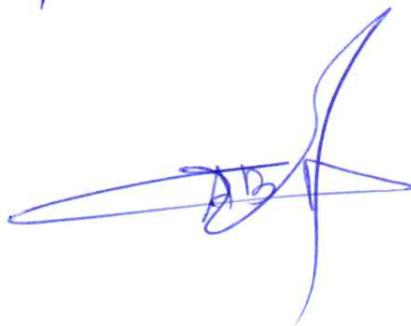
Ingénieur Informatique

Encadrant académique : **Mme Hajar SALHI**

Maître Assistante

Appréciations et signature de l'encadrant

Najess a montré d'excellentes compétences techniques en développement .Net. Elle a utilisé des technologies de pointe dans le cadre du projet. Elle est toujours ponctuelle et respecte les délais.



SOFIA TECH
Bloc B21, Technopark
Elghazela
10: 1221011 P - Tél: 71 856 900

J'autorise l'étudiant à faire le dépôt de son rapport de stage en vue d'une soutenance.

Encadrant professionnel, **M. Abdelhalim BEN JMILA**

Signature et cachet



SOFIA TECH
Bloc B21, Technopark
Elghazela
I.U. 1221011 P - Tel: 71 856 900

J'autorise l'étudiant à faire le dépôt de son rapport de stage en vue d'une soutenance.

Encadrant académique, **Mme Hajar SALHI**

Signature

Dédicace

Je dédie ce travail à :

Mes chers parents et ma sœur,

Pour votre soutien inconditionnel et votre amour qui m'ont toujours porté. Je prie pour que la vie vous offre tout le bonheur et l'amour que vous méritez.

Ma famille et mes amis,

Pour votre soutien indéfectible, vos encouragements et votre aide précieuse. Vous êtes ma force et ma motivation, et je vous en suis profondément reconnaissante.

Ma grand-mère et mon grand-père,

Votre départ durant cette période a laissé un vide immense dans mon cœur. J'ai une profonde affection pour vous, et votre mémoire continuera de briller en moi. Que vous reposiez en paix, entourés de l'amour que vous avez toujours partagé.

Ma chère Tasnim,

Ta présence et ton soutien indéfectible ont été essentiels dans ma réussite. Je te remercie du fond du cœur pour tout ce que tu as fait et continues de faire pour moi.

L'équipe de Sofiatech,

Merci pour votre aide précieuse tout au long de mon stage. Votre soutien et vos conseils m'ont été d'une grande utilité et ont contribué à ma réussite.

Remerciements

Il est particulièrement agréable, avant de présenter cette œuvre, d'exprimer toute ma gratitude envers les personnes qui, de près ou de loin, m'ont apporté leur aide inestimable lors de la réalisation de ce projet.

Je tiens à remercier tout particulièrement **Madame Hاجر Salhi**, mon encadrante universitaire, pour son soutien précieux tout au long de mon parcours. Sa disponibilité, ses conseils avisés et son encouragement constant ont grandement contribué à la réussite de mon travail.

Je souhaite remercier sincèrement **M. Hassen Ben Amira** et **M. Halim Ben Jmila**, mes encadrants professionnels. Leur soutien constant et leur expertise ont été essentiels durant mon stage. Grâce à leurs conseils et à leur enseignement, j'ai pu développer mes compétences et gagner en confiance.

Je leur suis profondément reconnaissante pour leur patience et leur disponibilité. Merci pour tout ce que vous avez fait pour moi.

Merci encore une fois

Table des matières

Introduction générale	1
1 Cadre de projet	3
Introduction	4
1.1 Cadre générale du projet	4
1.1.1 Présentation générale de l'organisme d'accueil	4
1.1.2 Présentation du projet	4
1.1.3 Cadre générale	4
1.1.4 Problématique	5
1.2 Étude de l'existant	5
1.2.1 Solutions existantes	5
1.2.2 Critique de l'existant	9
1.3 Solution proposée	10
1.4 Choix de la méthodologie du projet	11
1.4.1 Choix de l'approche Agile	11
1.4.2 Choix de la méthode Scrum	13
1.4.3 Introduction au Framework Scrum	15
1.5 Langage de modélisation : UML (Unified Modeling Language)	16
2 Analyse et spécification des besoins	18
Introduction	19
2.1 Spécification des besoins	19
2.1.1 Besoins fonctionnels	19
2.1.2 Besoins non fonctionnels	21
2.2 Diagramme de cas d'utilisation global	23
2.3 Diagramme de classe global	24
2.4 Planification de travail	25
2.4.1 Product backlog globale	25

2.4.2	Répartition des Releases	25
2.5	Architecture de l'application	26
2.5.1	Architecture logique de l'application	26
2.5.2	Architecture du Front-End	28
2.5.3	Architecture du Back-End	30
2.6	Environnement de travail	34
2.6.1	Environnement matériel	34
2.6.2	Outils de gestion de projet	34
2.6.3	Environnement logiciel	35
2.6.4	Images docker	37
2.6.5	Les technologies utilisées	38
2.6.6	Système de gestion de base de données	41
3	Release 1 : Gestion d'utilisateurs , Gestion des types du billet et des catégories	42
	Introduction	43
3.1	Sprint 0 : Conception et analyse des besoins , Recherche et autoformation	43
3.1.1	Objectifs du Sprint 0	43
3.1.2	Backlog du Sprint 0	43
3.2	Sprint 1 : Gestion des utilisateurs et d'Authentification	43
3.2.1	Objectifs du Sprint 1 :	44
3.2.2	Backlog du Sprint 1	44
3.2.3	Technologies utilisées	44
3.2.4	Conception	45
3.2.5	Réalisation	48
3.3	Sprint 2 : Gestion des types du billet et des catégories	52
3.3.1	Objectifs du Sprint 2	52
3.3.2	Backlog du Sprint 2	53
3.3.3	Conception	54
3.3.4	Réalisation	54
4	Release 2 : Gestion des événements, sessions et stocks	57
	Introduction	58
4.1	Sprint 3 : Gestion des événements et des sessions	58
4.1.1	Objectifs du Sprint 3	58
4.1.2	Backlog du Sprint 3	58

4.2	Conception	59
4.2.1	Spécification des besoins fonctionnels	59
4.2.2	Diagramme de classe de sprint 3	59
4.2.3	Réalisation	60
4.2.4	Interfaces Ajouter d'un évènement	61
4.3	Sprint 4 : Gestion des billets et du stock	66
4.3.1	Objectifs du Sprint 4	66
4.3.2	Backlog du Sprint 4	66
4.3.3	Spécification des besoins fonctionnels	67
4.3.4	Diagramme de classes de sprint 4	68
4.3.5	Réalisation	68
4.4	Conclusion	69
5	Release 3 :Consulter les évènements,gestion de panier , des réservations et de paiement	70
5.1	Sprint 5 : Consulter les évènements et les sessions disponibles	71
5.1.1	Objectifs du Sprint 5	71
5.1.2	Backlog du Sprint 5	71
5.1.3	Technologies utilisées	71
5.1.4	Spécification des besoins fonctionnels	72
5.1.5	Réalisation	72
5.2	Sprint 6 : Gestion panier	75
5.2.1	Objectifs du Sprint 6	75
5.2.2	Backlog du Sprint 6	75
5.2.3	Conception	76
5.2.4	Réalisation	77
5.3	Sprint 7 : Gestion des réservations et de paiement	78
5.3.1	Objectifs de Sprint 7	79
5.3.2	Backlog de Sprint 7	79
5.3.3	Technologies utilisées	79
5.3.4	Conception	79
5.3.5	Diagramme d'activité pour la confirmation de réservation et paiement	81
5.3.6	Réalisation	82
6	Release 4 : Tableau de Bord, Scanning de Billets et Interface Multilingue	87

6.1	Sprint 8 : Consulter le tableau de bord et les rapports de vente	88
6.1.1	Objectifs de Sprint 8	88
6.1.2	Backlog de Sprint 8	88
6.1.3	Technologies utilisées	88
6.1.4	Spécifications des besoins fonctionnels	89
6.1.5	Réalisation	89
6.1.6	Interface “ Statistiques des sessions”	90
6.2	Sprint 9 : Scanner les billets et Interface utilisateur multilingue	92
6.2.1	Objectifs de Sprint 9	92
6.2.2	Technologies utilisées	92
6.2.3	Conception	93
6.2.4	Réalisation	94
7	Release 5 : Tests et Déploiement	99
7.1	Introduction	100
7.2	Sprint 10 : Tests unitaires et tests d'intégration	100
7.2.1	Objectifs de Sprint 10	100
7.2.2	Choix du framework xUnit	100
7.2.3	Tests unitaires	101
7.2.4	Tests d'intégration	102
7.2.5	Implémentation des tests	102
7.3	Sprint 11 : Déploiement de l'application	104
7.3.1	Objectifs de Sprint 11	104
7.3.2	Environnement de Déploiement	104
7.3.3	Introduction sur L'approche DevOps	105
7.3.4	Analyse Comparative : Docker Compose vs Docker file	107
7.3.5	La processus du pipeline CI/CD Jenkins	109
7.3.6	Mise en place du Pipeline CI/CD	110
Conclusion générale	119	
Bibliographie		120
Webographie		121
Annexes		125

Table des figures

1.1	Logo de Sofiatech	4
1.2	Interface du site Pathé	6
1.3	Interface du site TESKERTI.tn	6
1.4	Interface du site Ticketmaster	7
1.5	Interface du site Eventbrite	8
1.6	Interface du site Eventbrite	8
1.7	Cycle de vie de la méthodologie SCRUM	15
1.8	Logo de UML	16
2.1	Diagramme « Cas d'utilisation global»	23
2.2	diagramme de classe global	24
2.3	l'architecture logique de l'application	27
2.4	Architecture logique Front-End (Angular)[11]	28
2.5	Gestion d'état avec NgRx	29
2.6	Logo de Git	34
2.7	logo de github	34
2.8	Logo de Google Meet	35
2.9	Logo de Trello	35
2.10	Logo de visual studio	35
2.11	Logo de Visual Studio Code	35
2.12	Logo de Docker Desktop	36
2.13	Logo de Postman	36
2.14	Logo de SSMS	36
2.15	logo de Visual Paradigm	37
2.16	Logo de RabbitMQ	37
2.17	Logo de MailCatcher	37
2.18	Logo de C#	38
2.19	Logo de TypeScript	38

2.20 Logo de SQL	38
2.21 Enter Caption	39
2.22 Logo de SCSS	39
2.23 Logo de Angular 16	39
2.24 Logo de Angular Material	39
2.25 Logo de DotNet Core	40
2.26 Logo de DotNET Aspire	40
2.27 Logo de XUnit	40
2.28 Logo de SQL Server	41
 3.1 Diagramme de cas d'utilisation du Sprint 1	45
3.2 Diagramme de classe du Sprint 1	46
3.3 Diagramme d'activité pour l'authentification	47
3.4 Diagramme de séquence pour l'accès aux données avec le jeton	48
3.5 interface de création de compte	49
3.6 Interface mail Catcher	49
3.7 Interface "Sign In"	50
3.8 Interface OTP par Email	50
3.9 Interface "OTP Verification"	50
3.10 Interface pour saisir l'adresse électronique	51
3.11 Interface E-mail de réinitialisation	51
3.12 Interface Reset Password	52
3.13 Interface "My Profil"	52
3.14 Diagramme de cas d'utilisation de gestion des types de billets et des catégories	54
3.15 Interface liste des type de billets	55
3.16 Interface Modifier un type de billet	55
3.17 Interface liste des catégories d'événements	56
3.18 Interface Modifier une catégorie d'événement	56
 4.1 Diagramme de cas d'utilisation "Gestion des événements et des sessions"	59
4.2 Diagramme de classe de sprint 3	60
4.3 Interface Liste des évènements	61
4.4 Interface de filtrage par categorie	61
4.5 Interface Saisie des données générales de l'événement	62
4.6 Interface d'ajout des sessions et leurs billets	63

4.7	Interface de Confirmation pour l'Ajout d'Événement	63
4.8	Interface de modification d'un évènement	64
4.9	Inteface de modification des sessions	64
4.10	Interface Details d'un évènement et leur sessions	65
4.11	Interface Filtrage des sessions	65
4.12	Interface "Ajouter une session	66
4.13	diagramme cas d'utilisation du Sprint 4	67
4.14	Diagramme de classe de sprint 4	68
4.15	Interface Liste des billets	68
4.16	interface Mettre à jour les billets et le stock	69
5.1	diagramme cas d'utilisation du Sprint 5	72
5.2	Interface évènements disponibles	73
5.3	Interface top 5 Évènements	73
5.4	Recherche par pays	74
5.5	Interface détails d'un événement	74
5.6	Interface sessions disponibles	75
5.7	diagramme de cas d'utilisation du Sprint 6	76
5.8	diagramme de cas d'utilisation du Sprint 6	76
5.9	Diagramme d'activité pour l'ajout d'une réservation au panier	77
5.10	Interface Page de Réservation	78
5.11	Interface " Panier"	78
5.12	diagramme de cas d'utilisation du Sprint 7	80
5.13	Diagramme de classe de Sprint 7	80
5.14	Diagramme d'activité pour la confirmation de réservation et paiement	81
5.15	Diagramme de séquences pour le payement avec stripe	82
5.16	Interface "Page de Confirmation de Commande"	83
5.17	Interface "Page de Paiement"	83
5.18	Interface "succès du paiement"	84
5.19	Enter Caption	84
5.20	Interface“ Mes réservations”	85
5.21	Interface “ Mes Billets”	85
6.1	Diagramme de cas d'utilisation de Sprint 8	89
6.2	Interface “ Statistiques des Événements”	90

6.3	Interface “ Statistiques des sessions”	90
6.4	Interface "Tableau de Bord des Performances"	91
6.5	Interface réservations par événement	91
6.6	Interface réservations par organisateur	92
6.7	diagramme de la procédure de scan d'un billet	93
6.8	diagramme de Séquence pour le Support Multilingue	94
6.9	Interface Scanner un billet	94
6.10	Interface Un billet est valide	95
6.11	Interface Un billet est non valide	95
6.12	Interface du panier en anglais	96
6.13	Interface du panier en coréen	96
6.14	Interface mon profil en français	97
6.15	Interface mon profil en Allemagne	97
6.16	Interface mes réservations en Arabe	98
6.17	Interface mes réservations en chinois	98
7.1	la structure des projets	103
7.2	Validation des méthodes développées par les tests	104
7.3	L'approche DevOps	106
7.4	les étapes de pipeline CI/CD [48]	109
7.5	CI/CD avec le docker compose [49]	110
7.6	Cloner le dépôt Git	110
7.7	Fetch All Branches	111
7.8	Configure Git User	111
7.9	Succès de la Fusion	112
7.10	Conflit de Fusion	112
7.11	Restore Dépendances	113
7.12	Construire'Build' du projet	113
7.13	Exécution des Tests	114
7.14	logs de construction des images Docker	115
7.15	Images docker en local	115
7.16	logs de 'docker push'	116
7.17	Images dans DockerHub	116
7.18	Logs de docker-compose up	117

7.19 docker compose	117
7.20 Artefacts du Pipeline	118

Liste des tableaux

1.1	Comparaison des services de billetterie	9
1.2	Comparaison des méthodologies de gestion de projet	12
1.3	Comparaison des méthodes agiles	13
2.1	Product backlog globale	25
2.2	Répartition des Releases	26
2.3	Comparaison des Solutions d'API Gateway	31
2.4	Comparaison des Solutions de Messagerie	32
2.5	Informations sur l'Ordinateur	34
3.1	Backlog du Sprint 0	43
3.2	Backlog du Sprint 1	44
3.3	Backlog du Sprint 2	53
4.1	Backlog du Sprint 3	59
4.2	Backlog du Sprint 4	67
5.1	Backlog de Sprint 5	71
5.2	Backlog pour la Sprint 6	75
5.3	Backlog de Sprint 7	79
6.1	Backlog de Sprint 8	88
6.2	Backlog de Sprint 9	92
7.1	Comparaison des frameworks de test .NET	101
7.2	Comparaison entre Dockerfile et Docker Compose	108

Liste des abréviations

- **API** = Application Programming Interface
- **GLSI** = Génie Logiciel et Système d'Information
- **HTTP** = Hypertext Transfer Protocol
- **JSON** = JavaScript Object Notation
- **JWT** = JSON Web Token
- **MQTT** = Message Queuing Telemetry Transport
- **OTP** = One-Time Password
- **QRCode** = Quick Response Code
- **TCP** = Transmission Control Protocol
- **UML** = Unified Modeling Language

Introduction générale

Dans un monde en perpétuelle évolution, le temps est une ressource précieuse que nous cherchons à optimiser. Avec la digitalisation omniprésente, la réservation en ligne est devenue indispensable pour notre quotidien. A l'occasion de notre projet de fin d'études, nous avons développé une plateforme Web innovante de billetterie en ligne, répondant aux besoins modernes de notre société en constante évolution.

Notre objectif est de faciliter l'organisation des événements et d'assurer une réservation rapide, fiable et sécurisée. Grâce à notre application Web intuitive et sécurisée, les utilisateurs peuvent facilement réserver des places pour divers événements, formations, excursions et restaurants en quelques clics. Nous croyons fermement que la qualité de vie s'améliore lorsque l'accès aux services et événements est simplifié.

Le présent rapport sera organisé de la manière suivante :

Le premier chapitre intitulé « Cadre de projet » présentera l'organisme d'accueil, le contexte de système, ainsi que l'analyse et les critiques de l'existant, la solution proposée et la méthodologie de travail.

Le deuxième chapitre intitulé « Analyse et spécification des besoins », présentera les acteurs, les besoins fonctionnels et non fonctionnels, le diagramme de cas d'utilisation général, le diagramme de classe général et le Backlog du produit. Les autres chapitres seront consacrés à détailler les différentes Releases de notre application, organisées comme suit :

- Le troisième chapitre détaillera la Release 1, qui présente la gestion d'utilisateurs, ainsi que la gestion des types de billets et des catégories.
- Le quatrième chapitre présentera la Release 2, qui détaille la gestion des événements, des sessions et des stocks.
- Le cinquième chapitre se concentrera sur la Release 3, qui présente la gestion des réservations et des paiements.
- Le sixième chapitre sera consacré à la Release 4, qui inclut le tableau de bord, le scanning de billets et l'interface utilisateur multilingue.
- Enfin, le septième chapitre abordera la Release 5, qui traite des tests et du déploiement de l'application.

Le rapport se clôturera par une conclusion générale et des perspectives.

CADRE DE PROJET

Plan

Introduction	4
1 Cadre générale du projet	4
2 Étude de l'existant	5
3 Solution proposée	10
4 Choix de la méthodologie du projet	11
5 Langage de modélisation : UML (Unified Modeling Language)	16

Introduction

Ce chapitre vise à situer notre projet dans son cadre global, permettant ainsi une meilleure compréhension du contexte dans lequel il s'inscrit. Dans un premier temps, nous présenterons l'organisme d'accueil, SofiaTech, fournissant des informations clés sur ses domaines d'expertise. Ensuite, nous exposerons la problématique centrale à laquelle notre projet s'attaque, mettant en évidence les enjeux et les défis sous-jacents. Enfin, nous définirons la méthodologie de travail adoptée pendant la réalisation de ce stage.

1.1 Cadre générale du projet

1.1.1 Présentation générale de l'organisme d'accueil

Sofiatech, filiale du groupe Onetech group, est une société d'ingénierie, de services et de solutions informatiques spécialisée dans la conception et l'intégration dans les domaines de l'électronique, la mécatronique, les logiciels, le cloud, la transformation numérique et l'Internet des objets [1].

Notre projet est pris en charge par le département Software, dont l'expertise réside dans la conception et le développement d'applications Web évolutives. L'équipe technique dispose d'une solide maîtrise de diverses technologies de pointe, leur permettant de proposer des solutions Web robustes, parfaitement adaptées aux exigences spécifiques de chaque client. Leur expertise s'étend sur des environnements technologiques différents tels que Symfony, Spring, React, Angular et bien sûr, .NET, l'écosystème dans lequel s'inscrit notre projet de stage.

La figure 1.1 représente le Logo de l'entreprise



FIGURE 1.1 : Logo de Sofiatech

1.1.2 Présentation du projet

1.1.3 Cadre générale

À l'ère du numérique, la demande pour des solutions de billetterie en ligne simples, pratiques et sécurisées n'a jamais été aussi forte. Les événements, qu'ils soient culturels, sportifs ou professionnels, attirent un nombre croissant de participants désireux de réserver leurs places de manière efficace.

C'est dans ce contexte que notre organisme a identifié une opportunité de proposer une plateforme de billetterie en ligne révolutionnaire, répondant aux exigences les plus strictes en matière de convivialité, de sécurité et de fiabilité.

1.1.4 Problématique

Bien que le marché de la billetterie en ligne soit en pleine expansion, il existe encore de nombreux défis à relever. Les systèmes existants sont souvent complexes, peu intuitifs et ne répondent pas aux normes de sécurité les plus élevées. Les acheteurs potentiels sont confrontés à des interfaces encombrées, à des processus de paiement peu sécurisés et à un manque de transparence sur les frais et les politiques de remboursement. De plus, les organisateurs d'événements rencontrent des difficultés pour gérer efficacement leurs inventaires, personnaliser leur offre et analyser les données de vente de manière approfondie. Comment assurer une expérience utilisateur satisfaisante pour l'acheteur ainsi que l'organisateur de l'événement afin de trouver leurs besoins.

1.2 Étude de l'existant

1.2.1 Solutions existantes

Après avoir effectué des recherches sur Internet, nous avons trouvé quelques exemples d'applications ayant un objectif similaire à notre projet.

Voici quelques applications qui permettent la réservation en ligne et la génération des tickets, parmi lesquels on site :

Pathé

Pathé est un site dédié aux films et événements cinématographiques, permettant aux utilisateurs de découvrir les dernières sorties et les événements spéciaux. Il offre une interface conviviale pour consulter les horaires des séances et réserver des billets.

Interface du site : La figure 1.2 représente l'interface d'accueil du site Pathé. [2]

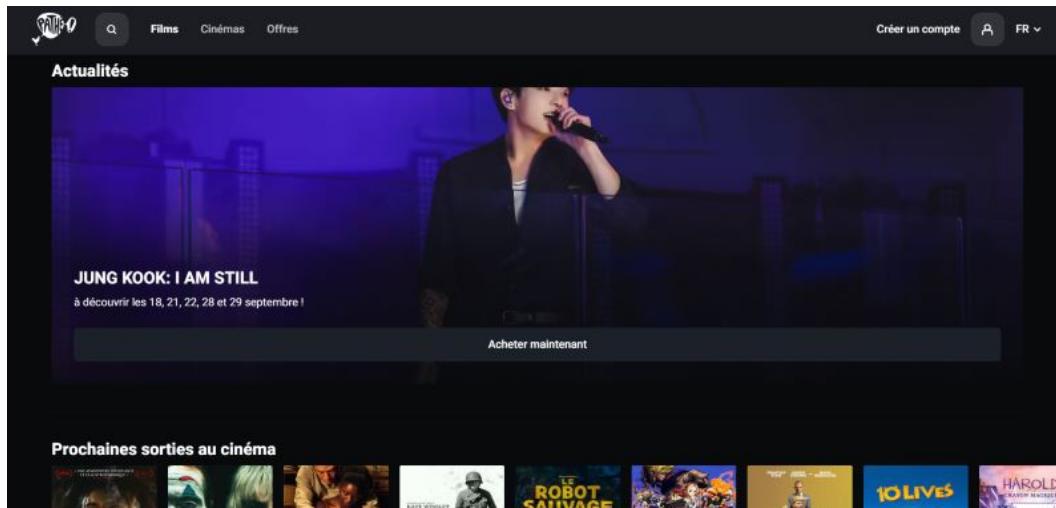


FIGURE 1.2 : Interface du site Pathé

TESKERTI.tn

TESKERTI.tn est une plateforme tunisienne dédiée à la réservation de billets pour divers événements culturels, artistiques et de divertissement. Elle permet aux utilisateurs de découvrir une large gamme d'événements, y compris des concerts, des pièces de théâtre, des festivals, des expositions et des spectacles. Le site offre une interface intuitive pour consulter les programmes, choisir les meilleurs sièges, et acheter des billets en ligne de manière sécurisée.

Interface du site : La figure 1.3 représente l'interface d'accueil du site TESKERTI.tn.[3]

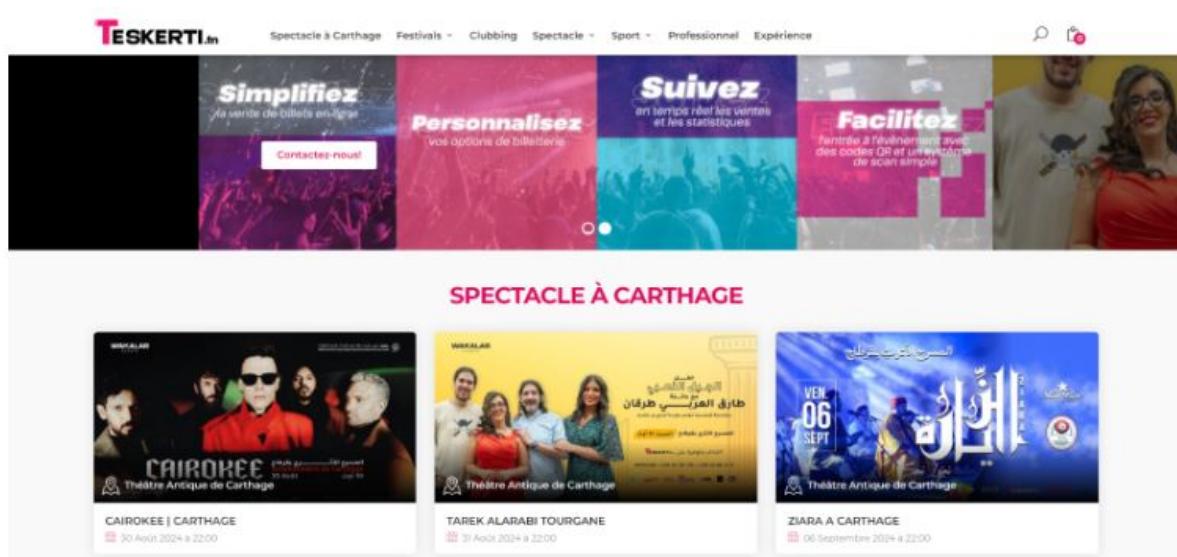


FIGURE 1.3 : Interface du site TESKERTI.tn

Ticketmaster

Ticketmaster est une plateforme de vente de billets pour une variété d'événements, y compris concerts, sports, théâtre et festivals. La plateforme offre des fonctionnalités avancées telles que le choix des sièges interactifs, des alertes sur les événements à venir, et des options de revente de billets. Ticketmaster collabore avec de nombreux organisateurs d'événements et des lieux de spectacle pour offrir un accès privilégié à des expériences live, garantissant ainsi une expérience de réservation fluide et fiable pour ses utilisateurs.

Interface du site : La figure 1.4 représente l'interface d'accueil du site Ticketmaster [4]

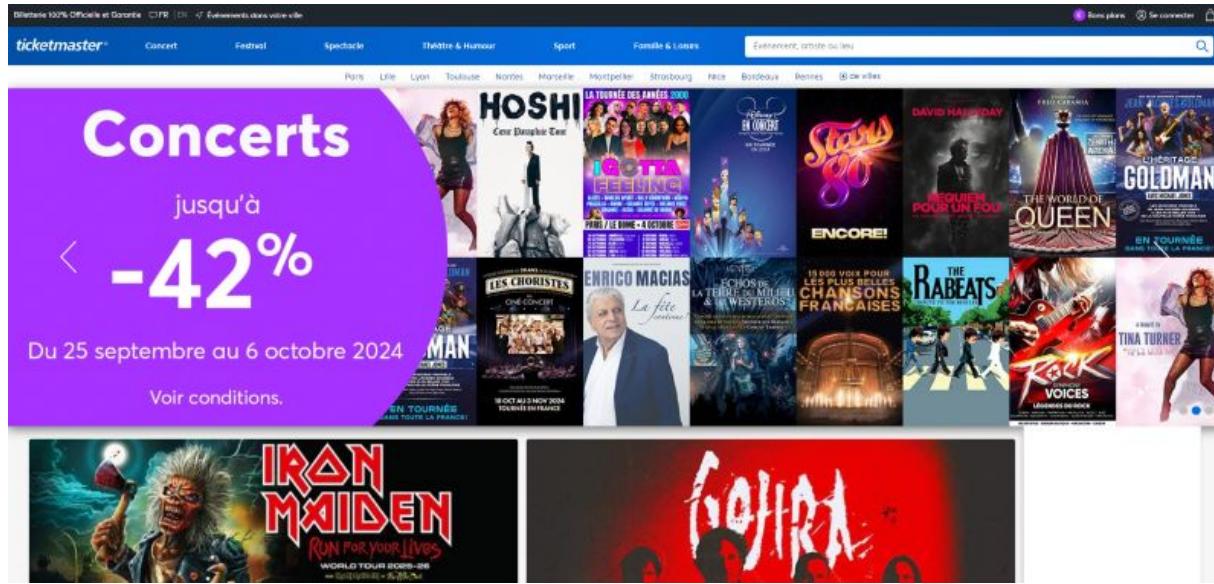


FIGURE 1.4 : Interface du site Ticketmaster

Eventbrite

Eventbrite est une plateforme mondiale de gestion et de billetterie d'événements qui permet aux organisateurs de créer, promouvoir, et vendre des billets pour divers types d'événements, y compris des conférences, des ateliers, des concerts, des fêtes et des événements sportifs. Fondée en 2006, Eventbrite offre des outils puissants pour la gestion d'événements, tels que la création de pages d'événements personnalisées, des options de billetterie flexible, des outils de marketing intégrés, et des solutions de paiement sécurisées.

Interface du site : La figure 1.5 représente l'interface d'accueil du site Eventbrite [5]

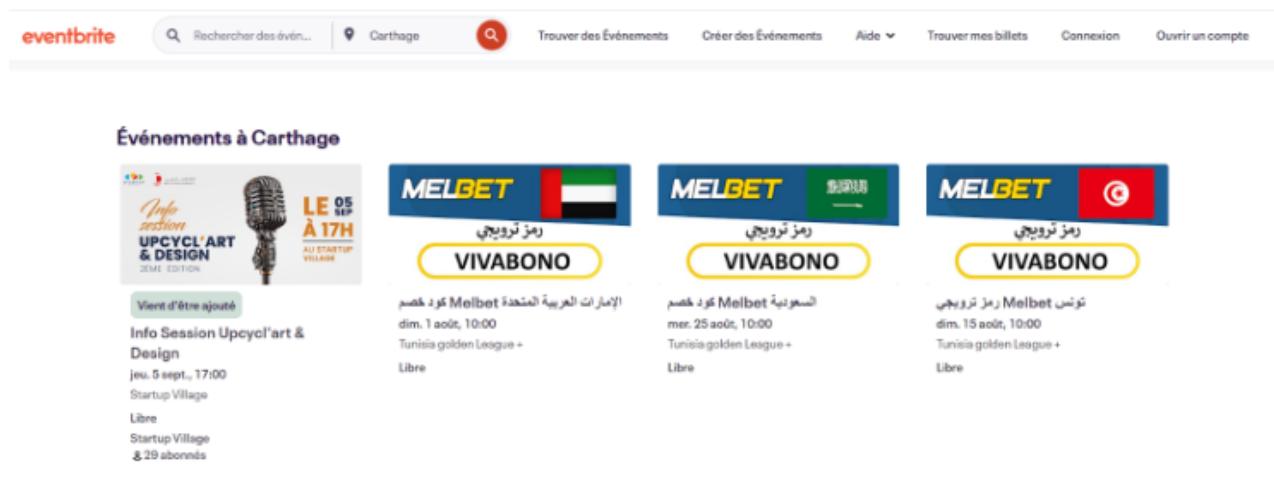


FIGURE 1.5 : Interface du site Eventbrite

My Funny Days

My Funny Days est une plateforme en ligne spécialisée dans la vente de billets pour des événements de divertissement et de loisirs. Elle propose une large gamme d'activités, y compris des spectacles, des festivals, des activités pour enfants, et des événements familiaux. My Funny Days se distingue par son interface conviviale qui permet aux utilisateurs de facilement explorer les événements disponibles, de consulter les détails et de réserver leurs billets en ligne en toute sécurité.

Interface du site : La figure 1.6 représente l'interface d'accueil du site My Funny Days [6]

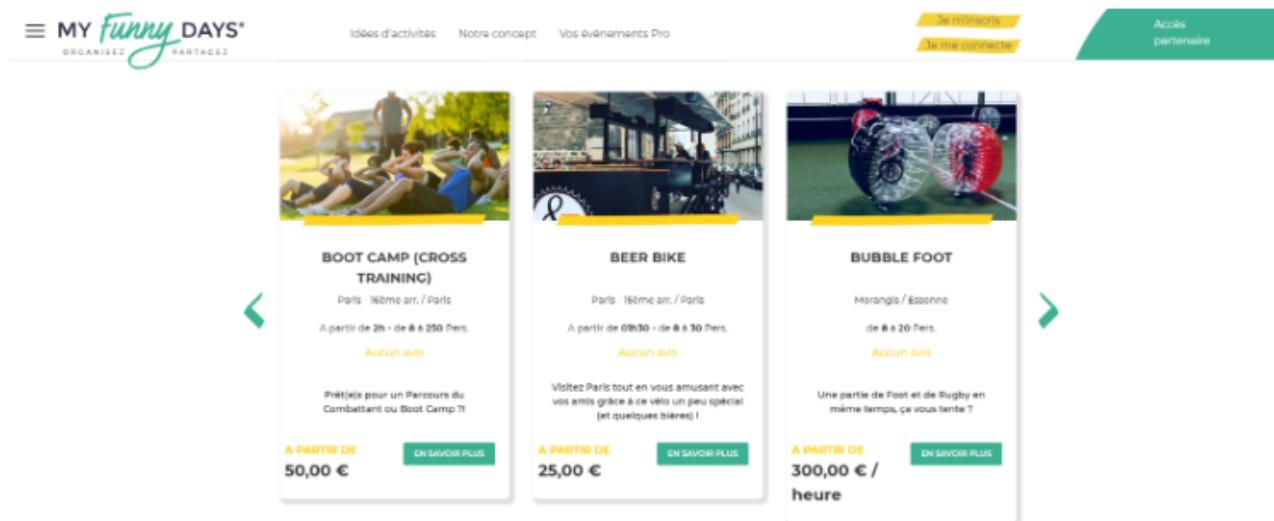


FIGURE 1.6 : Interface du site Eventbrite

1.2.2 Critique de l'existant

Le tableau 1.1 présente une comparaison des différentes applications existantes en fonction de plusieurs critères, indiquant pour chaque application si les fonctionnalités sont présentes ou non .

1. Réservation en ligne des places
2. Gestion des paniers
3. Génération des billets
4. Paiement en ligne et sécurisé
5. Recommandations personnalisées
6. Interface multilingue
7. Événements internationaux
8. Des options pour différents types d'événements
9. Fonctionnalités avancées pour la gestion des événements
10. Téléchargement des billets et détails d'événements pour un accès hors ligne

Critères	1	2	3	4	5	6	7	8	9	10
Pathé										
TESKERTI.tn										
Ticketmaster										
Eventbrite										
My Funny Days										

TABLEAU 1.1 : Comparaison des services de billetterie

Les solutions de billetterie en ligne disponibles aujourd'hui remplissent efficacement certaines des fonctions essentielles, telles que la réservation de places et le paiement sécurisé. Cependant, malgré

leur performance, ces plateformes souffrent de plusieurs limitations. Par exemple, beaucoup n'intègrent pas d'interface multilingue, ce qui réduit leur accessibilité pour un public international. De plus, certaines applications n'offrent pas de fonctionnalités avancées dédiées aux organisateurs d'événements, limitant la flexibilité dans la gestion des événements.

Par ailleurs, d'autres critères importants, comme la gestion d'options pour différents types d'événements ou la possibilité de télécharger les billets pour un accès hors ligne, sont parfois absents, ce qui impacte directement l'expérience utilisateur.

Ces limitations créent des freins pour les organisateurs d'événements ainsi que pour les utilisateurs cherchant une expérience plus fluide et complète. Face à ce constat, nous avons décidé de créer une nouvelle solution de billetterie en ligne. Notre application vise à combler ces lacunes le tout est proposé à un prix raisonnable afin de rendre la solution accessible à une large gamme d'utilisateurs, tout en optimisant l'expérience utilisateur et la gestion des événements.

1.3 Solution proposée

Notre projet vise à développer une plate-forme de billetterie en ligne, combinant une expérience utilisateur exceptionnelle avec des fonctionnalités de pointe en matière de sécurité et de gestion d'événements. Cette solution innovante comprendra les éléments clés suivants :

- **Interface utilisateur intuitive et conviviale** : Une interface épurée et réactive, offrant une navigation fluide et une visualisation claire des événements, des tarifs et des informations essentielles.
- **Processus de paiement sécurisé** : Intégration de passerelles de paiement de premier ordre, garantissant la sécurité des transactions et la protection des données sensibles des clients.
- **Gestion d'événements avancée** : Un outil puissant permettant aux organisateurs de créer, de configurer et de promouvoir leurs événements, de gérer les inventaires de billets, les tarifs et les politiques de remboursement.
- **Multilinguisme et internationalisation** : La plateforme sera conçue pour être accessible dans plusieurs langues, ce qui permettra d'attirer un public international et d'améliorer l'accessibilité des événements.
- **Sécurité renforcée** : Mise en œuvre de technologies avancées, telles que l'authentification à deux facteurs (2FA) et la vérification des adresses email, pour protéger les comptes utilisateurs et sécuriser les transactions.
- **Analytique et rapports** : Outils d'analyse et de reporting pour les organisateurs et les administrateurs,

permettant de suivre les ventes de billets et d'accéder à des rapports globaux pour optimiser la gestion des événements.

Grâce à cette solution de pointe, nous visons à révolutionner l'expérience de billetterie en ligne, offrant aux organisateurs d'événements un contrôle total sur leur offre et aux participants une expérience d'achat simplifiée, sécurisée et transparente.

1.4 Choix de la méthodologie du projet

1.4.1 Choix de l'approche Agile

Dans le cadre de notre projet, il est essentiel de choisir une méthodologie de gestion adaptée aux exigences et aux objectifs. Pour cela, nous avons comparé plusieurs approches selon des critères clés tels que la flexibilité, la participation des parties prenantes, la gestion des risques et la planification. La comparaison est présentée dans le tableau 1.2 :

Critère	Approche Traditionnelle (Waterfall)	Approche Adaptative (Agile)	Approche Hybride
Flexibilité	Rigide, changements difficiles à intégrer après le début du projet	Très flexible, adaptation continue aux nouvelles exigences	Flexibilité modérée, combine des éléments fixes et adaptatifs
Livraison	Livraison unique à la fin	Livrages fréquentes à chaque itération	Livrages intermédiaires pour certaines parties et livraison finale pour d'autres
Gestion des risques	Risques identifiés au début, difficile à ajuster en cours de projet	Risques évalués continuellement, ajustements rapides possibles	Risques évalués aux étapes fixes et ajustés pendant les phases adaptatives
Participation des parties prenantes	Implication limitée au début et à la fin du projet	Implication continue, feedback régulier	Participation modérée à certaines phases clés

Complexité du projet	Adaptée aux projets simples ou avec des exigences bien définies	Idéale pour les projets complexes ou à exigences incertaines	Convient à des projets à complexité mixte
Planification	Planification stricte dès le début, peu d'ajustements possibles	Planification flexible et continue, ajustements à chaque itération	Planification rigide pour certaines parties et flexible pour d'autres
Temps et coûts	Délais et coûts rigides, augmentent en cas de changement	Gestion optimale grâce à l'ajustement fréquent des priorités	Mixte, coûts et délais rigides pour certaines parties, ajustables pour d'autres

TABLEAU 1.2 : Comparaison des méthodologies de gestion de projet

Après avoir analysé les différentes méthodologies de gestion de projet, nous avons opté pour l'approche **Agile** pour les raisons suivantes :

- **Idéale pour les projets complexes** : Notre projet comporte des exigences évolutives et des éléments d'incertitude. Agile permet une meilleure gestion de ces aspects grâce à sa capacité d'adaptation continue.
- **Planification flexible** : L'approche Agile permet une planification itérative et réactive, ajustée à chaque phase du projet en fonction des besoins actuels. Cela nous permet de réagir rapidement aux changements.
- **Très flexible** : Contrairement à l'approche traditionnelle, Agile est particulièrement adaptée pour intégrer des modifications en cours de route, garantissant que le produit final répond aux attentes tout en restant aligné avec les objectifs globaux du projet.
- **Participation active des parties prenantes** : Agile encourage la collaboration continue avec les parties prenantes à chaque itération. Cela garantit que les retours sont pris en compte à chaque étape, minimisant les risques d'écart entre les attentes et le produit final.

En conclusion, l'approche **Agile** offre la flexibilité, la collaboration et l'adaptabilité nécessaires à la réussite de notre projet.

1.4.2 Choix de la méthode Scrum

Dans le monde du développement logiciel, la gestion de projet est cruciale pour le succès d'une initiative. Les méthodologies agiles, qui favorisent l'adaptabilité, la collaboration et l'itération, ont gagné en popularité en raison de leur capacité à répondre aux besoins changeants des clients et des projets complexes. Parmi ces méthodologies, plusieurs frameworks se distinguent, notamment Scrum, Kanban, Extreme Programming (XP) et Lean. Chacune de ces approches présente des caractéristiques uniques qui peuvent être bénéfiques selon le contexte du projet.

Nous allons comparer ces méthodes agiles à l'aide d'un tableau comparatif présenté dans le tableau 2.2 et justifier le choix du meilleur framework pour notre projet, en tenant compte de la flexibilité, de la planification, de la participation des parties prenantes et de la gestion des projets complexes.

TABLEAU 1.3 : Comparaison des méthodes agiles

Critères	Scrum	Kanban	Extreme Programming (XP)	Lean
Structure	Cadence basée sur des sprints (périodes courtes, itératives)	Flux continu de tâches	Livrailles fréquentes et améliorations continues	Amélioration continue, élimination des gaspillages
Flexibilité	Flexibilité à l'intérieur des sprints, mais planning fixé	Très flexible, pas de cycles fixes	Très flexible, adaptable à chaque étape	Flexible, avec un accent sur l'efficacité des processus
Planification	Planification au début de chaque sprint	Planification continue	Planification détaillée pour chaque cycle d'amélioration	Planification basée sur l'élimination des inefficacités
Rôles spécifiques	Product Owner, Scrum Master, Équipe de développement	Pas de rôles définis	Équipe de développement, Coach technique	Pas de rôles formellement définis

Critères	Scrum	Kanban	Extreme Programming (XP)	Lean
Suivi du progrès	Burndown chart, Scrum meetings (Daily Stand-ups)	Suivi par tableaux Kanban	Réunions quotidiennes, suivi des tests unitaires fréquents	Suivi des métriques d'efficacité
Adapté pour	Projets complexes, équipes qui collaborent étroitement	Gestion d'équipes avec flux de travail continu	Projets nécessitant des développements rapides et testés	Projets cherchant à maximiser la valeur et réduire les pertes
Participation des parties prenantes	Feedback à chaque sprint	Feedback en continu, à tout moment	Feedback continu, test et intégration fréquents	Participation modérée, souvent à des étapes clés

Nous avons choisi **Scrum** comme cadre de gestion agile pour notre projet en raison de plusieurs facteurs clés :

1. **Idéale pour les projets complexes** : Notre projet est suffisamment complexe pour bénéficier de l'organisation en sprints, qui permet de livrer des incrémentums de produit fonctionnels régulièrement. Scrum nous permet de mieux gérer les priorités changeantes et les imprévus.
 2. **Planification claire** : La méthodologie Scrum offre une planification structurée à chaque début de sprint, tout en restant flexible à la fin de chaque itération pour ajuster les priorités si nécessaire.
 3. **Flexibilité et itérations courtes** : Grâce aux sprints, nous pouvons adapter les développements en fonction des feedbacks et des besoins changeants, ce qui est crucial dans un environnement de développement rapide comme le nôtre.
 4. **Participation continue des parties prenantes** : Les réunions régulières (Daily Scrum, Sprint Review, Sprint Retrospective) assurent une communication constante entre l'équipe et les parties prenantes. Cela permet de corriger le cap rapidement en cas de besoin.
- En conclusion, **Scrum** représente la meilleure option pour notre projet, car il combine structure et

flexibilité, favorisant un développement itératif et participatif adapté aux environnements complexes.

1.4.3 Introduction au Framework Scrum

Scrum Considéré comme un Framework de gestion de projet Agile, Scrum décrit un ensemble de réunions, d'outils et de rôles qui interagissent de concert pour aider les équipes à structurer leur travail et à le gérer[7].

Les différentes parties du Scrum :

- **Product Backlog** : Une liste priorisée des fonctionnalités et des exigences à développer.
- **Sprint Backlog** : Un sous-ensemble du Product Backlog sélectionné pour un Sprint spécifique.
- **Sprint Planning** : Une réunion pour planifier les tâches à réaliser au cours du prochain Sprint.
- **Daily Scrum** : Une réunion quotidienne de courte durée pour synchroniser les efforts et identifier les obstacles.
- **Sprint Review** : Une réunion à la fin du Sprint pour démontrer les fonctionnalités développées et recueillir les commentaires.
- **Sprint Retrospective** : Une réunion pour réfléchir sur le Sprint précédent et identifier les améliorations potentielles.

La figure 1.7 présente les différentes parties de scrum[8] :

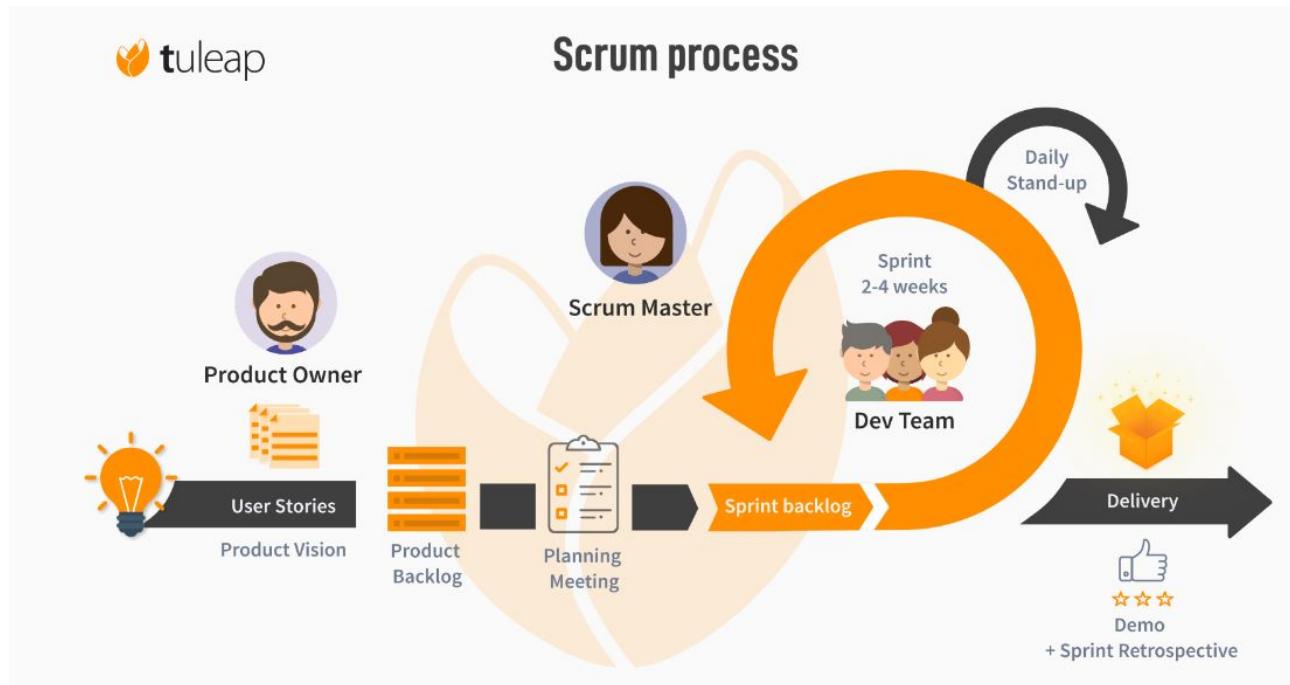


FIGURE 1.7 : Cycle de vie de la méthodologie SCRUM

Les rôles dans SCRUM La méthodologie d'agile SCRUM implique trois rôles principaux qui sont :

- **Product Owner : (Mr Hassen Ben Amira)** C'est le représentant des clients et des utilisateurs, et c'est lui qui est l'expert métier de l'équipe. C'est à lui de définir et prioriser la liste des fonctionnalités du produit et effectuer l'analyse nécessaire pour la prise des décisions.
- **Scrum Master : (Mr Halim Ben Jmila et Mr Hassen Ben Amira)** Ce sont les garants de la méthodologie de SCRUM, qui garantissent que tout le monde peut maximiser ses capacités en éliminant les obstacles et en protégeant l'équipe des perturbations externes. Par ailleurs, ils s'assurent que l'équipe chargée du projet adopte les principes et les valeurs de SCRUM.
- **Équipe : (Melle Neyssen Ben Romdhane)** L'équipe rassemble tous les rôles généralement nécessaires à un projet, elle est organisée et reste inchangée pendant la durée d'un Sprint.

1.5 Langage de modélisation : UML (Unified Modeling Language)

La modélisation du système d'information permet aux entreprises de communiquer avec des services ou des sociétés spécialisées en informatique pour décrire leurs opérations et leurs besoins. Le modèle peut être présenté de manière claire et compréhensible. Pour cela, nous avons choisi le langage de modélisation UML (Unified Modeling Language), qui repose sur la standardisation et la diversité de ses diagrammes, permettant ainsi une analyse détaillée des besoins ainsi que des vues statiques et dynamiques.

Le langage de modélisation unifié (UML) est un langage de modélisation visuelle standardisé pour la conception et la documentation de systèmes logiciels [9].

UML offre une notation graphique riche pour représenter différents aspects d'un système, tels que sa structure, son comportement, ses interactions et ses déploiements. Les diagrammes UML constituent des outils précieux pour communiquer, visualiser et documenter les spécifications d'un système, facilitant ainsi la collaboration et la compréhension entre les différentes parties prenantes d'un projet [10].



FIGURE 1.8 : Logo de UML

Conclusion

Ce chapitre a présenté le cadre général de notre projet, en abordant la problématique, l'analyse de l'existant, et la solution proposée. Nous avons souligné l'importance de l'approche Agile, avec un accent particulier sur la méthode Scrum, qui se révèle adaptée à nos besoins de flexibilité et de collaboration. Nous sommes maintenant prêts à passer à la spécification des besoins, étape cruciale pour la réussite de notre projet.

ANALYSE ET SPÉCIFICATION DES BESOINS

Plan

Introduction	19
1 Spécification des besoins	19
2 Diagramme de cas d'utilisation global	23
3 Diagramme de classe global	24
4 Planification de travail	25
5 Architecture de l'application	26
6 Environnement de travail	34

introduction

Cette phase préliminaire vise à comprendre en profondeur les exigences et les attentes des parties prenantes, qu'il s'agisse des utilisateurs finaux, des clients ou des experts métier. Cette analyse des besoins permettra de définir avec précision les fonctionnalités, les contraintes et les objectifs que le système devra satisfaire. En effet, dans cette partie nous allons présenter les acteurs de notre application, les besoins fonctionnels ainsi que les besoins non fonctionnels, tout en s'appuyant sur un diagramme de cas d'utilisation et un diagramme de classe.

2.1 Spécification des besoins

2.1.1 Besoins fonctionnels

Dans cette partie, nous allons commencer par définir les différents acteurs qui contribuent dans notre application ainsi que la définition de leurs rôles.

2.1.1.1 Identification des acteurs

Dans l'application “SofiaTicket”, les acteurs sont : Admin, Client et Organisateur. Il faut bien noter que chaque rôle possède les permissions du rôle précédent.

- **L'administrateur** : c'est l'administrateur de l'application.
- **Le client** : l'utilisateur qui va réserver des tickets.
- **L'organisateur** : celui qui va organiser des événements.

2.1.1.2 Spécification des besoins fonctionnels par acteur

Clients : Notre application permet aux clients de :

1. Gérer son compte
 - S'inscrire
 - Modifier ses Informations personnelles
 - Modifier ses Informations de connexion
2. Consulter les événements disponibles
 - Rechercher un événement
 - Filtrer les événements par catégorie et pays
3. Gérer leur panier

- Ajouter une pré-réservation au panier
 - Consulter leur panier
 - Ajuster la quantité de billets pour une pré-réservation
 - Supprimer une pré-réservation
4. Passer une réservation
 5. Payer leurs réservations pour les confirmer
 6. Consulter ses réservations
 - Consulter ses billets
 - Télécharger ses billets

Organisateurs Notre application permet aux organisateurs des événements de :

1. Gérer son compte
 - S'inscrire
 - Modifier ses Informations personnelles
 - Modifier ses Informations de connexion
2. Gérer les événements
 - Consulter les événements
 - Filtrer les événements
 - Rechercher un événement
 - Ajouter un événement
 - Modifier un événement
 - Supprimer un événement
3. Consulter les rapports de vente de ses billets
4. Scanner la validité des billets à l'entrée à l'aide de codes QR

administrateurs : Notre application permet aux administrateurs de l'application de :

1. S'authentifier
2. Consulter l'ensemble des rapports de ventes des billets
3. Consulter les coordonnées des organisateurs
4. Gérer les paramètres des événements
 - Gérer les types des billets
 - Gérer les catégories des événements

2.1.2 Besoins non fonctionnels

La spécification des besoins ne se limite pas à l'identification des acteurs et à la définition des besoins fonctionnels. D'autres limites doivent être définies pour faciliter et sécuriser l'utilisation, afin de mieux comprendre la structure et la fonctionnalité et assurer une bonne expérience utilisateur.

2.1.2.1 Sécurité

Notre application assure une sécurité robuste à travers plusieurs mécanismes clés.

Entity Framework utilise des migrations contrôlées et des requêtes paramétrées pour protéger les données contre les vulnérabilités.

Les données sensibles sont chiffrées pour garantir leur confidentialité.

L'authentification est sécurisée grâce à l'utilisation de jetons JWT et de jetons de réflexion, avec un renouvellement sécurisé et un stockage adéquat. Pour l'autorisation, des gardes sont mises en place pour vérifier les rôles et les accès des utilisateurs. Enfin, nous avons renforcé la sécurité avec l'authentification à deux facteurs (2FA) et la confirmation par email pour une protection supplémentaire.

2.1.2.2 Maintenabilité

Pour garantir que notre application puisse être facilement modifiée, corrigée ou étendue, nous avons mis en place plusieurs bonnes pratiques et stratégies de développement :

- Code Clair et Documenté : Nous écrivons un code propre et bien commenté pour faciliter la compréhension et la modification par d'autres développeurs.
- Contrôle de Version : Nous utilisons des systèmes de contrôle de version (Git) pour gérer les modifications du code, suivre les changements et faciliter la collaboration entre les développeurs.

2.1.2.3 Réutilisabilité

Notre application est conçue pour être hautement réutilisable grâce à plusieurs pratiques clés :

- Architecture Micro-services : Nous décomposons l'application en services autonomes, permettant un développement, un déploiement et une mise à jour indépendants, ce qui favorise leur réutilisation dans différents projets.
- Noms et Méthodes Génériques : L'utilisation de conventions de nommage cohérentes et de méthodes génériques nous aide à créer des composants modulaires et réutilisables, réduisant la duplication du code et augmentant la flexibilité de l'application.

2.1.2.4 Ergonomie

Notre application présente des interfaces élégantes et attrayantes, conçues pour être facilement consultables, offrant ainsi une expérience utilisateur fluide et captivante. Les utilisateurs peuvent accomplir leurs tâches de manière efficace, sans confusion ni frustration.

2.1.2.5 Performance

Notre application se distingue par sa performance élevée grâce à une architecture micro-services où chaque service possède sa propre base de données, permettant une gestion optimisée des ressources et des opérations.

L'intégration de NgRx pour la gestion des états assure une manipulation efficace des données et réduit les re-rendus inutiles, tandis que la pagination et Infinite scroll améliorent l'expérience utilisateur en optimisant le chargement et l'affichage des informations.

Cette combinaison de technologies garantit une application réactive et fluide, capable de répondre rapidement aux besoins des utilisateurs tout en maintenant une excellente performance globale.

2.1.2.6 Interface Multilingue

Notre application offre une interface multilingue pour améliorer l'accessibilité et enrichir l'expérience utilisateur. Les utilisateurs peuvent sélectionner leur langue préférée parmi plusieurs options. Les textes de l'interface se chargent dynamiquement en fonction de la langue choisie, garantissant ainsi une navigation fluide et adaptée aux préférences linguistiques de chaque utilisateur. Cette fonctionnalité vise à rendre notre application plus inclusive et à répondre aux besoins divers de notre audience internationale.

2.2 Diagramme de cas d'utilisation global

La figure 2.1 décrit le diagramme « Cas d'utilisation global ».

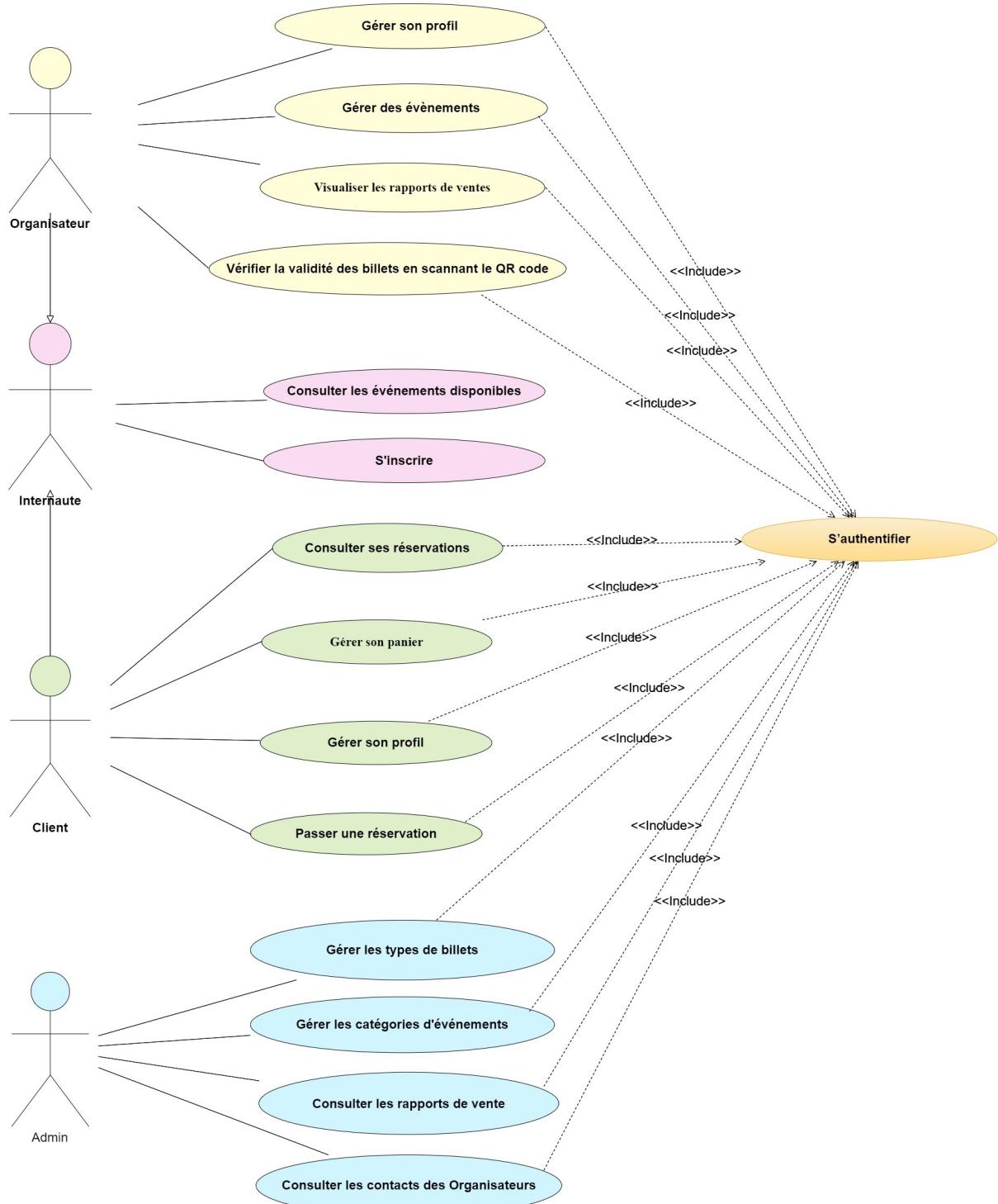


FIGURE 2.1 : Diagramme « Cas d'utilisation global »

2.3 Diagramme de classe global

La figure 2.2 présente le diagramme de classes global de notre application.

Pour une vue plus claire du diagramme, **voir annexe .1**.

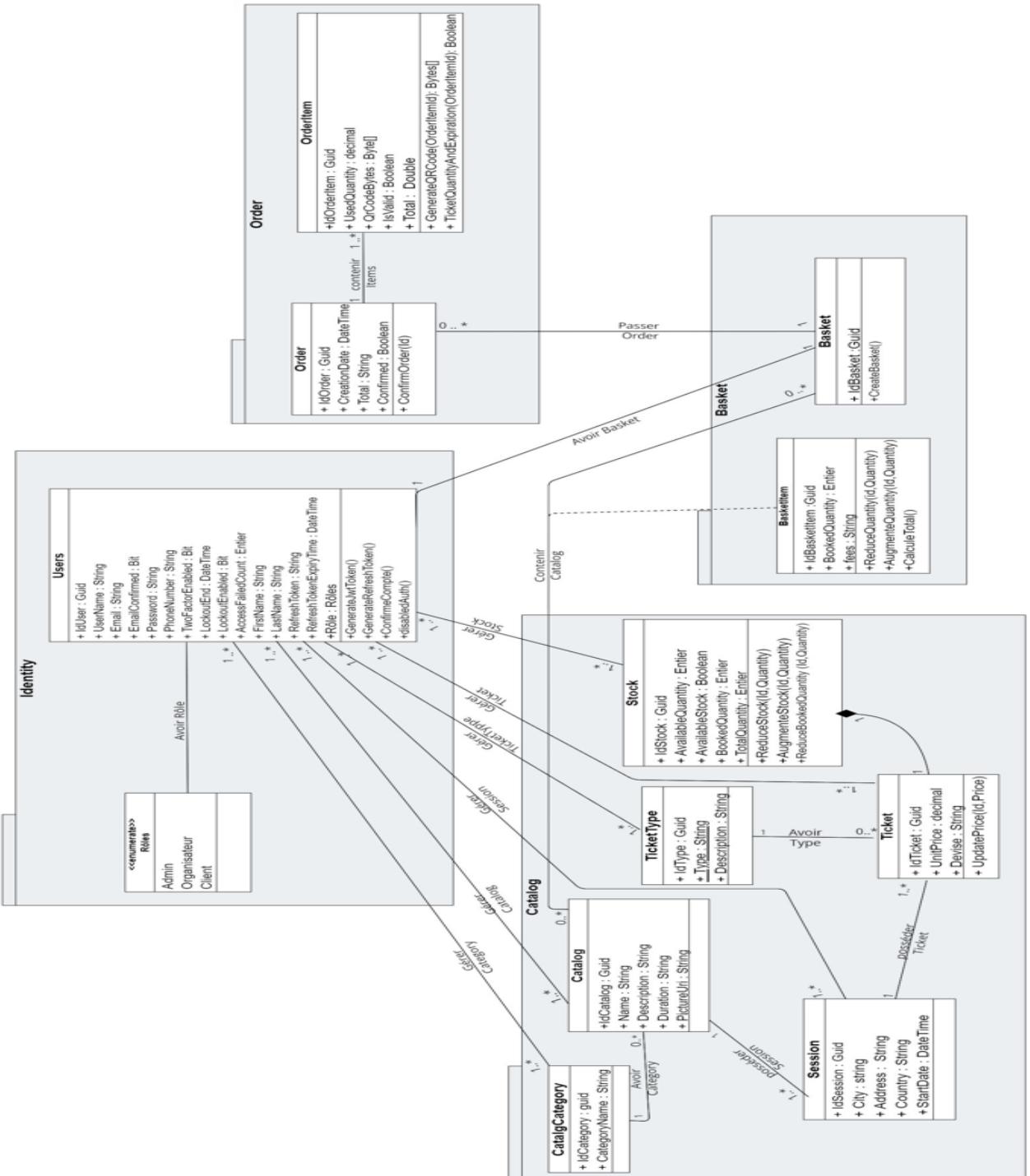


FIGURE 2.2 : diagramme de classe global

2.4 Planification de travail

2.4.1 Product backlog globale

Le backlog produit du projet est une liste organisée de toutes les tâches, fonctionnalités et exigences à réaliser pour atteindre les objectifs du projet. Examinons dans le tableau ci-dessous ce qui est inclus dans notre backlog.

Id	Nom de Sprint	Fonctionnalités	Priorité	Complexité	Estimation (Jour)
0	Recherche et auto-formation, Conception et analyse des besoins.	En tant que développeur, nous devons analyser les besoins et nous familiariser avec l'environnement logiciel de travail.	1	Difficile	15
1	Gestion des utilisateurs et d'Authentification	En tant qu'un client ou organisateur je peux m'authentifier ou m'inscrire.	1	Moyenne	20
2	Gestion des types des billets et gestion des catégories	En tant qu'un Admin je peux gérer les types des billets et les catégories des événements.	1	Facile	15
3	Gestion des événements et des sessions	En tant qu'un Organisateur je peux Gérer des événements et leurs sessions.	1	Moyenne	20
4	Gestion des billets et du stock	En tant qu'un Organisateur je peux gérer les billets et du stock.	1	Moyenne	15
5	Consulter les événements et les sessions disponibles	En tant que visiteur et client je peux consulter les événements et leur sessions disponibles.	1	Moyenne	15
6	Gestion panier	En tant qu'un client je peux Ajouter un événement au panier et le consulter.	2	Moyenne	15
7	Gestion des réservations et de paiement	En tant qu'un client je peux gérer des réservations et de paiement.	2	Difficile	20
8	Consulter le tableau de bord et les rapports de vente	En tant qu'un Organisateur je peux Consulter les rapports de vente En tant qu'un administrateur je peux consulter le tableau de bord	3	Difficile	20
9	Scanner les billets , Interface utilisateur Multilingue	En tant qu'un organisateur je peux scanner les QR code des tickets pour les valider. En tant qu'un utilisateur je peux choisir la langue du site.	4	Moyenne	15
10	Test Unitaire et d'intégration	En tant qu'un développeur je peux Tester l'application.	5	Moyenne	15
11	Déploiement de l'application	En tant que développeur je peux déployer l'application.	6	Difficile	15

TABLEAU 2.1 : Product backlog globale

2.4.2 Répartition des Releases

Le tableau 2.2 représente la répartition des Releases.

Release ID	Nom de Sprint	Estimation (Jour)
1	<ul style="list-style-type: none"> Sprint 0 : Conception et analyse de besoins et Recherche et auto-formation Sprint 1 : Gestion utilisateurs Sprint 2 : Gestion des types du billet et des catégories 	50
2	<ul style="list-style-type: none"> Sprint 3 : Gestion des événements et des sessions Sprint 4 : Gestion des billets et du stock 	35
3	<ul style="list-style-type: none"> Sprint 5 : Consulter les évènements et les sessions disponibles Sprint 6 : Gestion panier Sprint 7 : Gestion des réservations et paiement 	50
4	<ul style="list-style-type: none"> Sprint 8 : Consulter le tableau de bord et les rapports de vente Sprint 9 : Scanner les billets et Interface utilisateur multilingue 	35
5	<ul style="list-style-type: none"> Sprint 10 : Test Unitaire et d'intégration Sprint 11 : Déploiement de l'application 	30

TABLEAU 2.2 : Répartition des Releases

2.5 Architecture de l'application

2.5.1 Architecture logique de l'application

L'architecture logique de notre application est divisée en deux parties, une partie Back-end et une partie Front-end.

La partie Front-end est basée sur le fameux Framework modulaire Angular. Chaque module est composé d'un composant contenant la logique de base de la page et un template qui traite de la vue de l'application. Un composant injecte la couche service, une couche d'abstraction qui permet de gérer le logique métier de l'application. Il assure la communication avec les services du back-end, via les requêtes HTTP. Les données échangées entre la partie Front-end et la partie Back-end sont de type JSON.

La partie Back-end est structuré en microservices utilisant Aspire et .NET Core Web API. Chaque microservice gère une fonction spécifique de l'application, avec une architecture en couches qui sépare les responsabilités, améliorant ainsi la maintenabilité et la scalabilité du code. Les microservices communiquent entre eux via RabbitMQ pour la communication asynchrone et les événements. La communication avec le frontend est assurée via une API Gateway, permettant une interaction centralisée

et optimisée avec les différents services.

La figure 2.3 représente l'architecture logique de l'application.

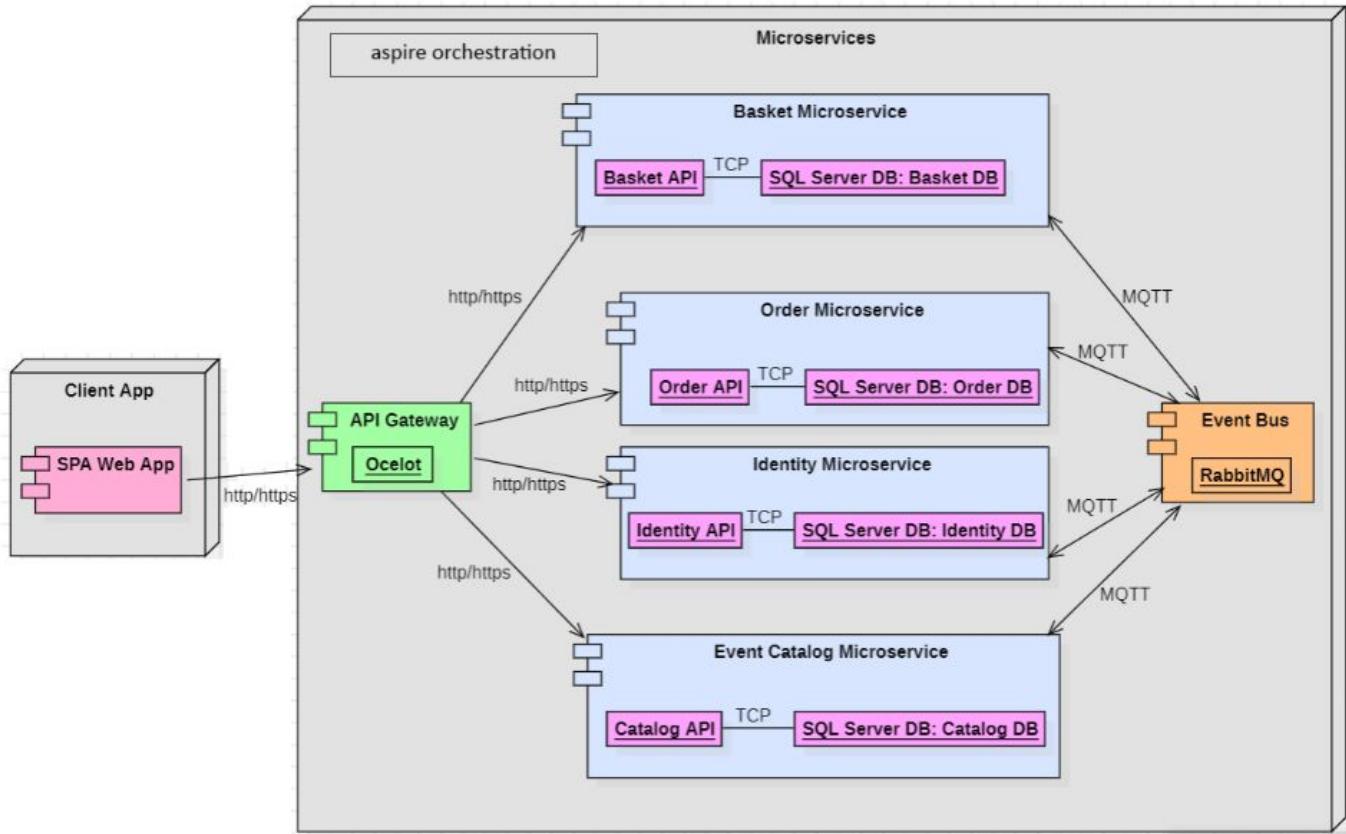


FIGURE 2.3 : l'architecture logique de l'application

Description des Composants :

- **Client App (Web app)** : L'application frontend développée avec Angular.
- **API Gateway (Ocelot)** : Point d'entrée unique pour toutes les requêtes.
Une passerelle API centralisée Ocelot pour faciliter la communication et l'intégration entre les services et les applications clientes.
- **Micro-services** :
 - Identity Service
 - EventCatalog Service
 - Basket Service
 - Order Service
 - Payment Service
- **EventBus (RabbitMQ)** : On a opté pour RabbitMQ comme bus d'événements pour la communication des messages entre les différents micro-services.

2.5.2 Architecture du Front-End

2.5.2.1 Architecture logique du Front-End

La partie Frontend qui présente l'interface de notre application utilise le Framework Angular qui repose principalement sur l'approche MVC.

En effet, Angular Permet de créer la partie frontend des applications Web de type SPA, il se présente sous la forme des composants (components) organisé par des modules.

La figure 2.4 illustre cette architecture :

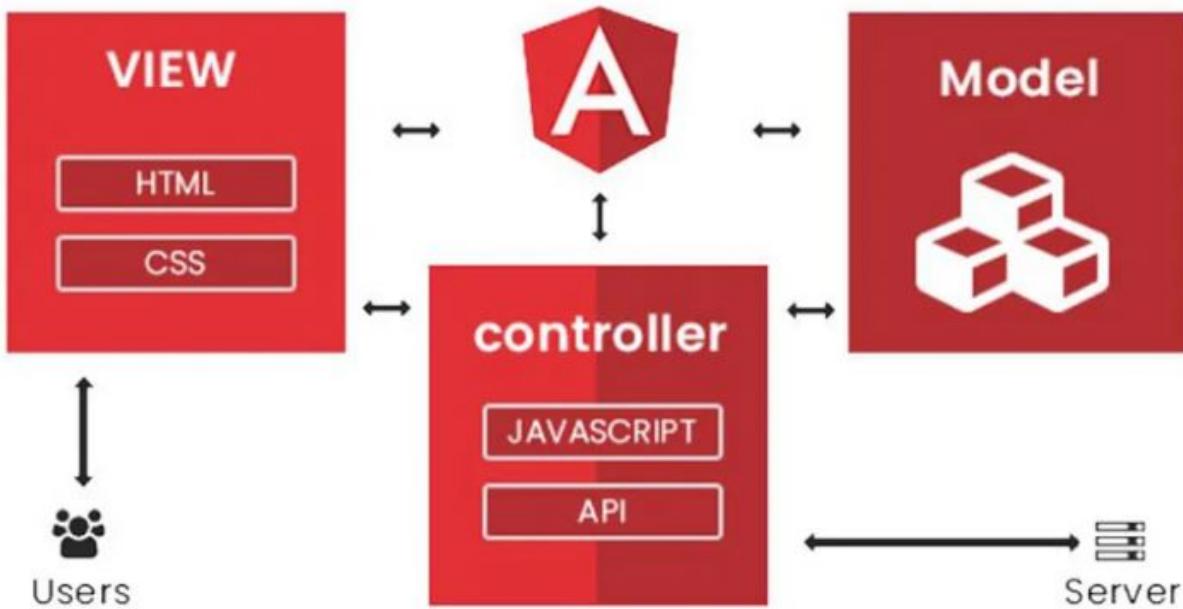


FIGURE 2.4 : Architecture logique Front-End (Angular)[11]

L'architecture MVC se compose des trois parties suivantes :

- **Module** : Chaque application Angular comprend un module racine, appelé conventionnellement AppModule, qui fournit le mécanisme d'amorçage qui lance l'application. Une application contient généralement plusieurs modules fonctionnels.
- **Vue** : C'est avec quoi l'utilisateur interagit, se nomme précisément la vue. Sa première tâche est de présenter les résultats renvoyés par le modèle. Sa seconde tâche est de recevoir toute action de l'utilisateur (hover, clic de souris, sélection d'un bouton radio, cochage d'une case, entrée de texte, mouvements, voix, etc.). Ces différents événements sont envoyés au contrôleur. La vue n'effectue pas de traitement, elle se contente d'afficher les résultats des traitements effectués par le modèle et d'interagir avec l'utilisateur.

- **Contrôleur** : Il est chargé de contrôler la relation entre les modèles et les vues. Il répond aux entrées de l'utilisateur et effectue des interactions sur les objets du modèle de données. Le contrôleur reçoit une entrée, la valide, puis exécute des opérations commerciales qui modifient l'état du modèle de données **mvc**.

2.5.2.2 Architecture Logique pour la Gestion de l'État avec NgRx

La gestion des états dans une application fait référence à la manière dont les données internes sont stockées, mises à jour et synchronisées en fonction des interactions des utilisateurs. Une gestion efficace de l'état permet à l'application de réagir rapidement aux changements, tout en garantissant que les différentes parties de l'interface utilisateur affichent les données les plus récentes.

Dans notre application Angular, nous utilisons NgRx pour la gestion d'état.

NgRx est une bibliothèque qui s'inspire du modèle Redux et centralise l'état de l'application dans un Store, maintenant ainsi un état immuable. Elle utilise des actions pour indiquer les changements, des reducers pour traiter ces actions et générer un nouvel état, et des selecteurs pour extraire des morceaux spécifiques de l'état. Cette approche assure une gestion d'état prévisible et efficace, permettant une meilleure réactivité et une maintenance simplifiée de l'application[12].

la figure 2.5 représente le flux global de l'état de l'application dans NgRx .

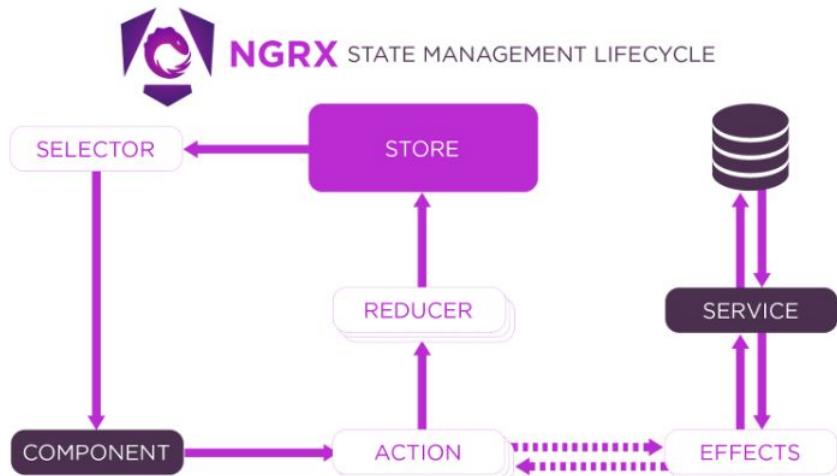


FIGURE 2.5 : Gestion d'état avec NgRx

Le diagramme illustre le cycle de vie de la gestion d'état (state management lifecycle) dans NgRx, un framework de gestion d'état pour les applications Angular. Il représente les différents composants impliqués dans ce cycle et leurs interactions.

Le composant (Component) déclenche une action (Action) en réponse aux interactions de l'utilisateur ou à d'autres événements. Cette action est ensuite traitée par un réducteur (Reducer), qui

met à jour le magasin d'état central (Store) en conséquence. Le sélecteur (Selector) permet d'extraire les données pertinentes du magasin pour être utilisées par les composants. Les effets (Effects) sont responsables des tâches secondaires, telles que les appels de service (Service) ou les interactions avec des sources de données externes.

Cette structure permet une organisation claire et une séparation des préoccupations dans la gestion de l'état de l'application, où les données sont importées une seule fois et stockées dans le store. La communication avec la base de données se fait à travers les effets, la visualisation des données à travers les sélecteurs, et le reducer est un intermédiaire entre notre store et les actions.

2.5.3 Architecture du Back-End

2.5.3.1 Architecture logique de backend

L'architecture logique du backend est construite autour d'une série de micro-services, chacun étant responsable d'une partie spécifique du domaine fonctionnel de l'application. Cette approche modulaire permet de déployer, de mettre à jour, et de maintenir indépendamment chaque service.

Micro-Services

- **Identity Service** : Responsable de l'authentification et de l'autorisation des utilisateurs. Ce service gère les jetons d'accès et les revendications des utilisateurs.
- **EventCatalog Service** : Gère les événements, y compris la création, la mise à jour, et la suppression des événements.
- **Basket Service** : Gère les paniers d'achat des utilisateurs. Il permet d'ajouter, de mettre à jour, ou de retirer des articles du panier.
- **Order Service** : Responsable de la gestion des commandes, y compris leur création.
- **Payment Service** : Gère le traitement des paiements pour les commandes effectuées par les utilisateurs.

Chaque service communique via des API REST exposées à travers une passerelle API (Ocelot), garantissant une interaction fluide entre le frontend et les différents services.

API Gateway

L'APIGateway est responsable de la gestion des appels API entrants et de la distribution vers les microservices appropriés. Elle centralise la gestion des requêtes, la sécurité, et les politiques de routage.

Choisir le Bon API Gateway : Étude Comparative

API Gateway	Intégration Cloud	Support de Protocoles	Sécurité	Scalabilité	Performance	Facilité de Configuration	Open Source
Amazon API Gateway	Intégration avec AWS	REST, WebSocket	IAM, WAF, contrôle d'accès	Haute (auto-scaling)	Excellent pour les services AWS	Interface intuitive avec la console AWS	Non
Azure API Management	Intégration avec Azure, gestion des API	REST, SOAP	Authentification, SSL/TLS	Évolutif avec Azure	Performances optimisées	Portail Azure convivial	Non
Boomi API Management	Intégration cloud-native, gestion des API	REST, SOAP	Contrôle d'accès basé sur les rôles	Scalabilité via le cloud	Bonne performance	Interface conviviale	Non
Google API Gateway	Intégration avec Google Cloud, gestion des microservices	REST	Authentification, contrôle d'accès	Évolutif avec Google Cloud	Haute performance	Facilité d'intégration avec GCP	Non
IBM API Connect	Intégration avec IBM Cloud, gestion des API	REST, SOAP	SSL/TLS, contrôle d'accès	Évolutif avec IBM Cloud	Performances stables	Interface conviviale	Non
Flex Gateway	Intégration multi-cloud, orchestration des flux de données	REST, SOAP	Authentification, sécurité des données	Évolutif et flexible	Performances adaptées	Interface intuitive	Non
Ocelot	Intégration avec .NET Core, gestion des routes	HTTP, HTTPS	Authentification JWT, API Key	Évolutif avec Docker	Très bonne pour les microservices	Configuration JSON simple	Oui

TABLEAU 2.3 : Comparaison des Solutions d'API Gateway

Depuis cette comparaison, illustrée par le tableau 2.3 , nous constatons que Ocelot est spécifiquement conçu pour les applications .NET Core, ce qui en fait une option idéale pour notre projet. De plus, en tant que solution open-source, Ocelot est légère, facilement adaptable et sans frais. Sa configuration basée sur des fichiers JSON simplifie sa mise en place et sa maintenance. Avec ses fonctionnalités de routage robustes, y compris la transformation et la gestion des taux de requêtes, Ocelot offre une grande flexibilité pour répondre aux besoins de routage. Ainsi qu'il permet la gestion des authentification et d'autorisation avec OAuth2 qui est une condition primordiale dans notre solution.

EventBus

EventBus est Utiliser pour la communication asynchrone entre les microservices, l'EventBus facilite la gestion des événements et permet une interaction non bloquante entre les différents services.

Choisir le Bon EventBus : Étude Comparative

Le tableau 2.4 présente une comparaison entre les différents Eventbus possibles pour la communication entre les micro-services en se concentrant sur les critères de performance, sécurité, configuration.

Caractéristique	RabbitMQ	Apache Kafka	Azure Service Bus	Amazon SNS/SQS
Modèle de Messagerie	File d'attente, Pub/Sub	Pub/Sub, Stream	File d'attente, Pub/Sub	File d'attente (SQS), Pub/Sub (SNS)

Persistance des Messages	Oui	Oui	Oui	Oui
Performance	Haute	Très haute	Haute	Haute
Tolérance aux Pannes	Oui	Oui	Oui	Oui
Sécurité	SSL, Authentification	SSL, Authentification	SSL, Authentification, RBAC	SSL, Authentification
Interopérabilité	Large (protocoles AMQP, MQTT, STOMP)	Large (APIs Java, .NET, Python, etc.)	Intégré avec Azure, APIs REST	Intégré avec AWS, APIs REST
Facilité de Mise en Place	Modérée (nécessite configuration)	Complexe (nécessite cluster, Zookeeper)	Facile (service managé sur Azure)	Facile (service managé sur AWS)
Coût	Gratuit (open source)	Gratuit (open source)	Payant (basé sur utilisation)	Payant (basé sur utilisation)

TABLEAU 2.4 : Comparaison des Solutions de Messagerie

Comme conclusion de ce tableau comparatif nous avons constaté que RabbitMQ est idéal pour les systèmes nécessitant des files d'attentes robustes, ce qui est demandé dans notre cas, et des modèles de messagerie variés. Comme il est bon pour la facilité d'intégration et les options de persistance. De plus il est gratuit et simple à mettre en place ce qui lui rend un des meilleurs choix pour notre cas.

2.5.3.2 Architecture Logicielle des Micro-Services

Pour assurer cohérence et efficacité dans tous nos Micro-services, nous avons adopté une architecture en couches standardisée. Cette approche divise l'application en plusieurs couches distinctes, chacune ayant une responsabilité précise, facilitant ainsi la maintenance et l'évolution du système. Les couches de notre architecture sont les suivantes :

- **Couche de Présentation :**

- Gère les requêtes HTTP et les réponses.
- Comprend les contrôleurs API qui orchestrent les interactions entre les clients et les services backend.

- **Couche de Service :**

- Contient la logique métier de l'application.
- Les services traitent les données, appliquent les règles de gestion, et interagissent avec les repositories pour accéder aux données.

- **Couche de Domaine :**

- Représente les entités et les règles spécifiques du domaine.
- Encapsule les modèles qui reflètent les concepts et les règles de notre domaine métier.

- **Couche de Persistance :**

- Gère l'accès aux données en utilisant des ORM (Entity Framework Core).
- Les repositories dans cette couche s'occupent des opérations CRUD et interagissent avec la base de données.

- **Couche d'Infrastructure :**

- S'occupe des aspects techniques tels que la configuration de la base de données et les intégrations avec des systèmes externes.
- Gère également les migrations de la base de données.

- **DTOs (Data Transfer Objects) :**

- Utilisés pour transférer les données entre le client et le serveur.
- Aident à isoler la logique métier des détails de l'interface utilisateur.

- **Migrations :**

- Gèrent les modifications du schéma de la base de données.
- Permettent d'ajuster la structure de la base de données en parallèle avec les changements du modèle de données.

En adoptant cette architecture en couches, nous avons créé une base solide qui soutient la robustesse et la flexibilité de notre système global. Cette approche permet une séparation claire des responsabilités et simplifie la gestion et l'évolution de nos services.

2.6 Environnement de travail

2.6.1 Environnement matériel

Caractéristique	Détails
Ordinateur	Asus
Propriétaire	BEN ROMdhane Neyssen
Processeur	Intel Core i7 11th Gen
RAM	24 Go
Disque dur	1 To SSD
Système d'exploitation	Windows 11 Famille Unilingue
Carte graphique	NVIDIA MX330

TABLEAU 2.5 : Informations sur l'Ordinateur

2.6.2 Outils de gestion de projet

- **Git**

Git est un système de contrôle de version qui a été inventé et développé par Linus Torvalds, en 2005. Il s'agit d'un outil de développement qui aide une équipe de développeurs à gérer les changements apportés au code source au fil du temps [13].



FIGURE 2.6 : Logo de Git

- **GitHub**

GitHub est une plateforme open source de gestion de versions et de collaboration destinée aux développeurs de logiciels. GitHub permet aux développeurs de modifier, d'adapter et d'améliorer le logiciel gratuitement à partir de référentiels publics [14].



FIGURE 2.7 : logo de github

- **Google Meet**

Google Meet est un outil de vidéo conférence largement utilisé pour faciliter les réunions à distance, les présentations et la collaboration en équipe [15].



FIGURE 2.8 : Logo de Google Meet

- **Trello**

Trello est un outil de gestion de projet en ligne qui utilise une métaphore de tableau Kanban pour organiser les tâches, les suivre et les affecter aux membres de l'équipe [16].



FIGURE 2.9 : Logo de Trello

2.6.3 Environnement logiciel

- **Visual Studio**

Visual Studio est un environnement de développement intégré (IDE) complet créé par Microsoft. Principalement utilisé pour le développement d'applications Windows, Web et mobiles sur diverses plates-formes, il offre un large éventail d'outils et de fonctionnalités, tels que l'édition de code, le débogage, la gestion des versions et le déploiement [17].



FIGURE 2.10 : Logo de visual studio

- **Visual Studio Code**

Développé par Microsoft, Visual Studio Code est un éditeur de code source léger et polyvalent. Il prend en charge de nombreux langages de programmation et frameworks, et peut être étendu à l'aide d'extensions tierces. Sa nature multiplateforme et son interface utilisateur épurée en font un choix populaire pour les développeurs [18].



FIGURE 2.11 : Logo de Visual Studio Code

- **Docker Desktop**

Docker Desktop est une application de bureau qui facilite l'installation et la gestion de Docker sur les systèmes d'exploitation Windows et macOS. Elle permet aux développeurs de créer, déployer et exécuter des applications dans des conteneurs Docker de manière simplifiée, en fournissant une interface utilisateur conviviale et des outils de gestion [19].



FIGURE 2.12 : Logo de Docker Desktop

- **Postman**

C'est un outil pour tester et interagir avec des API RESTful. Il permet aux développeurs d'envoyer des requêtes HTTP, d'inspecter les réponses, de créer des collections de tests et de documenter les API. Postman simplifie le processus de développement et de test des API grâce à son interface utilisateur intuitive [20].



FIGURE 2.13 : Logo de Postman

- **SQL Server Management Studio (SSMS)**

SSMS est un outil logiciel intégré à Microsoft SQL Server. Il fournit un environnement complet pour configurer, gérer et administrer toutes les composantes d'une infrastructure SQL Server. Les développeurs peuvent utiliser SSMS pour créer et modifier des bases de données, écrire des requêtes, concevoir des schémas de données et surveiller les performances [21].



FIGURE 2.14 : Logo de SSMS

- **Visual Paradigm**

Visual Paradigm est un outil de modélisation logiciel qui prend en charge UML, BPMN, ERD, et d'autres diagrammes. Il est utilisé pour la conception de systèmes, la modélisation de processus métiers, et la gestion des exigences, avec des fonctionnalités de génération de code et de collaboration en équipe [22].



FIGURE 2.15 : logo de Visual Paradigm

2.6.4 Images docker

- **RabbitMQ**

RabbitMQ est un message broker open-source qui implémente le protocole AMQP (Advanced Message Queuing Protocol). Il est utilisé pour gérer la transmission et le routage de messages entre les différentes parties d'une application, souvent dans des environnements distribués ou de microservices [23].



FIGURE 2.16 : Logo de RabbitMQ

- **MailCatcher**

MailCatcher est un serveur de messagerie de développement qui capture les e-mails envoyés par une application et les affiche dans une interface Web. Il est utilisé pour tester et déboguer les fonctionnalités d'envoi d'e-mails sans les envoyer réellement à des adresses de destinataires réels. MailCatcher intercepte les e-mails sortants et les stocke dans une interface de visualisation accessible via un navigateur, permettant aux développeurs de vérifier le contenu, les en-têtes, et les formats des e-mails [24].



FIGURE 2.17 : Logo de MailCatcher

2.6.5 Les technologies utilisées

2.6.5.1 Langages de développement utilisés

- **C# :**

C# est un langage de programmation moderne, orienté objet, développé par Microsoft comme partie intégrante de la plateforme .NET. Il est largement utilisé pour créer une variété d'applications, allant des logiciels de bureau Windows aux applications Web et mobiles [25].



FIGURE 2.18 : Logo de C#

- **TypeScript :**

TypeScript est un langage de programmation open-source, créé par Microsoft, qui étend les capacités de JavaScript en ajoutant un système de types optionnel. La principale innovation de TypeScript est son système de types, qui permet de détecter les erreurs pendant la phase de développement, avant même que le code ne s'exécute dans un navigateur. Cela conduit à un code plus fiable et plus facile à maintenir [26].



FIGURE 2.19 : Logo de TypeScript

- **SQL :**

SQL (Structured Query Language) est le langage standard pour interagir avec des bases de données relationnelles. Il est crucial pour stocker, récupérer et manipuler des données structurées de manière efficace [27].



FIGURE 2.20 : Logo de SQL

- **HTML :**

HTML, ou HyperText Markup Language, est la langue maternelle du World Wide Web. Contrairement

aux autres langages mentionnés, HTML n'est pas un langage de programmation, mais un langage de balisage. Son rôle est de définir la structure et le contenu des pages Web [28].



FIGURE 2.21 : Enter Caption

- **SCSS :**

SCSS (Sassy CSS) est un langage de feuille de style qui est une extension de CSS. Il permet d'utiliser des fonctionnalités avancées telles que des variables, des imbriquations et des mixins. Cela rend l'écriture de styles plus efficace et organisée, facilitant ainsi la gestion de feuilles de style complexes [29].



FIGURE 2.22 : Logo de SCSS

2.6.5.2 Frameworks de développement et de tests

- **Angular 16**

Angular 16 est un framework de développement front-end open-source maintenu par Google. Il permet de créer des applications Web dynamiques et interactives côté client. Utilisant TypeScript, Angular 16 offre une structure robuste pour développer des applications à page unique (SPA) avec une architecture modulaire et des performances optimisées. [angular](#)



FIGURE 2.23 : Logo de Angular 16

- **Angular Material**

Angular Material est un framework de composants d'interface utilisateur pour Angular, basé sur les principes du design Material de Google. Il fournit une collection de composants d'interface prêts à l'emploi, comme des boutons, des formulaires, des cartes, des barres de navigation, et bien d'autres, qui respectent les standards du design Material[30].



FIGURE 2.24 : Logo de Angular Material

- **ASP.NET Core 8**

ASP.NET Core 8 est un framework de développement Web Open source et multiplateforme développé par Microsoft. Il est conçu pour créer des applications Web modernes, évolutives et performantes [31].



FIGURE 2.25 : Logo de DotNet Core

- **.NET Aspire**

.NET Aspire est un framework développé par Microsoft, introduit en 2023, conçu pour simplifier le développement, l'orchestration et la gestion des microservices au sein des applications .NET. Il fournit des outils et des abstractions pour faciliter la gestion des API, l'intégration des services et la communication entre les services. Ce framework représente une évolution récente dans l'écosystème .NET, ciblant l'amélioration de la gestion des microservices et des API dans les applications modernes[32].



FIGURE 2.26 : Logo de DotNET Aspire

- **xUnit**

xUnit est un framework de tests unitaires pour les applications .NET, permettant aux développeurs d'écrire et d'exécuter des tests pour assurer la qualité et la fiabilité du code. Il utilise des attributs pour désigner les méthodes de test et inclut des assertions pour vérifier les résultats attendus. xUnit favorise les bonnes pratiques de test et permet une exécution rapide des tests en parallèle.
[33]



FIGURE 2.27 : Logo de XUnit

2.6.6 Système de gestion de base de données

- **SQL Server** SQL Server est un système de gestion de bases de données relationnelles (SGBDR) développé par Microsoft. Il est conçu pour stocker, récupérer et gérer des données dans des applications variées, allant des applications de petite taille aux systèmes d'entreprise à grande échelle. SQL Server offre des fonctionnalités avancées telles que la sécurité des données, la gestion des transactions, et des outils de reporting et d'analyse [34].



FIGURE 2.28 : Logo de SQL Server

Conclusion

La phase de spécification des exigences est cruciale dans le cycle de vie d'un projet car elle permet d'avoir une meilleure compréhension du système et des principales caractéristiques à atteindre. Dans ce contexte, j'ai opté pour l'utilisation d'un backlog produit et de la planification des Releases afin de faciliter la transition vers la phase de conception, nous passons maintenant à la réalisation du premier Release "Gestion d'utilisateurs , Gestion des types du billet et des catégories".

RELEASE 1 : GESTION D'UTILISATEURS , GESTION DES TYPES DU BILLET ET DES CATÉGORIES

Plan

Introduction	43
1 Sprint 0 : Conception et analyse des besoins , Recherche et autoformation	43
2 Sprint 1 : Gestion des utilisateurs et d'Authentification	43
3 Sprint 2 : Gestion des types du billet et des catégories	52

introduction

Au cours de ce chapitre, nous allons présenter les différentes étapes de réalisation des Sprints de notre projet. Le premier Sprint, Sprint 0 : Conception et analyse de besoins, sera consacré à la recherche et à l'auto-formation nécessaires pour poser les bases de notre application. Ensuite, dans le Sprint 1 : Gestion des utilisateurs, nous développerons les fonctionnalités relatives à la gestion des utilisateurs et à l'authentification. Enfin, le Sprint 2 : Gestion des types du billet et des catégories abordera la mise en place des différentes catégories de billets et leur gestion.

3.1 Sprint 0 : Conception et analyse des besoins , Recherche et autoformation

Dans cette section nous allons présenter les différentes étapes de la réalisation du Sprint 0.

3.1.1 Objectifs du Sprint 0

L'objectif de Sprint d'analyse de besoins et recherche et autoformation est citer la première tâche réaliser par l'équipe de développement avant la réalisation de projet pour bien se documenter sur les plateformes similaires existantes, et à s'autoformer sur les technologies de développement et l'environnement de travail.

3.1.2 Backlog du Sprint 0

Id	Fonctionnalités (Technical Stories)	Priorité	Estimation (Jour)
0	Recherche et Documentation sur les plateformes similaires	1	2
1	Spécification des besoins et élaboration de premier version du Product backlog	1	3
2	Spécification des technologies et des outils de travail	2	3
3	Installation de l'environnement du travail	2	3
4	Autoformation sur les technologies à utiliser	3	4

TABLEAU 3.1 : Backlog du Sprint 0

3.2 Sprint 1 : Gestion des utilisateurs et d'Authentification

Dans cette section nous allons présenter les différentes étapes permettant la réalisation du Sprint "Gestion des utilisateurs et d'Authentification".

3.2.1 Objectifs du Sprint 1 :

L'objectif de ce Sprint est de mettre en place un système d'authentification sécurisé et de gérer les droits d'accès des utilisateurs.

3.2.2 Backlog du Sprint 1

Id	Fonctionnalités (User Stories)	Priorité	Estimation (Jour)
0	En tant qu'un client ou organisateur, je peux créer un compte pour se bénéficier des fonctionnalités offerte par "SofiaTicket"	1	5
1	En tant que client ou organisateur, je peux confirmer mon compte par un email de vérification.	1	5
2	En tant qu'un client ou organisateur de l'application je peux m'authentifier	2	5
3	En tant qu'un client ou organisateur, je peux gérer mon profil	3	5
4	En tant qu'un client ou organisateur, je veux réinitialiser mon mot de passe afin de récupérer mon compte en cas d'oubli de mot de passe	4	5

TABLEAU 3.2 : Backlog du Sprint 1

3.2.3 Technologies utilisées

- **ASP.NET Core Identity** : ASP.NET Core Identity est un framework puissant et flexible pour gérer l'authentification et l'autorisation dans les applications ASP.NET Core. Il offre une architecture modulaire qui permet aux développeurs de personnaliser et d'étendre les fonctionnalités selon les besoins spécifiques de leurs applications. Sa conception repose sur des principes de sécurité solides, assurant que les informations sensibles des utilisateurs sont protégées [35] .
- **JSON Web Token** : Un JSON Web Token est un jeton d'accès, qui permet un échange sécurisé de données entre deux ou plusieurs parties. Les JWT sont particulièrement appréciés pour les opérations d'identification [36]. Cette sécurité se traduit par la vérification de l'intégrité des données à l'aide d'une signature numérique, qui vérifie si le message a été modifié pendant le transfert, et authentifie également l'expéditeur du JWT dans le cas d'un jeton signé avec une clé privée.
- **jwt.io** : JWT.io est un outil en ligne pratique pour travailler avec les JSON Web Tokens (JWT). Il permet de décoder, valider, et générer des JWT facilement. En collant un JWT dans le champ de texte, le site décode et affiche automatiquement les en-têtes, la charge utile, et la signature. De plus, vous pouvez aussi vérifier la validité d'un JWT [36].
- **Auth Guard** : Un Auth Guard (Gardien d'authentification) est un mécanisme dans Angular qui protège les routes en s'assurant que seuls les utilisateurs authentifiés peuvent accéder à des

parties spécifiques de l'application [37].

- **Role Guard** : Un Role Guard (Gardien de rôle) est un type spécifique de garde de route dans Angular qui vérifie non seulement si un utilisateur est authentifié, mais aussi s'il possède les rôles ou permissions requis pour accéder à une route donnée [37].

3.2.4 Conception

3.2.4.1 Diagramme de cas d'utilisation du Sprint 1

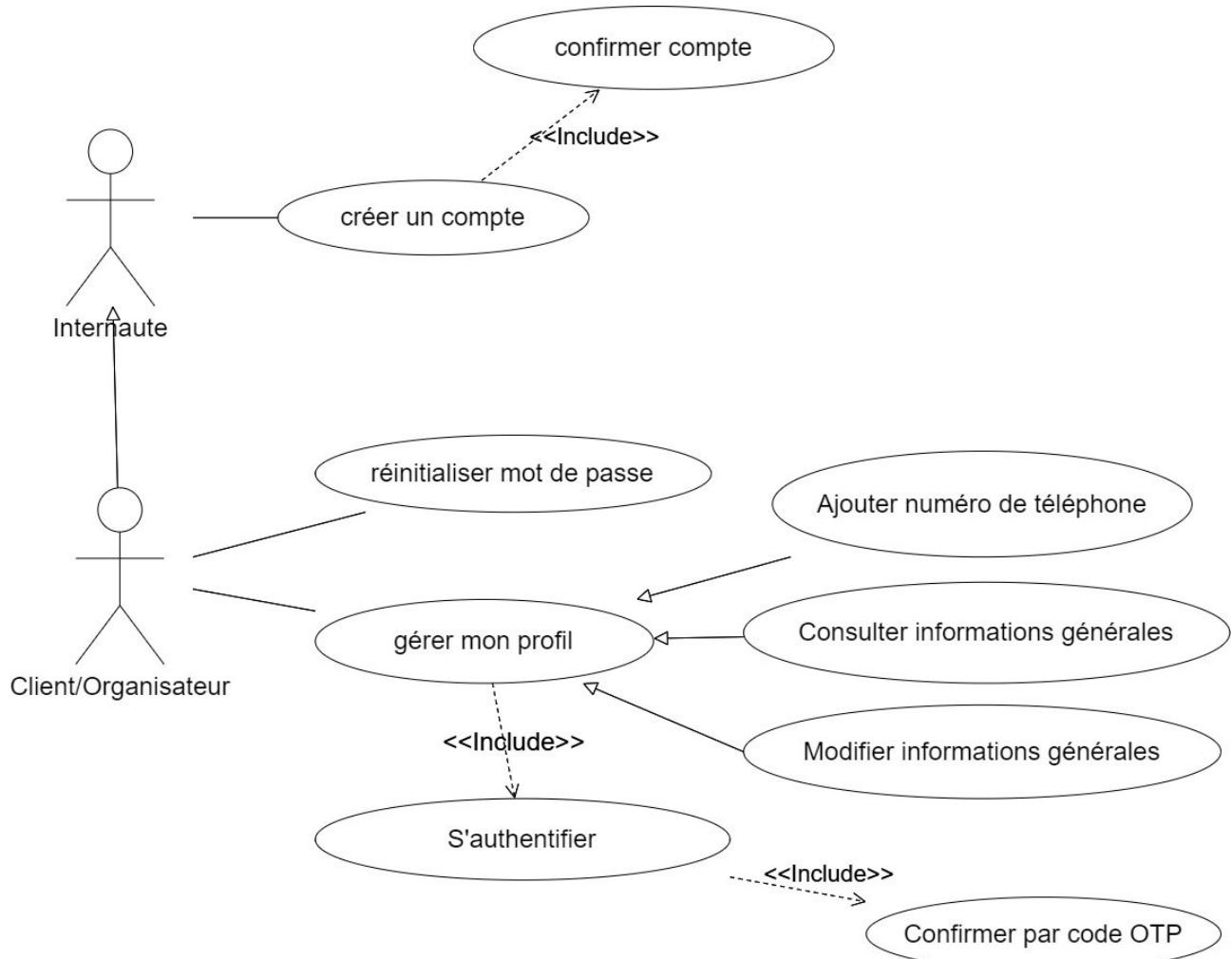


FIGURE 3.1 : Diagramme de cas d'utilisation du Sprint 1

3.2.4.2 Diagramme de classe du Sprint 1

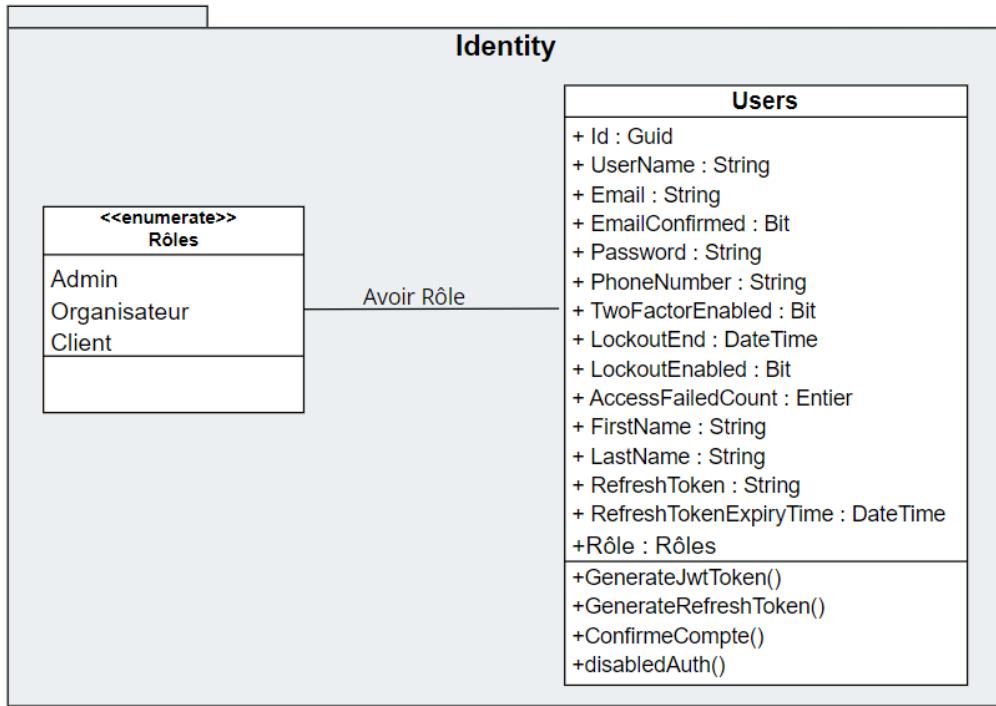


FIGURE 3.2 : Diagramme de classe du Sprint 1

3.2.4.3 Diagramme d'activité pour l'authentification

Ce diagramme illustré par la figure 3.3 présente le processus de connexion à notre plateforme. L'utilisateur commence par tenter de se connecter, puis est dirigé soit vers l'authentification s'il possède déjà un compte, soit vers la création d'un compte s'il n'en a pas.

Lors de la création de compte, le système envoie un email de confirmation à l'utilisateur pour valider son inscription. Une fois l'email confirmé, le compte est activé, et l'utilisateur peut procéder à l'authentification.

Lors de l'authentification, le système vérifie la validité des informations fournies, permettant jusqu'à cinq tentatives avant de bloquer l'accès. Si les informations sont valides, l'utilisateur reçoit un code OTP par email qu'il doit entrer pour valider son authentification. Si le code OTP est incorrect, l'utilisateur est redirigé vers l'étape d'authentification. En cas de succès, l'authentification est finalisée.

Une fois authentifié avec succès, le système détermine le rôle de l'utilisateur : simple utilisateur, administrateur ou organisateur, et le dirige vers l'interface correspondante.

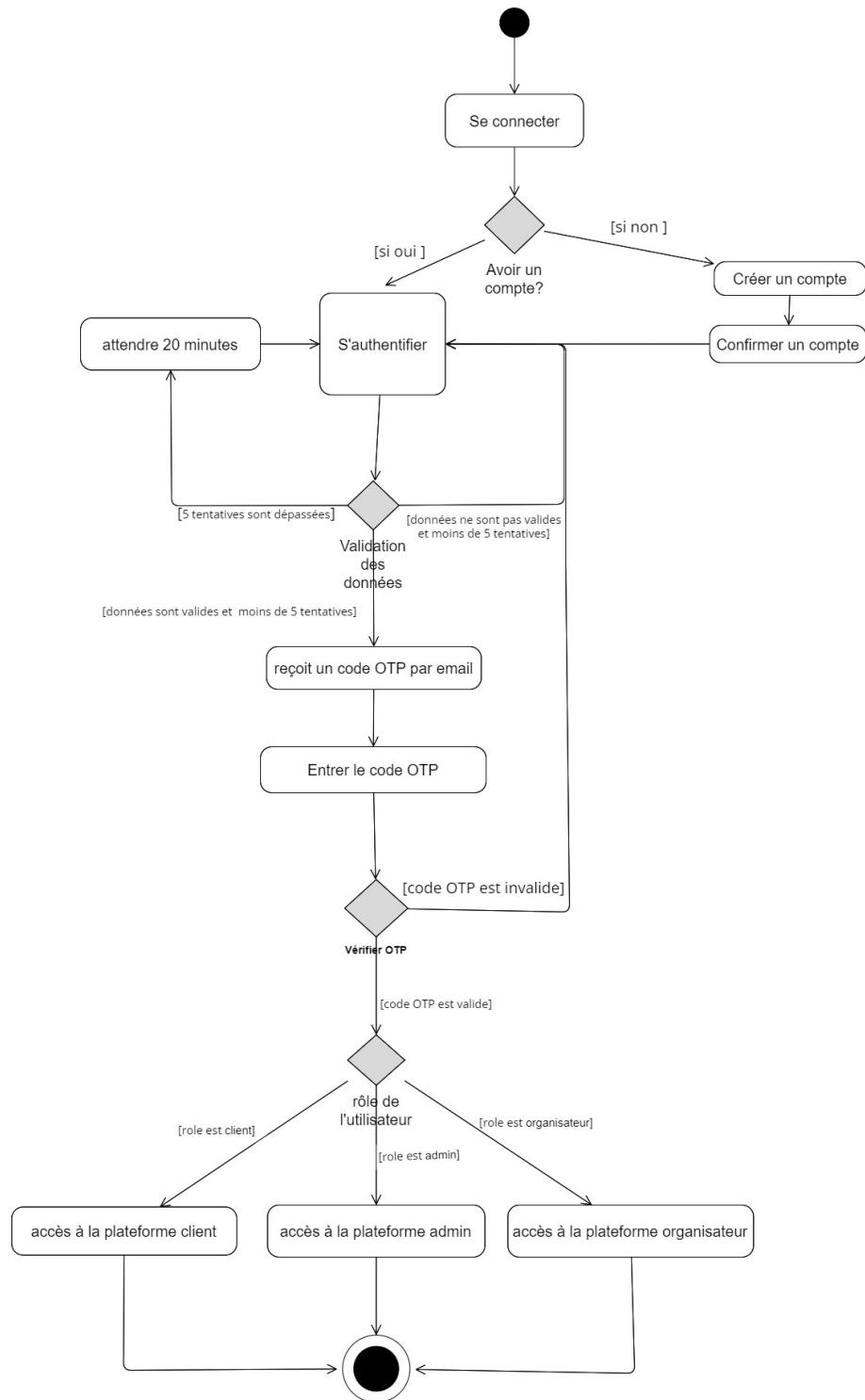


FIGURE 3.3 : Diagramme d'activité pour l'authentification

3.2.4.4 Diagramme de séquence pour l'accès aux données avec le jeton

Le diagramme de séquence illustré par la figure 3.3 représente le processus d'accès aux données avec le jeton.

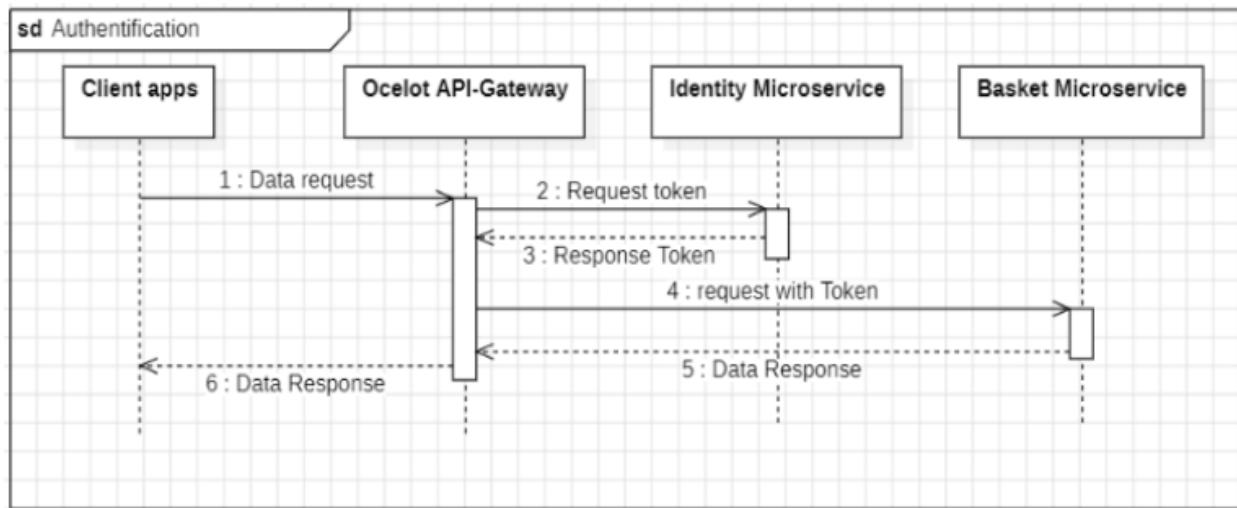


FIGURE 3.4 : Diagramme de séquence pour l'accès aux données avec le jeton

Le processus se déroule comme suit : D'abord l'application cliente envoie une requête demande de ressource du backend à travers une API qui doit contenir dans le header le jeton d'accès. La passerelle API Ocelot vérifie l'autorisation et l'authentification depuis le jeton d'accès et le micro-service Identity qui vérifie la validité du jeton. Si tout est valide, Ocelot fait le mappage des services et demande la ressource du service appelé.

3.2.5 Réalisation

Pour mieux comprendre le fonctionnement de notre projet, nous allons présenter les différents fonctions de l'application "SofiaTicket" en se basant sur un scénario.

Dans ce contexte pour s'inscrire à l'application un utilisateur doit remplir le formulaire d'inscription. La figure 3.4 nous montre le formulaire d'inscription pour l'Organisateur et pour le client.

The screenshot shows a 'Sign up' form with five input fields:

- First Name:** An input field with a person icon and placeholder text.
- Last Name:** An input field with a person icon and placeholder text.
- Email:** An input field with an envelope icon and placeholder text.
- Password:** An input field with a lock icon and placeholder text, accompanied by a visibility toggle icon.
- Confirm Password:** An input field with a lock icon and placeholder text, accompanied by a visibility toggle icon.

[Signup as user](#)

[Signup as organizer](#)

[Already have an account? Log In](#)

FIGURE 3.5 : interface de création de compte

Après l'inscription, un e-mail de confirmation sera envoyé à l'adresse fournie par l'utilisateur pour valider son adresse. L'envoi de cet e-mail est simulé à l'aide de Mail Catcher, comme illustré dans la figure 3.5 .

De plus, après l'inscription, si l'utilisateur est un client, un panier sera créé au sein du service BasketAPI à l'aide de RabbitMQ. Dans le cas où l'utilisateur est un organisateur, un nouvel organisateur sera créé dans le service CatalogItemAPI.

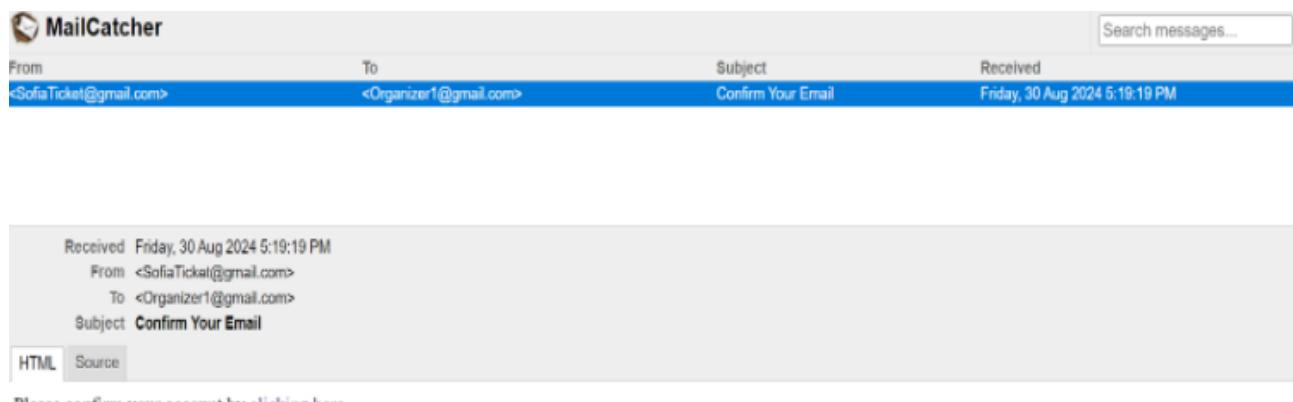
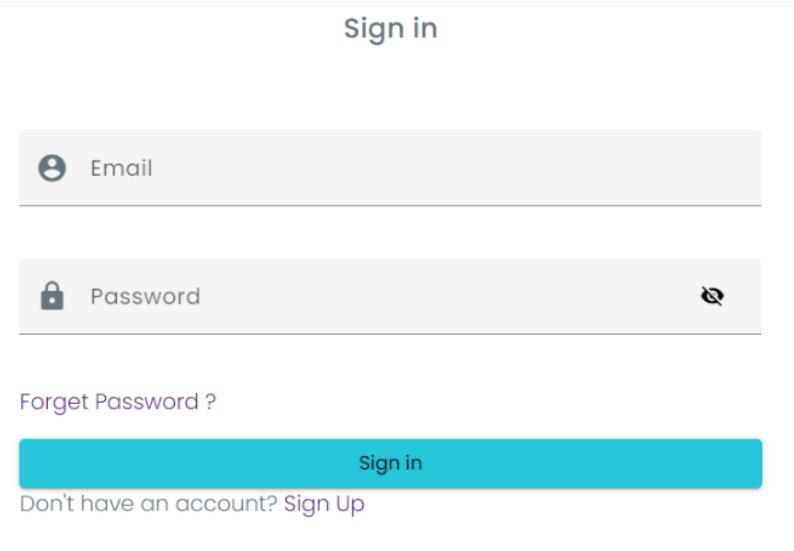
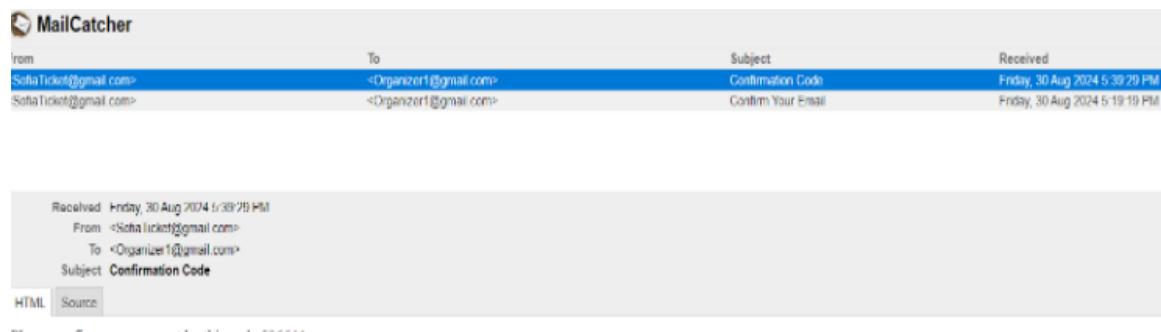
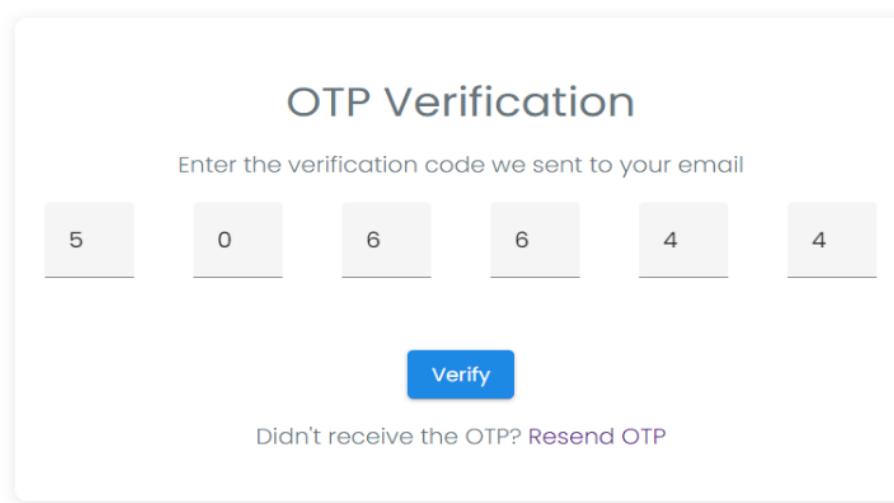


FIGURE 3.6 : Interface mail Catcher

Après avoir confirmé votre email, vous serez redirigé vers l'interface de Sign in.

**FIGURE 3.7 :** Interface "Sign In"

Après avoir validé les données de connexion, un code OTP à 6 chiffres sera envoyé à l'adresse e-mail de l'utilisateur. Celui-ci doit entrer ce code pour finaliser la connexion.

**FIGURE 3.8 :** Interface OTP par Email**FIGURE 3.9 :** Interface "OTP Verification"

Si l'utilisateur aurait oublié son mot de passe, il peut cliquer sur "Forget Password ? ". Ensuite, un e-mail va être envoyé à l'adresse e-mail qu'il a fourni. Ce message contiendra les instructions nécessaires pour définir un nouveau mot de passe afin qu'il puisse se connecter à nouveau.

Forgot your password?

Please enter the email you use to sign in .

Your email*

neyssen@gmail.com

Request password reset

Back to Sign in

FIGURE 3.10 : Interface pour saisir l'adresse électronique

Après avoir demandé la réinitialisation du mot de passe, un e-mail de réinitialisation sera envoyé à l'adresse fournie.

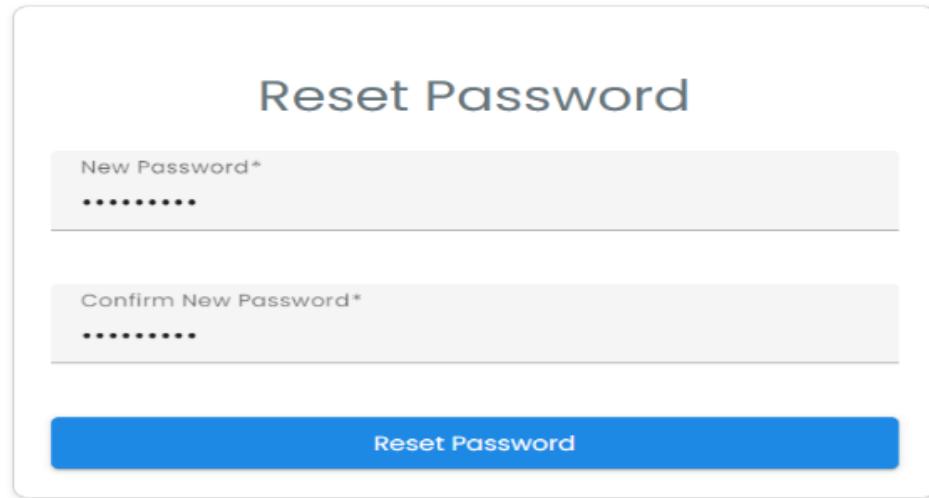
Received Friday, 30 Aug 2024 5:52:35 PM
From <SofiaTicket@gmail.com>
To <neyssen@gmail.com>
Subject Reset your Password

HTML Source

Follow the instruction to reset your Password

to reset your password [clicking here](#)

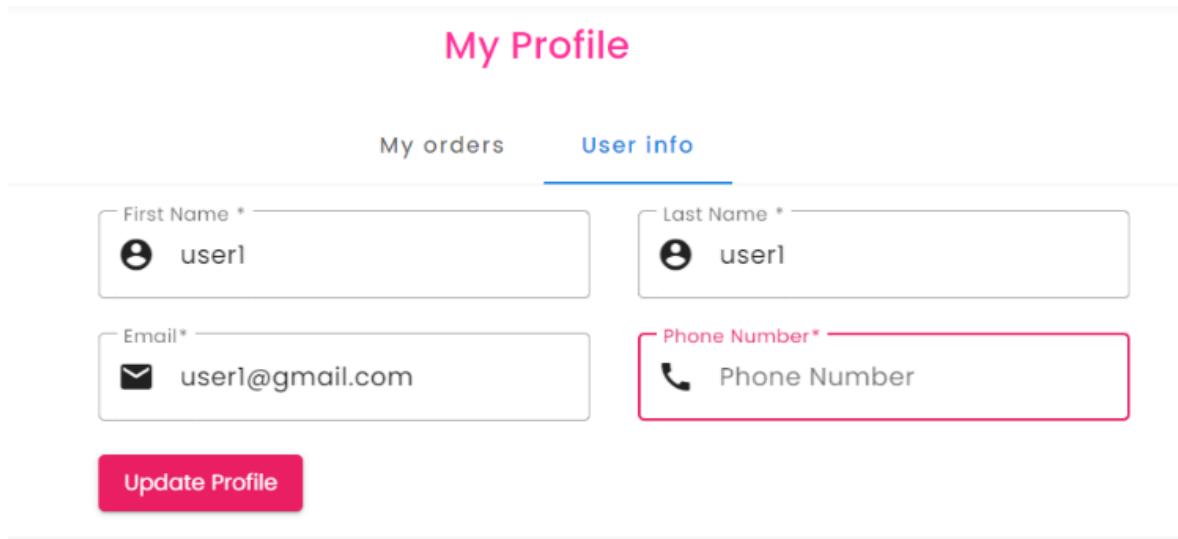
FIGURE 3.11 : Interface E-mail de réinitialisation



The image shows a 'Reset Password' interface. It features two input fields: 'New Password*' and 'Confirm New Password*', both containing placeholder text '.....'. Below these fields is a blue button labeled 'Reset Password'.

FIGURE 3.12 : Interface Reset Password

Après vous être connecté, vous pourrez gérer vos informations, ajouter un numéro de téléphone ou modifier vos informations de profil.



The image shows a 'My Profile' interface. It includes tabs for 'My orders' and 'User info', with 'User info' being the active tab. Under 'User info', there are four input fields: 'First Name *' (placeholder 'user1'), 'Last Name *' (placeholder 'user1'), 'Email *' (placeholder 'user1@gmail.com'), and 'Phone Number*' (placeholder 'Phone Number'). A pink button labeled 'Update Profile' is located at the bottom left.

FIGURE 3.13 : Interface "My Profil"

3.3 Sprint 2 : Gestion des types du billet et des catégories

Dans cette section nous allons présenter les différentes étapes permettant la réalisation du Sprint "Gestion des types du billet et des catégories".

3.3.1 Objectifs du Sprint 2

L'objectif de ce Sprint est de permettre à l'administrateur de gérer les types de billets ainsi que les catégories d'événements, y compris l'ajout, la modification et la suppression des différents types

et catégories. Cette fonctionnalité vise à offrir une interface intuitive pour faciliter l'administration des divers types de billets et des catégories disponibles dans l'application, améliorant ainsi la gestion des événements et l'expérience utilisateur.

3.3.2 Backlog du Sprint 2

Id	Fonctionnalités (User stories)	Priorité	Estimation (Jour)
1	En tant qu'Administrateur, je peux afficher la liste des types de billets.	1	3
2	En tant qu'Administrateur, je peux rechercher un type de billet.	3	2
3	En tant qu'Administrateur, je peux ajouter un type de billet.	1	2
4	En tant qu'Administrateur de l'application, je peux supprimer un type de billet.	2	1
5	En tant qu'Administrateur, je peux afficher la liste des catégories d'événements.	1	2
6	En tant qu'Administrateur, je peux rechercher une catégorie d'événement.	3	2
7	En tant qu'Administrateur, je peux ajouter une catégorie d'événement.	1	2
8	En tant qu'Administrateur, je peux supprimer une catégorie d'événement.	2	1

TABLEAU 3.3 : Backlog du Sprint 2

3.3.3 Conception

3.3.3.1 Diagramme de cas d'utilisation du Sprint "Gestion des types du billet et des catégories"

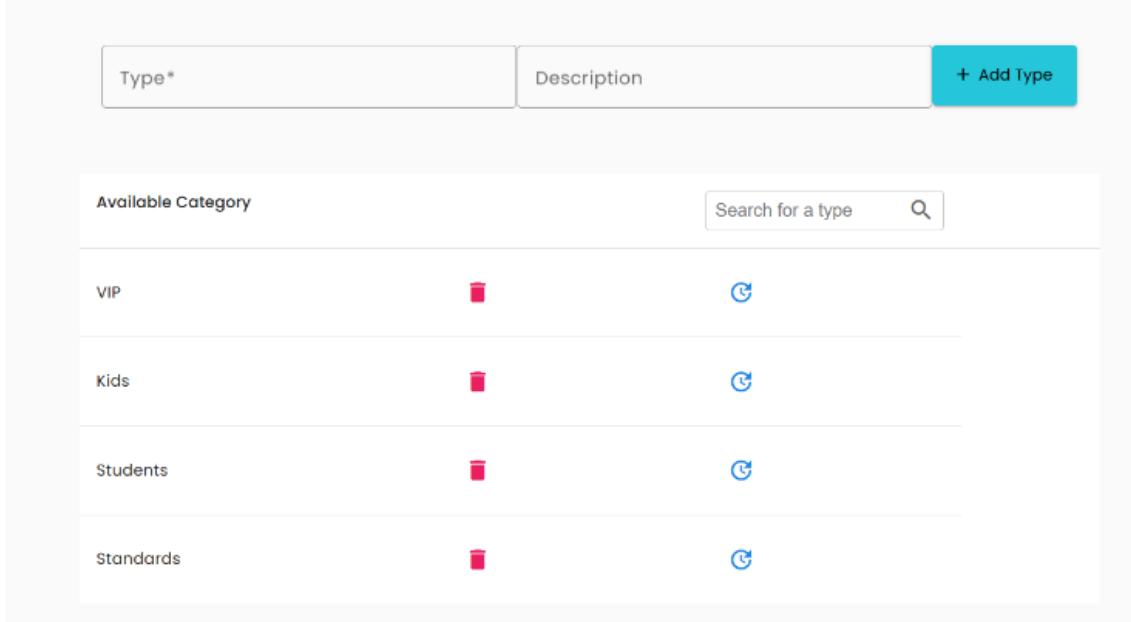
La figure 3.14 représente le diagramme cas d'utilisation du Sprint "Gestion des types du billet et des catégories".



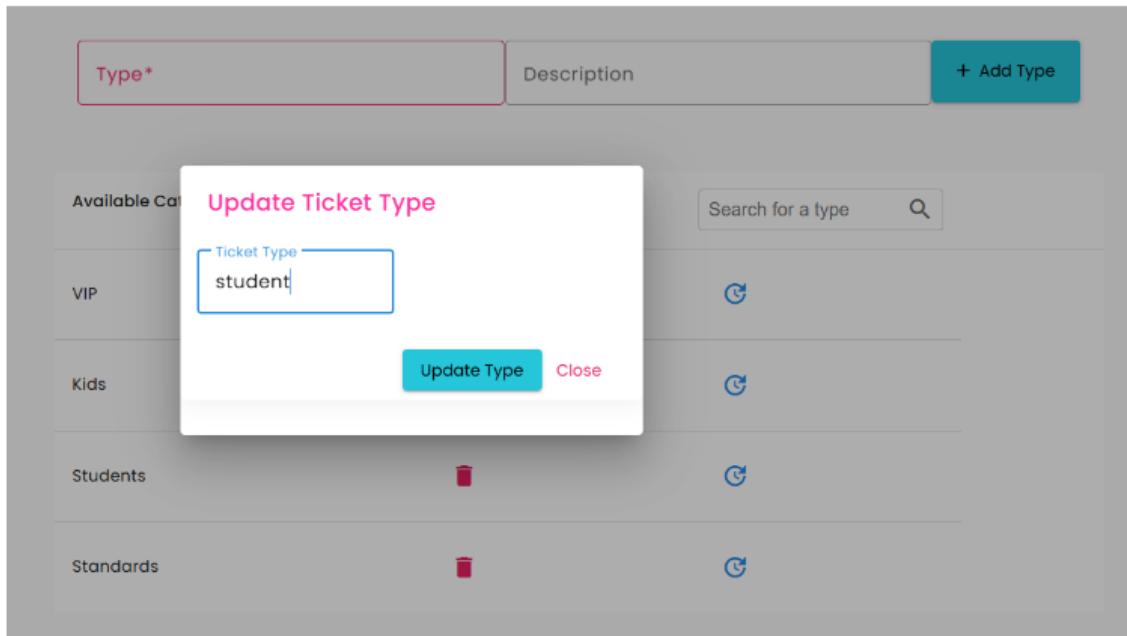
FIGURE 3.14 : Diagramme de cas d'utilisation de gestion des types de billets et des catégories

3.3.4 Réalisation

La figure 3.15 illustre l'interface permettant à l'administrateur de consulter la liste des types de billets, ainsi que d'ajouter, de rechercher, de modifier ou de supprimer un type.

**FIGURE 3.15 :** Interface liste des types de billets

La figure 3.16 représente l'interface permettant à l'administrateur de modifier un type de billet.

**FIGURE 3.16 :** Interface pour modifier un type de billet

La figure 3.17 illustre l'interface permettant à l'administrateur de consulter la liste des catégories d'événements, ainsi que d'ajouter, de rechercher, de modifier ou de supprimer une catégorie.

Category		
Available Category		
Theatre		
Cirque		
SPECTACLE		
Concert		

FIGURE 3.17 : Interface liste des catégories d'événements

La figure 3.18 représente l'interface permettant à l'administrateur de modifier une catégorie.

Update Category

Category Name
Theatre

Update Category

Close

FIGURE 3.18 : Interface Modifier une catégorie d'événement

Conclusion

Au cours de ce chapitre, nous avons présenté la réalisation de la première Release.

Pour ce faire, nous avons passé par l'analyse, la conception et la réalisation des trois premiers Sprints, nous passons maintenant à la réalisation du deuxième Release".

RELEASE 2 : GESTION DES ÉVÉNEMENTS, SESSIONS ET STOCKS

Plan

Introduction	58
1 Sprint 3 : Gestion des événements et des sessions	58
2 Conception	59
3 Sprint 4 : Gestion des billets et du stock	66
4 Conclusion	69

introduction

Dans ce chapitre, nous allons aborder les étapes de réalisation du troisième Sprint, intitulé "Gestion des événements et des sessions," et du quatrième Sprint, "Gestion des billets et du stock." Ces Sprints visent à optimiser les fonctionnalités de notre application, en améliorant la gestion des événements et des billets.

4.1 Sprint 3 : Gestion des événements et des sessions

Dans cette section nous allons présenter les différentes étapes de la réalisation du Sprint 3 "Gestion des événements et des sessions "

4.1.1 Objectifs du Sprint 3

L'objectif du troisième Sprint est de développer le module « Gestion des événements et des sessions » qui permet la gestion efficace des événements.

4.1.2 Backlog du Sprint 3

Id	Fonctionnalités (User stories)	Priorité	Estimation (Jour)
1	En tant qu'un organisateur, je peux afficher la liste de mes événements.	1	2
2	En tant qu'un organisateur, je peux rechercher un événement par son nom et filtrer par catégorie.	3	2
3	En tant qu'un organisateur, je peux ajouter un événement.	1	1
4	En tant qu'un organisateur, je peux modifier un événement.	1	2
5	En tant qu'un organisateur, je peux supprimer un événement.	2	1
6	En tant qu'un organisateur, je peux Visualiser les détails d'un événement.	2	1
7	En tant qu'un organisateur, je peux visualiser la liste des sessions pour un événement.	1	2
8	En tant qu'organisateur, je peux ajouter une session.	1	1
9	En tant qu'organisateur, je peux modifier une session.	1	1
10	En tant qu'organisateur, je peux supprimer une session.	2	1

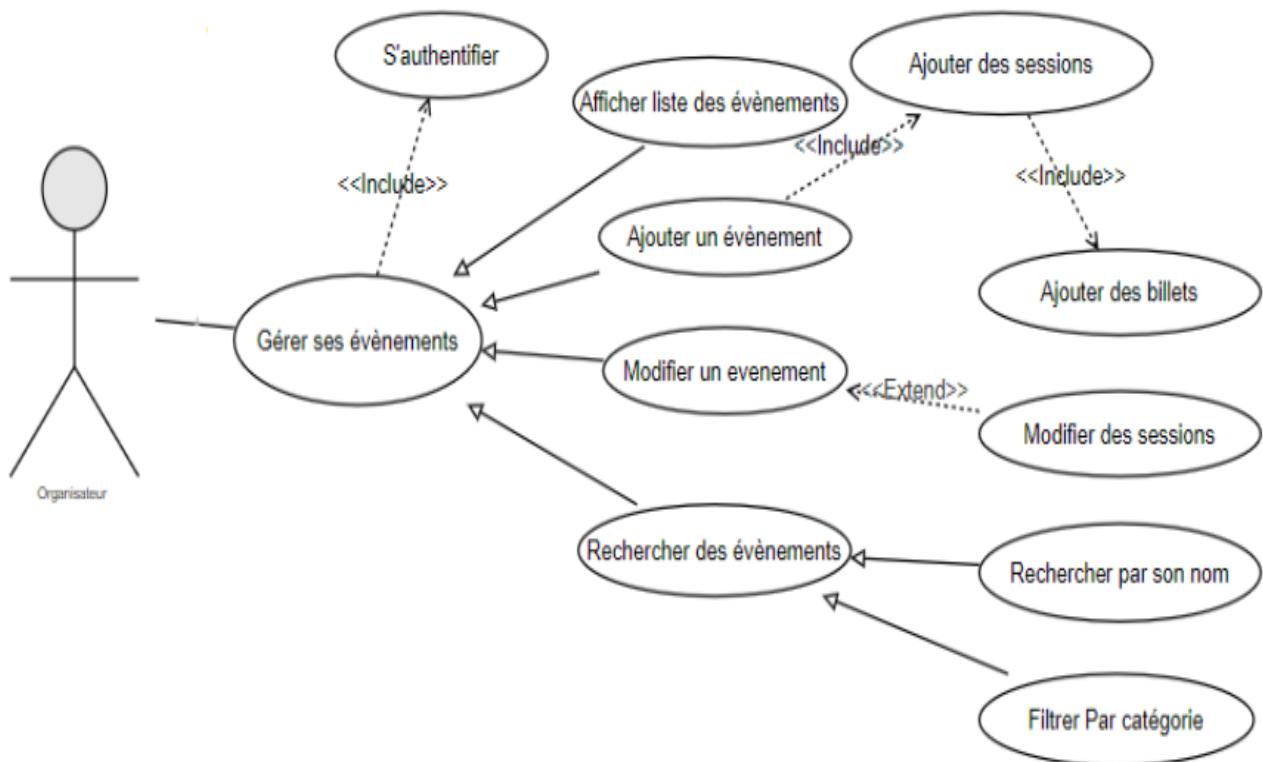
11	En tant qu'organisateur, je peux filtrer les sessions par local et temps.	3	2
----	---	---	---

TABLEAU 4.1 : Backlog du Sprint 3

4.2 Conception

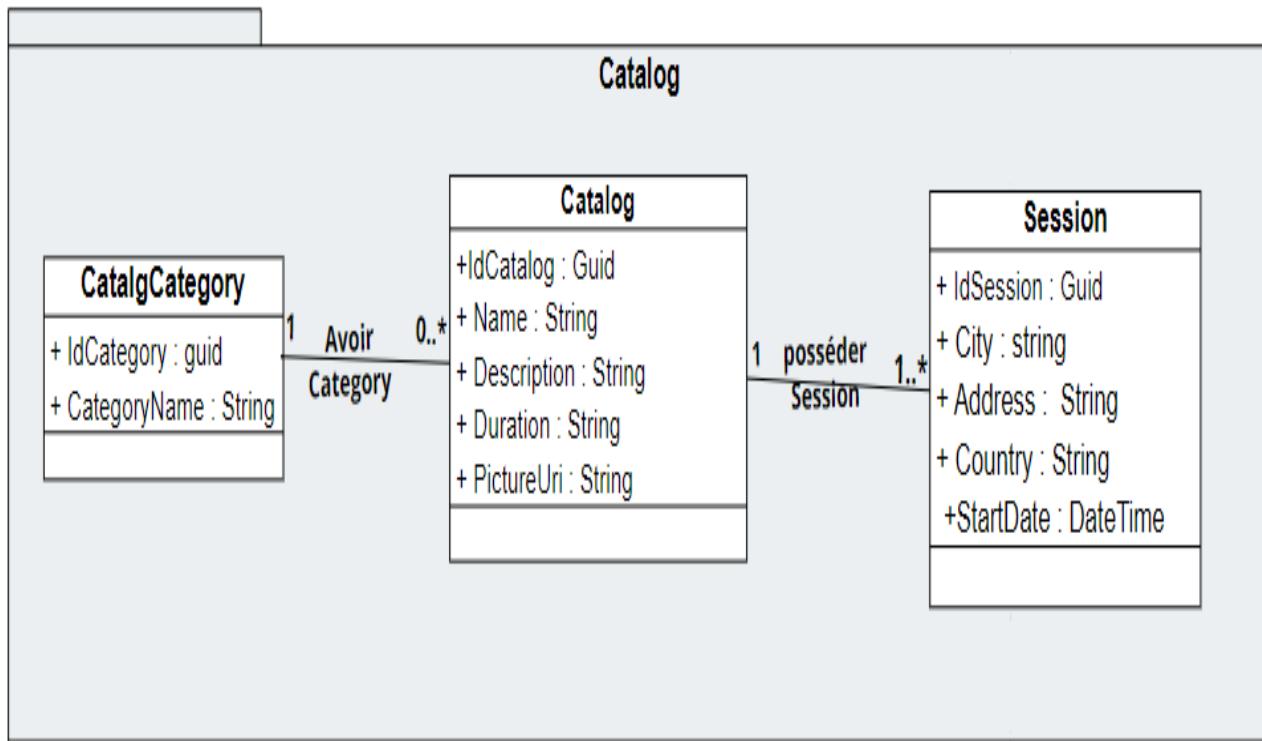
4.2.1 Spécification des besoins fonctionnels

La figure 4.1 représente le diagramme cas d'utilisation du Sprint "Gestion des événements et des session".

**FIGURE 4.1** : Diagramme de cas d'utilisation "Gestion des événements et des sessions"

4.2.2 Diagramme de classe de sprint 3

La figure 4.2 représente le diagramme de classe de Sprint "Gestion des événements et des session".

**FIGURE 4.2 :** Diagramme de classe de sprint 3

4.2.3 Réalisation

Interface affichage de Liste des événements

La figure 4.3 illustre une interface permettant d'afficher la liste des événements avec pagination.

L'utilisation de la pagination permet de diviser le contenu en sections plus gérables, améliorant ainsi la performance de l'application et offrant une meilleure expérience utilisateur en évitant de surcharger l'interface avec trop d'informations à la fois.

Les organisateurs peuvent rechercher un événement par son nom, filtrer par catégorie, ainsi que modifier ou supprimer un événement.

FIGURE 4.3 : Interface Liste des évènements

La figure 4.4 permet de filtrer les événements par catégorie.

FIGURE 4.4 : Interface de filtrage par categorie

4.2.4 Interfaces Ajouter d'un évènement

La création d'un événement se compose de trois étapes principales :

- **Saisie des données générales de l'événement** : Cette première étape consiste à entrer les informations de base concernant l'événement, telles que le nom, la description, la date, l'heure, la durée, et la catégorie de l'événement.
- **Ajout des sessions et leurs billets** : Une fois les données générales enregistrées, la deuxième étape implique l'ajout des sessions pour l'événement. Pour chaque session, vous devez définir les

détails comme la date et l'heure spécifiques, ainsi que de définir les types de billets (par exemple, VIP, standard) et leurs stocks disponibles.

- **Confirmation** : Enfin, la dernière étape consiste à revoir toutes les informations saisies, à s'assurer que tous les détails sont corrects, puis à confirmer la création de l'événement. Cette confirmation finalise le processus et rend l'événement accessible aux utilisateurs.

La figure 4.5 représente l'interface Saisie des données générales de l'événement.

FIGURE 4.5 : Interface Saisie des données générales de l'événement

la figure 4.6 représente l'interface d'ajout des sessions et leurs billets disponibles.

Fill this form

Event Description Sessions Done (3)

Event Address*
metline

City*
bizerte

country*
tunisia

Starting Date*
20/10/2024 10:00

Select Ticket Type
VIP

Price*
10

Devise*
EUR

ticket quantity*
100

FIGURE 4.6 : Interface d'ajout des sessions et leurs billets

La figure 4.7 représente l'interface de Confirmation pour l'ajout d'évènement.

Home About Contact Us Infos Soon

Create New Event

Fill this form

Event Description Sessions Done

You are now done.

FIGURE 4.7 : Interface de Confirmation pour l'Ajout d'Événement

Interfaces de modification d'un évènement

L'interface illustrée par la figure 4.8 montre le formulaire déjà rempli par les données que l'organisateur a créées ou il peut modifier n'importe quel champ à propos les informations liées à l'événement : nom, description, durée, image associée,

The screenshot shows the 'Update Event' form on a web application. At the top, there's a navigation bar with links for Home, About, Contact Us, and Help. On the left, there are two buttons: 'DashBoard' and 'Manage CatalogItem'. The main form has a title 'Update Event' and a sub-section 'Event Description'. It contains fields for 'Event Name*' (set to 'Exo Planet #5 - Exploration'), 'Event Description*' (a detailed text about the concert tour), 'Category' (set to 'Concert'), and 'Duration*' (set to '100'). Below these is a 'Select Image' section with a preview image of a concert stage. At the bottom are two buttons: a teal 'Update' button and a blue 'Cancel' button.

FIGURE 4.8 : Interface de modification d'un évènement

L'organisateur peut même modifier les détails des sessions disponibles ou il peut modifier la date et la localisation de chacune comme illustré par la figure 4.9.

The screenshot shows the 'Update Session' form. At the top, it says 'Update Event' and 'Update this form'. There are two sections, each with an 'Event Description' checkbox and a 'Sessions' icon. The first section has fields for 'Event Address*' (ras jebal), 'City*' (bizerte), and 'country*' (tunisia). The second section has fields for 'Event Address*' (el jem), 'City*' (tunis), and 'country*' (tunisie). Both sections have a 'Starting Date*' field (10/11/2024 20:00 and 15/09/2024 19:00 respectively) and a teal 'Update Session' button at the bottom.

FIGURE 4.9 : Inteface de modification des sessions

Interface Details d'un évènement et leur List des sessions

La Figure 4.10 représente l'interface permet aux organisateurs de consulter les informations détaillées d'un événement et de gérer efficacement les sessions associées.

The screenshot shows the SofiaTicket platform. On the left sidebar, there are links for 'Dashboard' and 'Manage CatalogItem'. The main content area displays an event titled 'Exo Planet #5 – Exploration'. It includes a thumbnail image of the event, a category ('Concert'), duration ('180 Minute(s)'), and a detailed description about the tour. To the right, a section titled 'Available Sessions' lists sessions based on location, date, and time. Each session entry includes a 'Manage Tickets' button. A blue button at the bottom right of the session list says '+ Session'.

FIGURE 4.10 : Interface Details d'un évènement et leur sessions

L'interface illustre, comme montré dans la Figure 4.11, la possibilité de filtrer les sessions selon plusieurs critères. L'organisateur peut sélectionner les sessions en fonction de la localisation, ainsi que définir une période spécifique en indiquant une date de début et une date de fin. Ces options de filtrage permettent à l'organisateur de trouver facilement les sessions correspondant aux besoins précis.

This screenshot shows the 'Available Sessions' filter interface. It features a search bar for 'Location' and filters for 'FROM' and 'TO' dates. Below these filters is a table listing sessions by address, city, and country. The table entries are: 'gangnam Seoul korea', 'pathe beijing China', and 'ariana tunis tunisia'. At the bottom left is a blue 'Add session' button. To the right of the table is a calendar for September 2024, with the 2nd highlighted. To the right of the calendar, there are three 'Manage Tickets' buttons, each corresponding to one of the listed sessions.

FIGURE 4.11 : Interface Filtrage des sessions

Interface ajouter une session

La Figure 4.11 montre l'interface "Ajouter une session". Elle permet aux organisateurs d'entrer les informations nécessaires pour créer une nouvelle session.

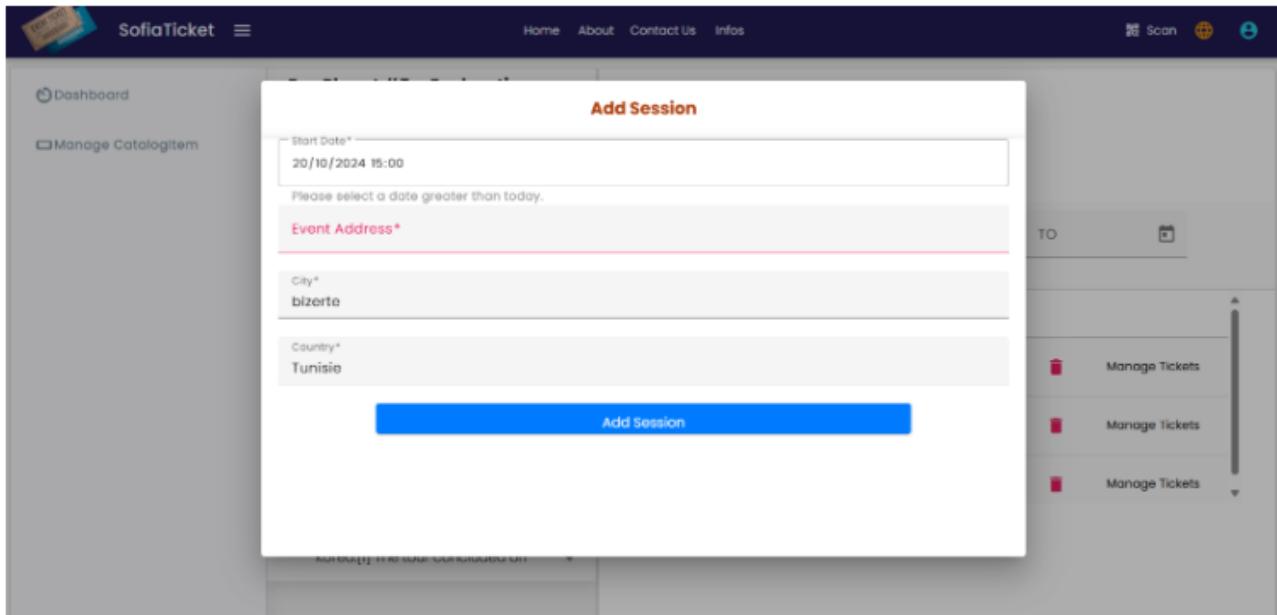


FIGURE 4.12 : Interface "Ajouter une session"

4.3 Sprint 4 : Gestion des billets et du stock

Dans cette section nous allons présenter les différentes étapes de la réalisation du Sprint 4 : "Gestion des billets et du stock"

4.3.1 Objectifs du Sprint 4

La gestion des billets et du stock est un processus clé qui assure le suivi et le contrôle de l'inventaire des billets disponibles, y compris l'ajout de différents types de billets tels que VIP, standard, ou à tarif réduit. Cette gestion permet d'optimiser leur distribution, d'éviter les ruptures de stock, et de garantir une expérience fluide pour les utilisateurs tout en maximisant les ventes et la disponibilité des billets.

4.3.2 Backlog du Sprint 4

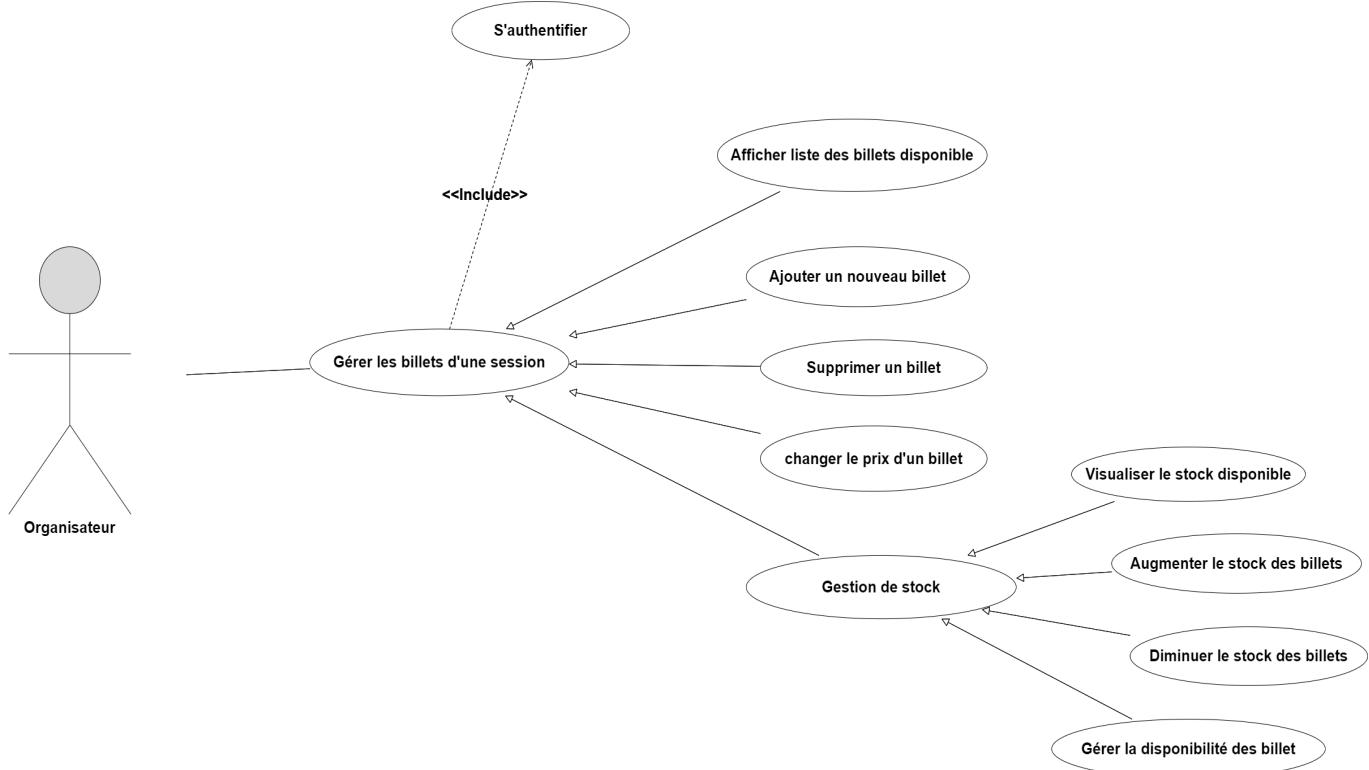
Id	Fonctionnalités (User stories)	Priorité	Estimation (Jour)
1	En tant qu'un organisateur, je peux afficher la liste des billets et leur stock pour une session.	1	2
2	En tant qu'un organisateur, je peux ajouter un nouveau billet.	1	2

3	En tant qu'un organisateur, je peux modifier le prix d'un billet.	2	2
4	En tant qu'un organisateur, je peux supprimer un billet.	3	2
5	En tant qu'un organisateur, je peux visualiser les billets disponibles en stock.	4	2
6	En tant qu'un organisateur, je peux augmenter le nombre de billets.	3	2
7	En tant qu'un organisateur, je peux diminuer le nombre de billets.	3	2
8	En tant qu'un organisateur, je peux gérer la disponibilité du stock.	2	1

TABLEAU 4.2 : Backlog du Sprint 4

4.3.3 Spécification des besoins fonctionnels

La figure 4.13 représente le diagramme cas d'utilisation du Sprint 4 "Gestion des billets et du stock".

**FIGURE 4.13 :** diagramme cas d'utilisation du Sprint 4

4.3.4 Diagramme de classes de sprint 4

La Figure 4.14 représente le diagramme de classe de Sprint 4 "Gestion des billets et du stock".

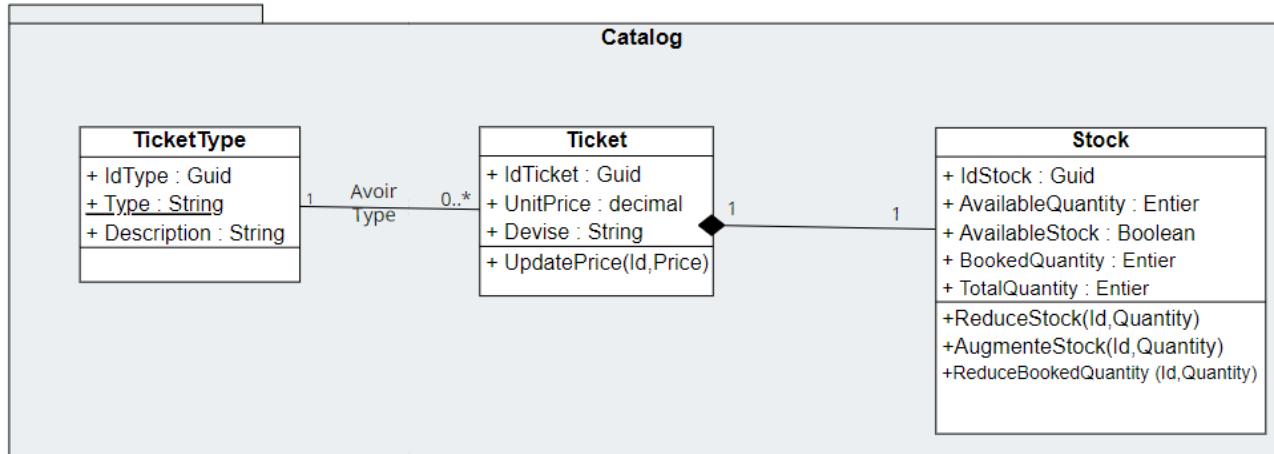


FIGURE 4.14 : Diagramme de classe de sprint 4

4.3.5 Réalisation

l'interface Liste des billets

La Figure 4.15 montre l'interface qui permet la gestion des billets pour une session bien déterminée où l'organisateur peut ajouter un billet en précisant son type, la quantité proposée et le prix. Comme il peut supprimer ou modifier un billet.

The screenshot shows the SofiaTicket application interface. The top navigation bar includes links for Home, About, Contact Us, and Infos. On the left, there are links for Dashboard and Manage CatalogItem. The main content area displays a table of available tickets for an event in El Jem, Tunisia, on September 15, 2024, at 07:00 PM. The table columns are **TicketType**, **Price**, **Quantity**, and **Stock Is Available**. Each row includes edit and delete icons. The table data is as follows:

TicketType	Price	Quantity	Stock Is Available	Edit	Delete
VIP	50 TND	120 Tickets	false		
Kids	10 TND	110 Tickets	true		
Students	15 TND	200 Tickets	true		
VIP	20 TND	100 Tickets	true		
VIP	1 TND	1000 Tickets	true		

FIGURE 4.15 : Interface Liste des billets

L'interface Mettre à jour les billets et le stock

L'interface illustrée par la Figure 4.16 montre la page de modification où l'organisateur peut mettre à jour les billets en ajustant le prix, en ajoutant ou en diminuant la quantité du stock, et en gérant la disponibilité des billets.

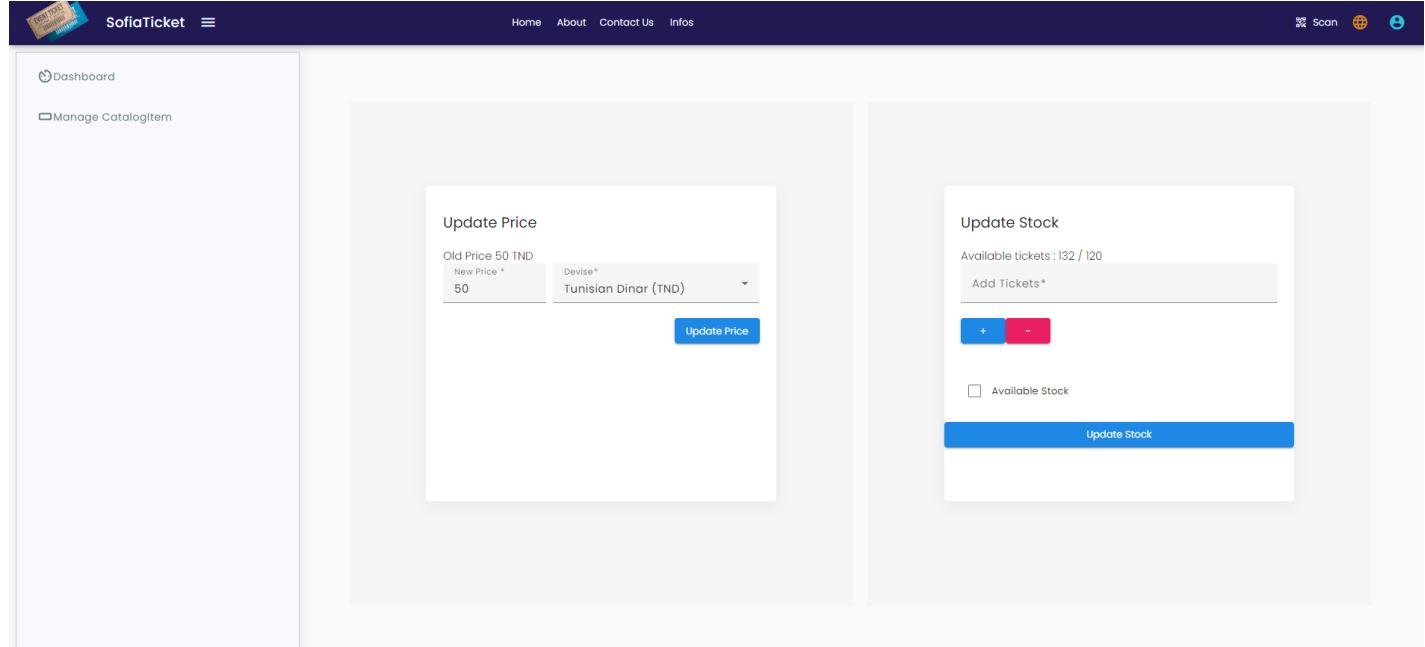


FIGURE 4.16 : interface Mettre à jour les billets et le stock

4.4 Conclusion

La Release 2 a introduit des fonctionnalités essentielles pour la gestion complète des événements, des sessions, des billets et du stock. Elle permet une gestion efficace des événements et des billets, tout en offrant un suivi précis du stock. Ces ajouts assurent une administration optimisée et une meilleure expérience pour les organisateurs.

RELEASE 3 :CONSULTER LES ÉVÈNEMENTS,GESTION DE PANIER , DES RÉSERVATIONS ET DE PAIEMENT

Plan

1	Sprint 5 : Consulter les évènements et les sessions disponibles	71
2	Sprint 6 : Gestion panier	75
3	Sprint 7 : Gestion des réservations et de paiement	78

Introduction

La Release 3 introduit des fonctionnalités essentielles pour le client, notamment la consultation des événements, la gestion du panier, et la réalisation des réservations et paiements en ligne. Cette phase couvre les sprints 5 à 7, qui permettent aux utilisateurs de visualiser les événements disponibles, d'ajouter des billets à leur panier, puis de finaliser leur réservation via un processus de paiement sécurisé.

5.1 Sprint 5 : Consulter les évènements et les sessions disponibles

Dans cette section nous allons présenter les différentes étapes de la réalisation du premier Sprint 5 " Consulter les évènements et les sessions disponibles".

5.1.1 Objectifs du Sprint 5

Ce Sprint permet à l'utilisateur de consulter les événements et les sessions disponibles avec une interface attrayante et une performance optimisée. Cela facilite la réservation rapide et efficace des événements.

5.1.2 Backlog du Sprint 5

Id	Fonctionnalités (User stories)	Priorité	Estimation (Jour)
1	En tant qu'un utilisateur, je peux consulter les événements disponibles.	1	4
2	En tant qu'un utilisateur, je peux rechercher un événement spécifique ou filtrer par catégorie et paye.	2	2
3	En tant qu'un utilisateur, je peux voir les événements les mieux recommandés à venir.	3	2
4	En tant qu'un utilisateur, je peux voir les détails de l'événement.	1	3
5	En tant qu'utilisateur, je peux voir les sessions disponibles d'un événement.	1	2
6	En tant qu'utilisateur, je peux filtrer les sessions par lieu et par date.	2	2

TABLEAU 5.1 : Backlog de Sprint 5

5.1.3 Technologies utilisées

- **Ngx Infinite Scroll** : ngx infinite scroll est une bibliothèque Angular qui facilite l'implémentation du chargement infini dans les applications Web. Cette technique permet de charger automatiquement de nouveaux éléments lorsque l'utilisateur fait défiler la page vers le bas, éliminant ainsi la nécessité d'une pagination traditionnelle.

- **Swiper** : Swiper est une bibliothèque de diaporama moderne et flexible pour les sites Web et les applications mobiles. Elle permet de créer des galeries d'images, des sliders et des carrousels, tout en offrant des fonctionnalités avancées et une personnalisation poussée.

5.1.4 Spécification des besoins fonctionnels

La figure 4.1 représente le diagramme cas d'utilisation du Sprint 5 "Consulter les évènements et les sessions disponibles".

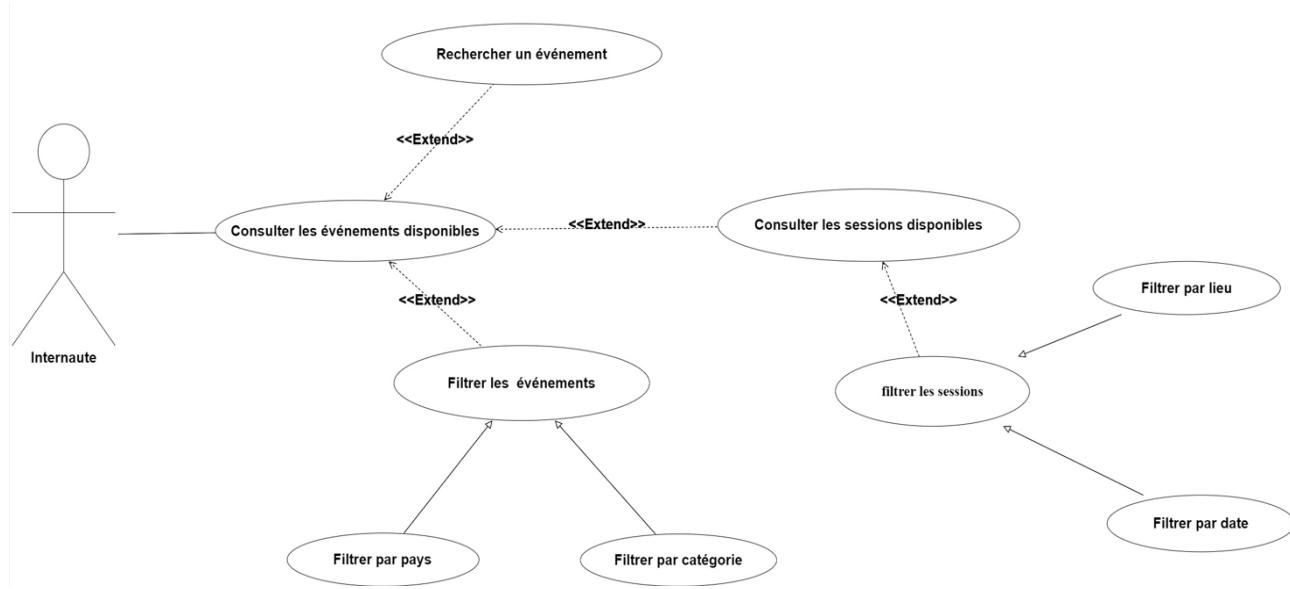


FIGURE 5.1 : diagramme cas d'utilisation du Sprint 5

5.1.5 Réalisation

Interfaces consulter les évènements disponibles

La figure 5.2 présente les événements regroupés par catégorie.

Chaque événement affiche des informations telles que le nom de l'organisateur, le prix minimum du billet, et la durée.

De plus, la fonctionnalité de défilement infini utilisant " ngx-infinite-scroll " permet de charger automatiquement de nouveaux événements à mesure que l'utilisateur fait défiler la page vers le bas.

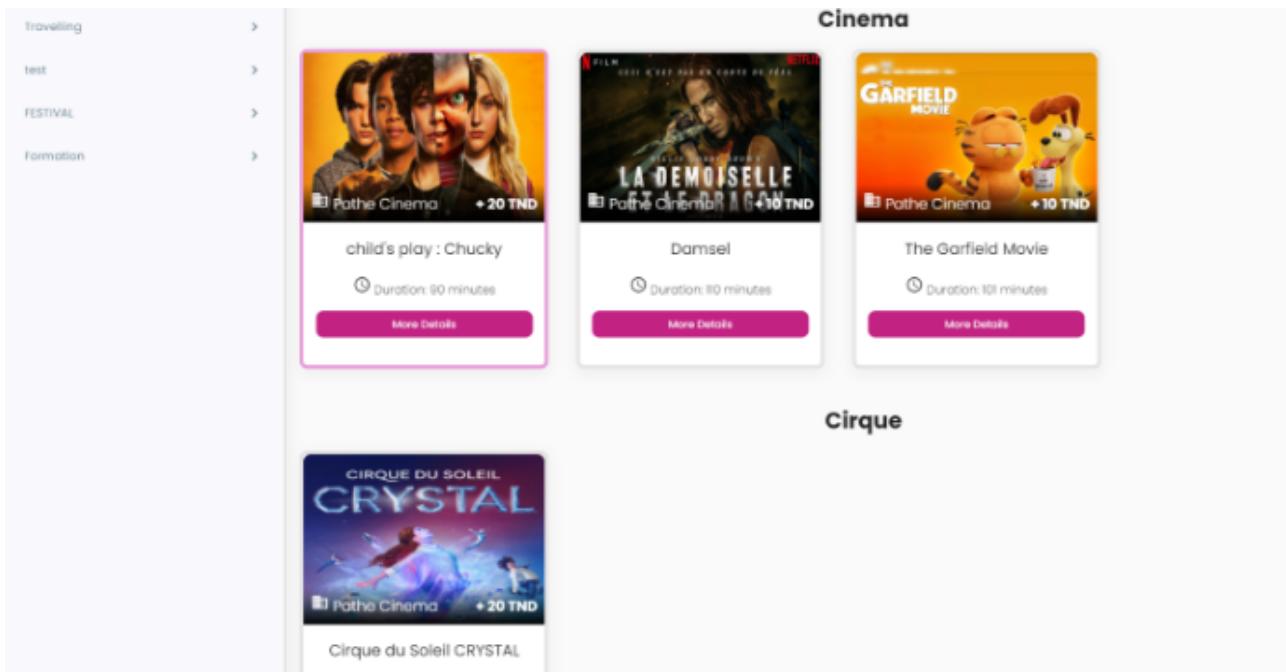


FIGURE 5.2 : Interface évènements disponibles

La figure 5.3 illustrée par le composant Swiper, est conçue pour mettre en avant les 5 événements à venir les mieux recommandés. Sur cette interface, Swiper crée un carrousel interactif qui permet aux utilisateurs de faire défiler horizontalement les événements. Chaque diapositive du carrousel présente un événement avec ses détails essentiels, comme le titre, la catégorie et le poster.

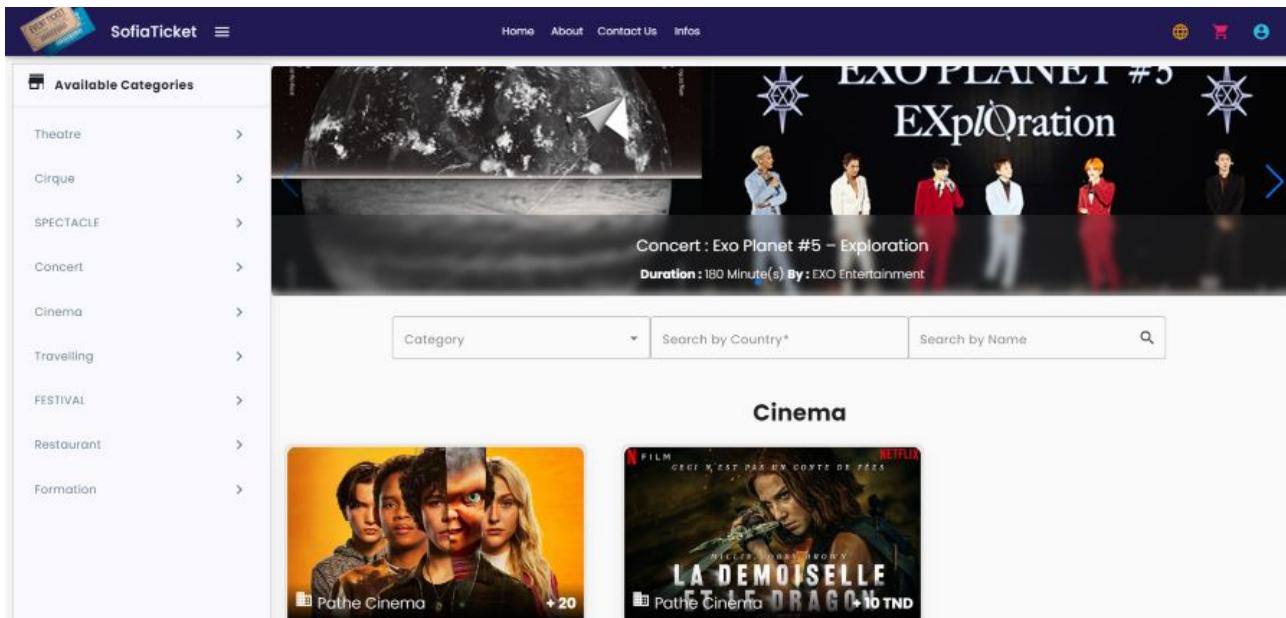


FIGURE 5.3 : Interface top 5 Évènements

L’interface, comme présenté dans la figure 5.4, permet aux utilisateurs de rechercher des événements en filtrant par catégorie ou paye spécifique ou en recherchant un événement particulier. Cette fonctionnalité facilite l’accès rapide aux événements.

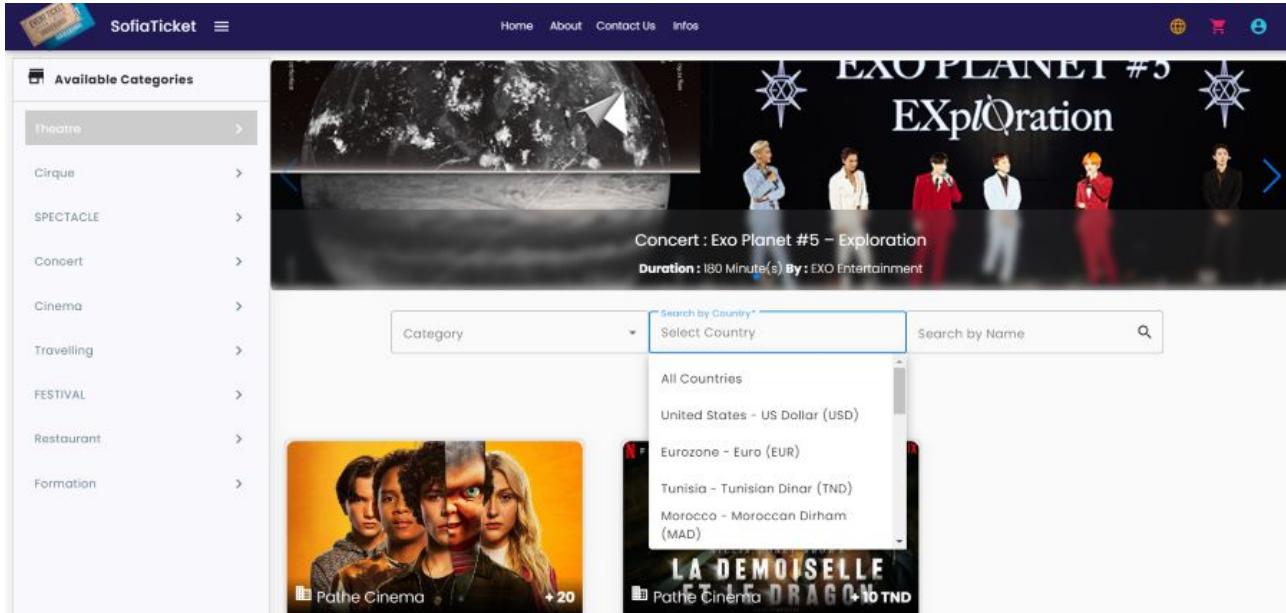


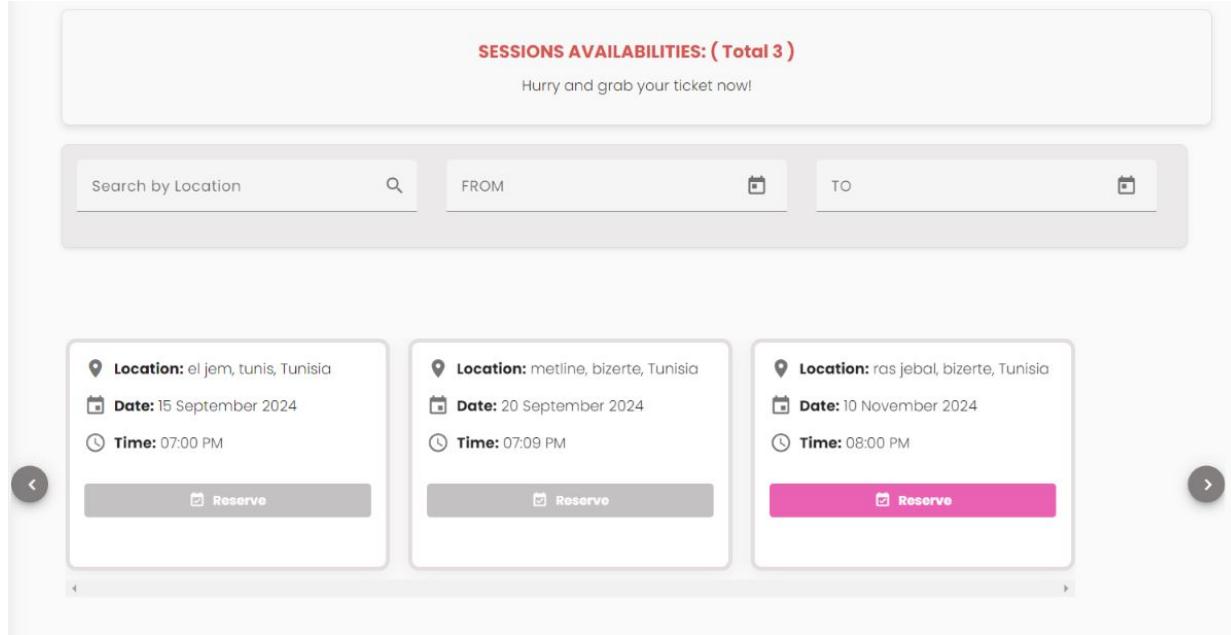
FIGURE 5.4 : Recherche par pays

Interface de détails d'un événement

Cette interface illustrée par les figures 5.5 et 5.6 montre deux sections une liée à tous les détails d'un événement donné : image, description, durée, organisateur. Quant à la deuxième section est dédiée aux sessions disponibles. Dans le cas où une session est déjà passée le bouton "réserver" sera désactivé. L'utilisateur peut filtrer les sessions en fonction du lieu de l'événement ou en définissant une période avec une date de début et une date de fin.

A screenshot of the SofiaTicket website showing the details page for the 'Exo Planet #5 – Exploration' concert. At the top, there's a banner with the event name and duration (180 minutes). Below the banner, the title 'Exo Planet #5 – Exploration' is displayed. The page is divided into two main sections: 'General Informations' on the left and 'Description:' on the right. Under 'General Informations', there are several items: a 'Concert' icon followed by 'Concert', a clock icon followed by '180 Minute(s)', a person icon followed by 'Organized By: EXO Entertainment', an envelope icon followed by 'EXO@gmail.com', and a checkmark icon followed by 'Payment en ligne'. The 'Description:' section contains a detailed paragraph about the tour, mentioning it was announced on May 30, 2019, began on July 19, 2019, and concluded on December 31, 2019, in 9 countries. A 'See Less' link is at the bottom of this section.

FIGURE 5.5 : Interface détails d'un événement

**FIGURE 5.6 :** Interface sessions disponibles

Après avoir consulté les détails des événements et des sessions, l'utilisateur devra ajouter une réservation au panier. Cette fonctionnalité sera décrite en détail lors du prochain Sprint.

5.2 Sprint 6 : Gestion panier

Dans cette section nous allons présenter les différentes étapes de la réalisation du premier Sprint 6 " Gestion panier".

5.2.1 Objectifs du Sprint 6

L'objectif de ce Sprint est d'implémenter la fonctionnalité de gestion du panier, permettant aux utilisateurs de consulter le contenu du panier, d'ajouter, de modifier et de supprimer des réservations.

5.2.2 Backlog du Sprint 6

Id	Fonctionnalités (User stories)	Priorité	Estimation (Jour)
1	En tant qu'un client, je veux ajouter une réservation à mon panier.	1	5
2	En tant qu'un client, je veux consulter mon panier.	1	3
3	En tant qu'un client, je veux modifier la quantité de billets que je souhaite réserver.	2	4
4	En tant qu'un client, je veux supprimer une réservation de mon panier.	2	3

TABLEAU 5.2 : Backlog pour la Sprint 6

5.2.3 Conception

5.2.3.1 Spécification des besoins fonctionnels

La figure 5.7 représente le diagramme de cas d'utilisation du Sprint 6.

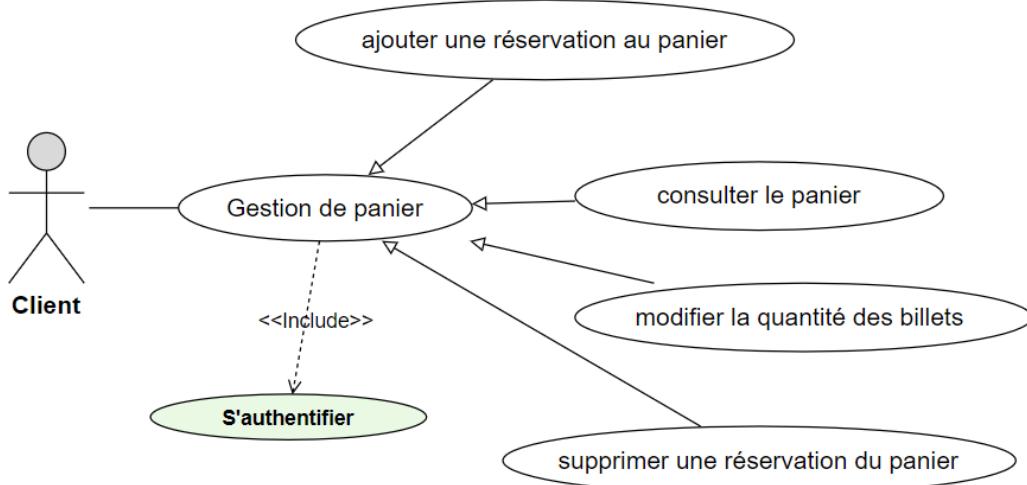


FIGURE 5.7 : diagramme de cas d'utilisation du Sprint 6

5.2.3.2 Diagramme de classe du Sprint 6

La figure 5.8 représente le diagramme de classe du Sprint 6.

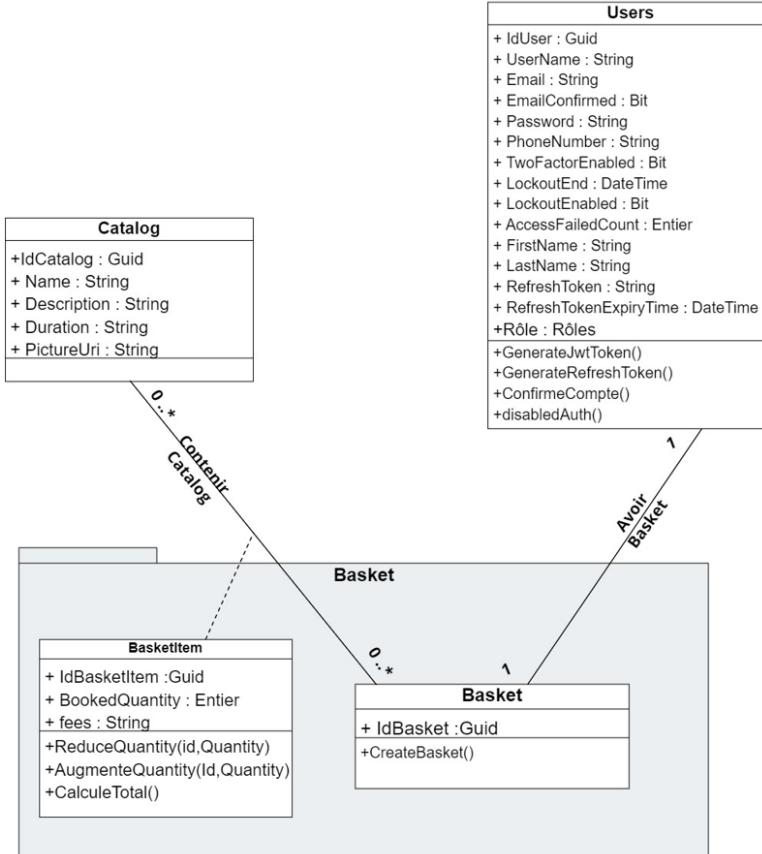


FIGURE 5.8 : diagramme de cas d'utilisation du Sprint 6

5.2.3.3 Diagramme d'activité pour l'ajout d'une réservation au panier

La figure 5.9 représente le diagramme d'activité pour l'ajout d'une réservation au panier.

Le diagramme montre le processus de réservation de tickets pour un événement ou l'utilisateur accède au site web de l'application et sélectionne une session d'un événement puis il choisit le nombre de bià réserver pour chaque type. En cliquant sur "ajouter au panier" si le nombre de billets à réserver est strictement positif et le client est authentifié, il peut visualiser son panier sinon il doit s'authentifier avant de continuer.

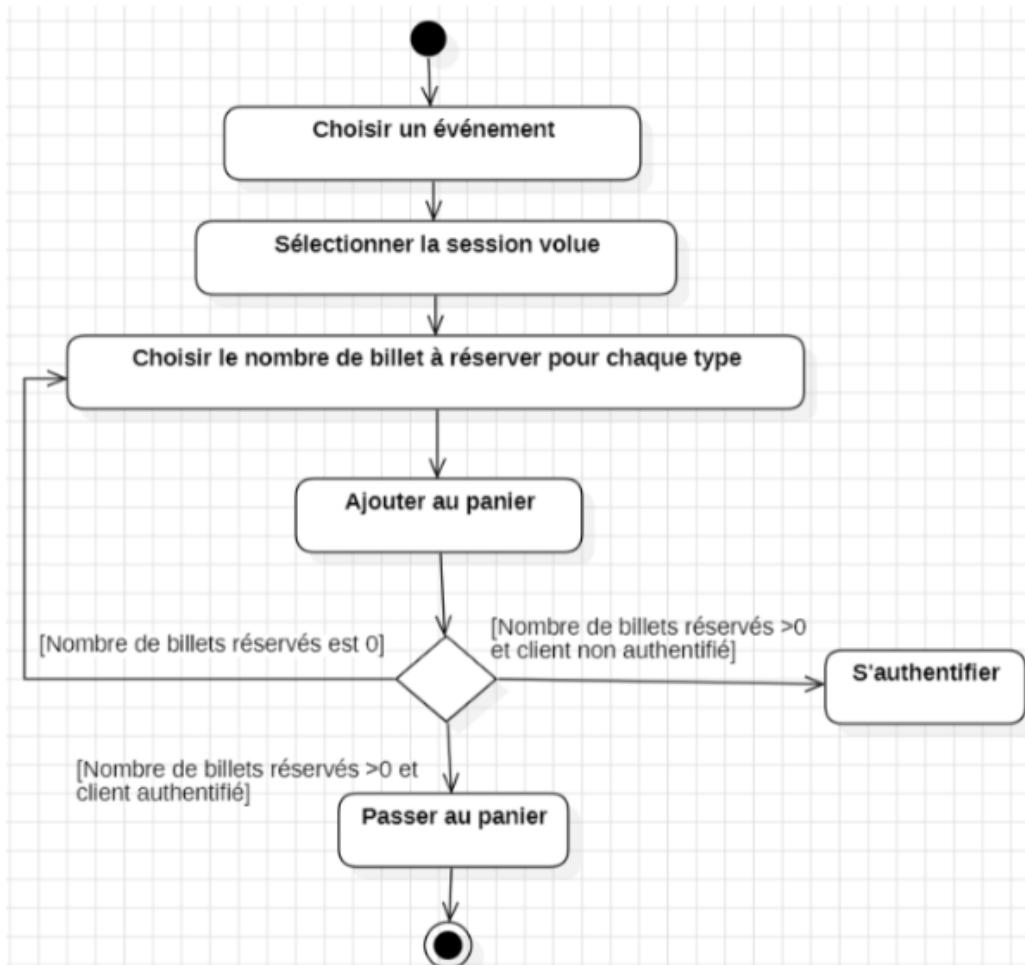


FIGURE 5.9 : Diagramme d'activité pour l'ajout d'une réservation au panier

5.2.4 Réalisation

Interface Page de Réservation

L'interface présentée par La Figure 5.9, montre les détails de l'événement ainsi que les différents types de billets disponibles. L'utilisateur peut sélectionner la quantité de billets à réserver pour chaque type et, s'ils sont disponibles, les ajouter à son panier.

Après l'ajout réussi d'une réservation au panier, le bouton "Pass Order" devient activé. En cliquant

sur "Pass Order", l'utilisateur sera redirigé vers son panier afin de le consulter.

The screenshot shows the SofiaTicket website's ticket booking interface. On the left, a sidebar lists categories: Theatre, Cirque, SPECTACLE, Concert, Transport, Cinema, Travelling, test, and FESTIVAL. The main area displays a ticket for 'Exo Planet #5 – Exploration'. The ticket details are: Ticket Type: VIP, Price: 20 TND, State: Disponible, and Quantity: 0. Below these details are two buttons: 'Add to basket' (in red) and 'Pass Order' (in blue). To the right, the event title 'Exo Planet #5 – Exploration' is displayed, along with its date (10/11/2024 20:00), duration (180 Minute(s)), and location (ras jebal - bizerete). A small thumbnail image of the event is shown, featuring a group of people standing in front of a large celestial body labeled 'EXO PLANET #5 EXplOration'. Below the thumbnail, a caption reads: 'Exo Planet #5 – Exploration (stylized as EXO PLANET #5 – EXplOration) is the fifth concert tour headlined by South Korean-Chinese boy band Exo. The tour was officially announced on May 30, 2019, and'.

FIGURE 5.10 : Interface Page de Réservation

Interface “ Panier”

L'interface illustrée par la figure 5.11 Présente le panier d'un client montrant les billets à réserver en précisant la quantité, le prix unitaire et le montant avec frais. L'utilisateur peut augmenter ou diminuer le nombre des billets comme il peut le supprimer de son panier.

The screenshot shows the SofiaTicket website's basket interface. The title 'Your basket' is at the top. Below it is a table listing two items:

Ticket	BASKET.SESSION	Price	Fees	Quantity	Total	Delete
Exo Planet #5 – Exploration	10/11/2024, 08:00 pm bizerete, ras jebal, Tunisia	20	1.5	- 2 +	TND43.000	
Cirque du Soleil CRYSTAL	30/10/2024, 09:00 am tunis, oriana soghra, Tunisia	20	1.5	- 2 +	TND43.000	

At the bottom of the basket, it says 'Total: 86 TND' and has a 'Confirm' button.

FIGURE 5.11 : Interface “ Panier”

5.3 Sprint 7 : Gestion des réservations et de paiement

Dans cette section nous allons présenter les différentes étapes de la réalisation du premier Sprint 7 " Gestion des réservations et de paiement".

5.3.1 Objectifs de Sprint 7

L'objectif de ce sprint est de permettre à l'utilisateur de finalisé une réservation en intégrant un processus de paiement sécurisé. Un QR code unique sera généré pour chaque réservation confirmée, servant de preuve de réservation. Ce QR code permettra une vérification rapide et sécurisée à l'entrée de l'événement, facilitant l'accès et réduisant les risques de fraude.

5.3.2 Backlog de Sprint 7

Id	Fonctionnalités (User stories)	Priorité	Estimation (Jours)
1	En tant qu'un client, je peux passer une réservation .	1	4
1	En tant qu'un client, je peux payer une réservation.	1	3
2	En tant qu'un client, je peux recevoir un email contenant la liste de mes réservations.	3	3
3	En tant qu'un client, je veux avoir un billet sécurisé avec un QR code unique.	1	3
4	En tant qu'un client, je peux consulter mes réservations et billets.	1	6
5	En tant qu'un client, je peux télécharger mes billets.	2	1

TABLEAU 5.3 : Backlog de Sprint 7

5.3.3 Technologies utilisées

- **Stripe** : Stripe est une plateforme de paiement en ligne qui permet aux entreprises de traiter des paiements par carte de crédit, d'effectuer des paiements récurrents, et de gérer diverses transactions financières via une API [38].
- **Stripe.NET** : Stripe.NET est une bibliothèque officielle pour intégrer les services de paiement Stripe dans des applications .NET [39].
- **QRCoder** : QRCode est une bibliothèque simple, écrite en C#.NET, qui permet de créer des codes QR. Elle n'a aucune dépendance envers des bibliothèques externes, est disponible sous forme de package sur NuGet, et prend en charge .NET Framework, .NET Core, .NET Standard et .NET [40].

5.3.4 Conception

5.3.4.1 Spécification des besoins fonctionnels

La figure 5.12 représente le diagramme de cas d'utilisation de Sprint 7.

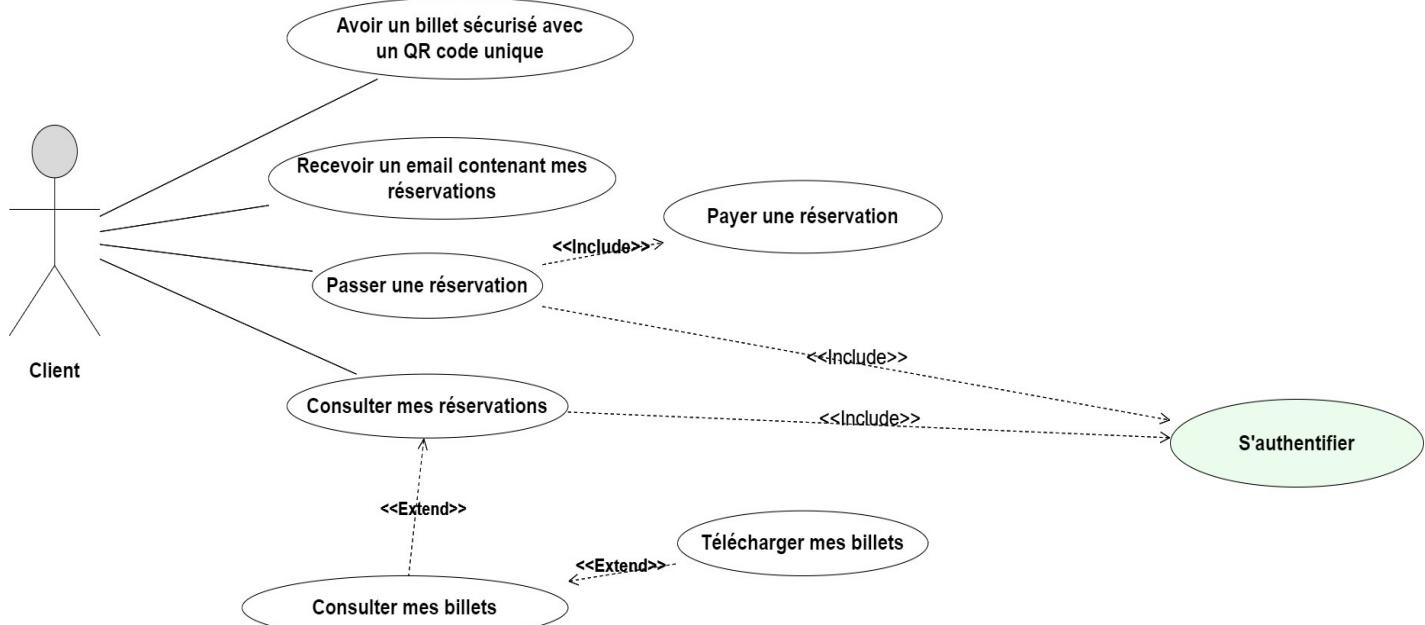


FIGURE 5.12 : diagramme de cas d'utilisation du Sprint 7

5.3.4.2 Diagramme de classes de Sprint 7

La figure 5.14 représente le diagramme de classes de Sprint 7.

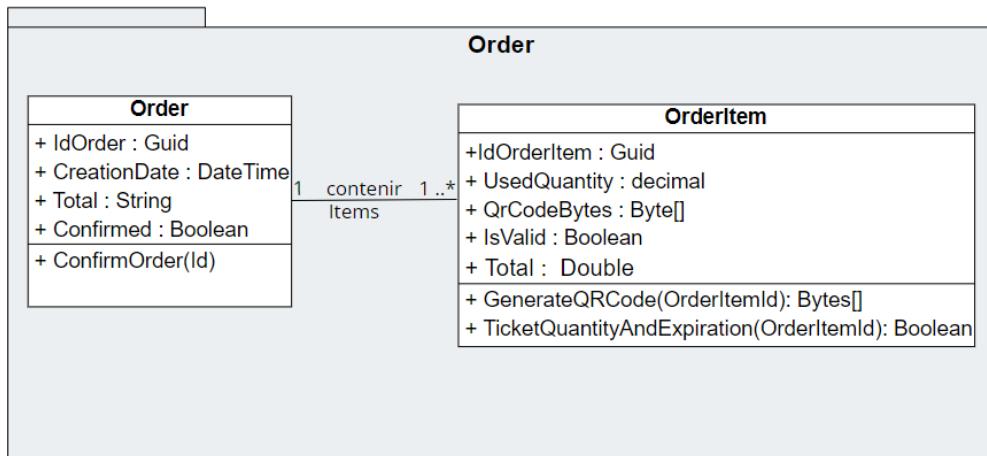


FIGURE 5.13 : Diagramme de classe de Sprint 7

La relation entre ces deux entités est une relation de contenance, où une commande (Order) peut contenir plusieurs articles de commande (OrderItem). Cette structure de données permet de stocker les informations de la commande, telles que la date de la réservation, le montant total, tout en conservant les détails de chaque article commandé, comme le type de billet, l'événement, la session, le prix et les frais.

5.3.5 Diagramme d'activité pour la confirmation de réservation et paiement

Après la visualisation du panier, l'utilisateur peut passer sa réservation. Une décision est prise : si la commande est confirmée, il passe au paiement sinon, la commande est annulée. Pour pouvoir finalisé le paiement il doit confirmer ses données personnelles puis la commande. Ensuite il passe à la session de paiement ou il doit insérer les données de sa carte. Si le paiement est fait avec succès, il peut passer voir ses commandes sinon il reviendra à la page de son panier. Ce processus est illustré par le diagramme de la figure 5.14

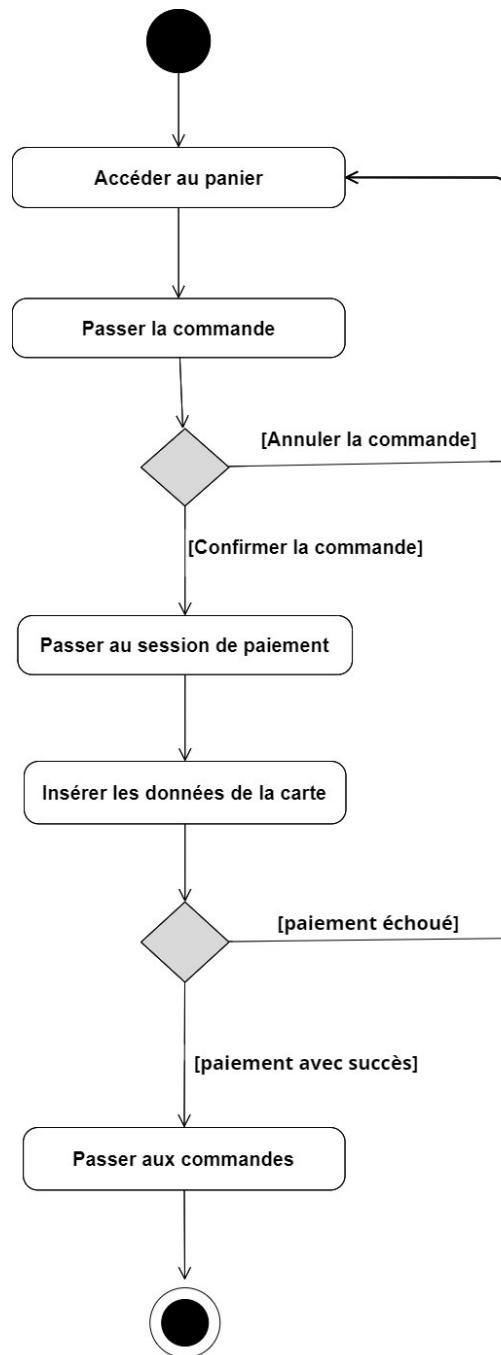


FIGURE 5.14 : Diagramme d'activité pour la confirmation de réservation et paiement

5.3.5.1 Diagramme de séquences pour le paiement avec stripe

Le diagramme de la figure 5.15 illustre le processus de paiement utilisant le service Stripe.

Il montre comment une application web intègre Stripe pour gérer les paiements de manière sécurisée, en utilisant des sessions de paiement générées par Stripe et en redirigeant l'utilisateur vers l'interface de paiement de Stripe avant de le ramener sur le site de l'application.

Le processus commence par une demande de paiement envoyée de l'application Web Client contenant le montant à payer au service paiement à travers l'API Gateway ocelot. Le service "Payment.API" va envoyer une demande de session au service Stripe qui va créer une nouvelle session de paiement et le retourne l'ID de la session. PaymentAPI transmet l'URL de la session à travers l'api Gateway au Web Client qui va naviguer vers ce dernier où l'utilisateur peut effectuer le paiement sur la page Stripe.

Une fois le paiement traité, Stripe redirige l'utilisateur vers la page correspondante sur le site Web de l'application ; si le paiement est effectué avec succès il va naviguer vers la page de succès de paiement sinon il va retourner à son panier.

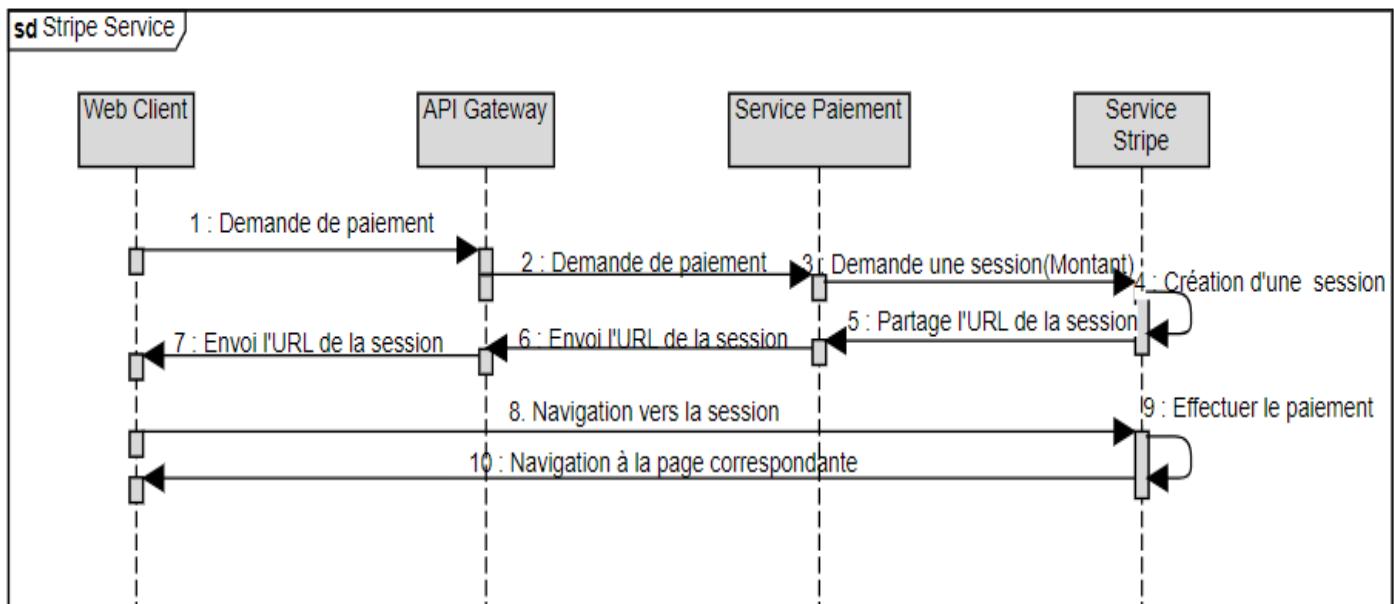


FIGURE 5.15 : Diagramme de séquences pour le paiement avec stripe

5.3.6 Réalisation

Interface "Page de Confirmation de Commande"

Cette interface, illustrée par la figure 5.16, présente l'étape avant le paiement où le client confirme ses données personnelles (nom, prénom, email et numéro de téléphone) ainsi que les détails de sa commande. L'utilisateur peut vérifier les billets réservés, la quantité, les frais associés et le montant

total à payer avant de procéder au paiement.

The interface consists of two main sections: 'IDENTIFICATION DE L'ACHETEUR' (Buyer Identification) on the left and 'VOTRE COMMANDE' (Your Order) on the right.

IDENTIFICATION DE L'ACHETEUR:

- Prenom: neyssen
- Nom: ben romdhane
- Email*: user1@gmail.com
- Numéro de téléphone*: 53696823

VOTRE COMMANDE:

Billet	Quantité	Prix
Exo Planet #5 - Exploration	1	50 TND
Exo Planet #5 - Exploration	2	10 TND
Exo Planet #5 - Exploration	2	10 TND
Exo Planet #5 - Exploration	1	50 TND

Nb réservation: 6
Frais: 9 TND
Montant Total: 149 TND

Payer

FIGURE 5.16 : Interface "Page de Confirmation de Commande"

Interface "Page de Paiement"

Cette interface, illustrée par la figure 5.17 , présente le processus de paiement où le client doit entrer ses informations de paiement. L'utilisateur est invité à spécifier son email, le numéro de la carte de crédit, la date d'expiration et le nom du titulaire de la carte. Une fois ces données saisies, le système vérifie leur validité. Si les informations sont valides et le paiement est réussi, l'utilisateur sera redirigé vers une page de confirmation où il pourra consulter les détails de ses commandes et le statut de son paiement.

The payment page is divided into two main sections: 'Product Name' and 'Payer par carte'.

Product Name:

- ProductName: 149,00 \$US
- Product Description:

Payer par carte:

- E-mail: [Input field]
- Informations de la carte: [Card number: 1234 1234 1234, Expire: MM / AA, CVC: CVC]
- Nom du titulaire de la carte: [Input field]
- Pays ou région: Tunisie
- Enregistrer mes informations en toute sécurité pour le paiement en un clic
- Payer**

Propulsé par stripe | Conditions d'utilisation | Confidentialité

FIGURE 5.17 : Interface "Page de Paiement"

Si les informations de paiement sont valides et que la transaction est réussie, l'utilisateur sera

redirigé vers une page de "succès du paiement". Sur cette page, il pourra consulter les détails de ses commandes.

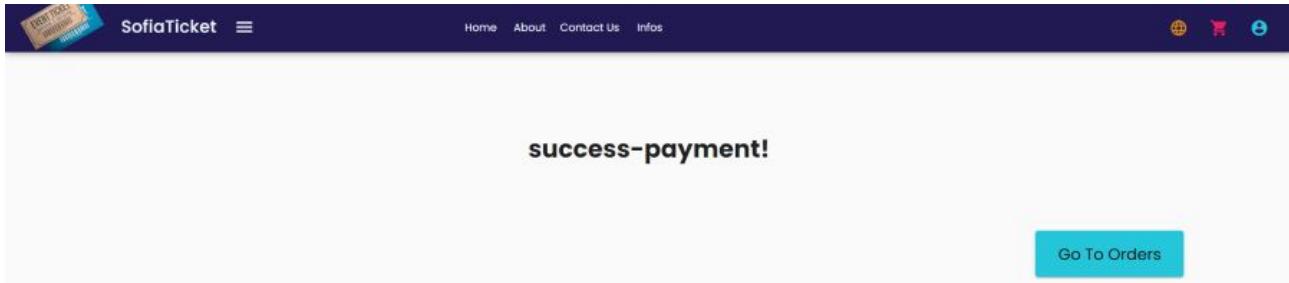


FIGURE 5.18 : Interface "succès du paiement"

De plus, un email récapitulatif de sa réservation lui sera envoyé, contenant les informations détaillées de sa commande.



FIGURE 5.19 : Enter Caption

Interface“ Mes réservations”

L’interface illustrée par la figure 5.20 montre les réservations qu’un client a passé ou il peut visualiser ses billets en cliquant sur “show me” .

Event	Number of tickets	Total amount
Exo Planet #5 - Exploration	2	43
Exo Planet #5 - Exploration	2	43
Exo Planet #5 - Exploration	2	43

FIGURE 5.20 : Interface“ Mes réservations”

Interface “ Mes Billets”

L’interface suivante illustrée par la figure 5.21 montre un billet généré pour un client à une session d’un événement. Le client peut télécharger le billet , qui peut être utilisé directement pour accéder à l’événement sans connexion internet.

Concert Ticket

Event: Exo Planet #5 – Exploration

Date: 10/11/2024 | 08:00 PM

Location: ras jebal , bizerte

Ticket Holder: Ben romdhane neyssen
2 tickets
Ticket Type: VIP
Total Price: 43 TND
Reserved in: 30/08/2024 11:43 PM

QR Code:

Export as JPG

FIGURE 5.21 : Interface “ Mes Billets”

Conclusion

Ce chapitre aborde la conception et la réalisation des fonctionnalités clés de la Release 3, notamment la consultation des événements, la gestion du panier, les réservations et le paiement. Ces améliorations ont été soigneusement développées pour offrir une navigation fluide et une expérience de paiement sécurisée, renforçant ainsi la satisfaction des utilisateurs et illustrant notre engagement à optimiser la plate-forme en fonction de leurs besoins.

RELEASE 4 : TABLEAU DE BORD, SCANNING DE BILLETS ET INTERFACE MULTILINGUE

Plan

1	Sprint 8 : Consulter le tableau de bord et les rapports de vente	88
2	Sprint 9 : Scanner les billets et Interface utilisateur multilingue	92

introduction

Dans ce chapitre, nous allons décrire la Release 4, qui se concentre sur les tableaux de bord, le scanning des billets et l'interface multilingue. Nous présentons chaque fonctionnalité en illustrant leur conception à l'aide de diagrammes, ainsi que leur réalisation, notamment les interfaces de l'application.

6.1 Sprint 8 : Consulter le tableau de bord et les rapports de vente

Dans cette section nous allons présenter les différentes étapes de la réalisation du Sprint 8 "Consulter le tableau de bord et les rapports de vente".

6.1.1 Objectifs de Sprint 8

L'objectif de ce sprint est de permettre aux organisateurs de consulter en temps réel les rapports de vente de leurs tickets via le tableau de bord. Cela inclut la visualisation du nombre de tickets réservés, du total des tickets par événement, et des types de billets vendus pour chaque session. L'état financier n'est pas inclus dans ce sprint, car il est géré directement via le tableau de bord Stripe, qui fournit tous les rapports nécessaires.

De plus, ce sprint permettra à l'administrateur de suivre les statistiques de l'application, en mettant en avant les événements, catégories et organisateurs les plus performants en termes de billets vendus, ainsi que d'accéder aux informations de contact des organisateurs.

6.1.2 Backlog de Sprint 8

ID	Fonctionnalités (User Stories)	Priorité	Estimation (Jours)
1	En tant qu'un organisateur, je peux visualiser le total des billets vendus par événement.	1	4
2	En tant qu'un organisateur, je peux visualiser les types de billets vendus par session.	1	4
3	En tant qu'un administrateur, je peux identifier les événements, les catégories et les organisateurs les plus performants.	2	4
4	En tant qu'un administrateur de l'application, je peux suivre les ventes de billets par événement et organisateur.	2	6
5	En tant qu'un administrateur, je peux consulter les coordonnées des organisateurs.	3	2

TABLEAU 6.1 : Backlog de Sprint 8

6.1.3 Technologies utilisées

ApexCharts : est une bibliothèque de graphiques moderne qui aide les développeurs à créer de belles visualisations interactives pour les pages Web.

Il s'agit d'un projet open source sous licence MIT et son utilisation est gratuite dans des applications commerciales. [41]

6.1.4 Spécifications des besoins fonctionnels

La figure 5.1 représente le diagramme de cas d'utilisation de Sprint 8.

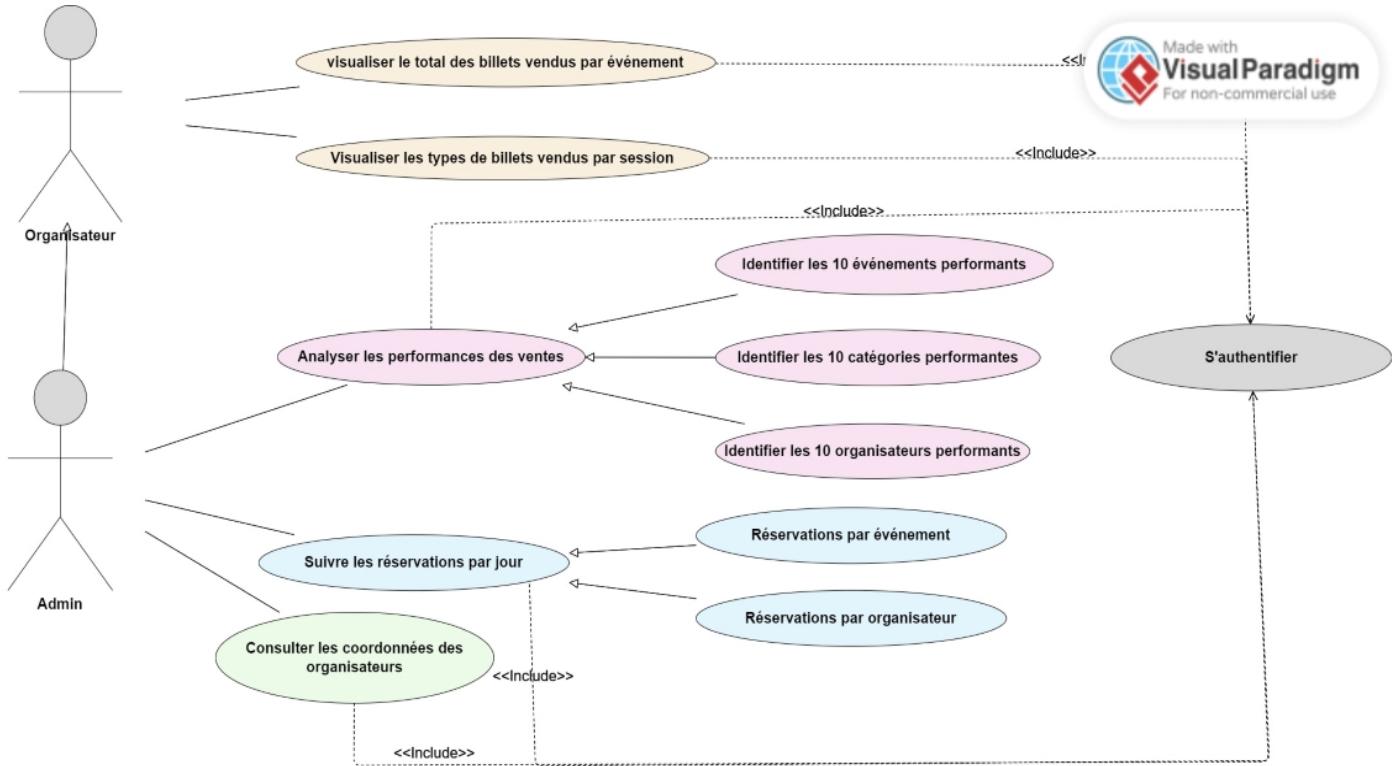


FIGURE 6.1 : Diagramme de cas d'utilisation de Sprint 8

6.1.5 Réalisation

Interface “ Statistiques des Événements”

La page du tableau de bord présenté par la figure 6.2 permet à l'organisateur de visualiser le nombre de billets vendus par rapport au total des billets pour chaque événement, ainsi que la liste des événements qu'il a organisés.

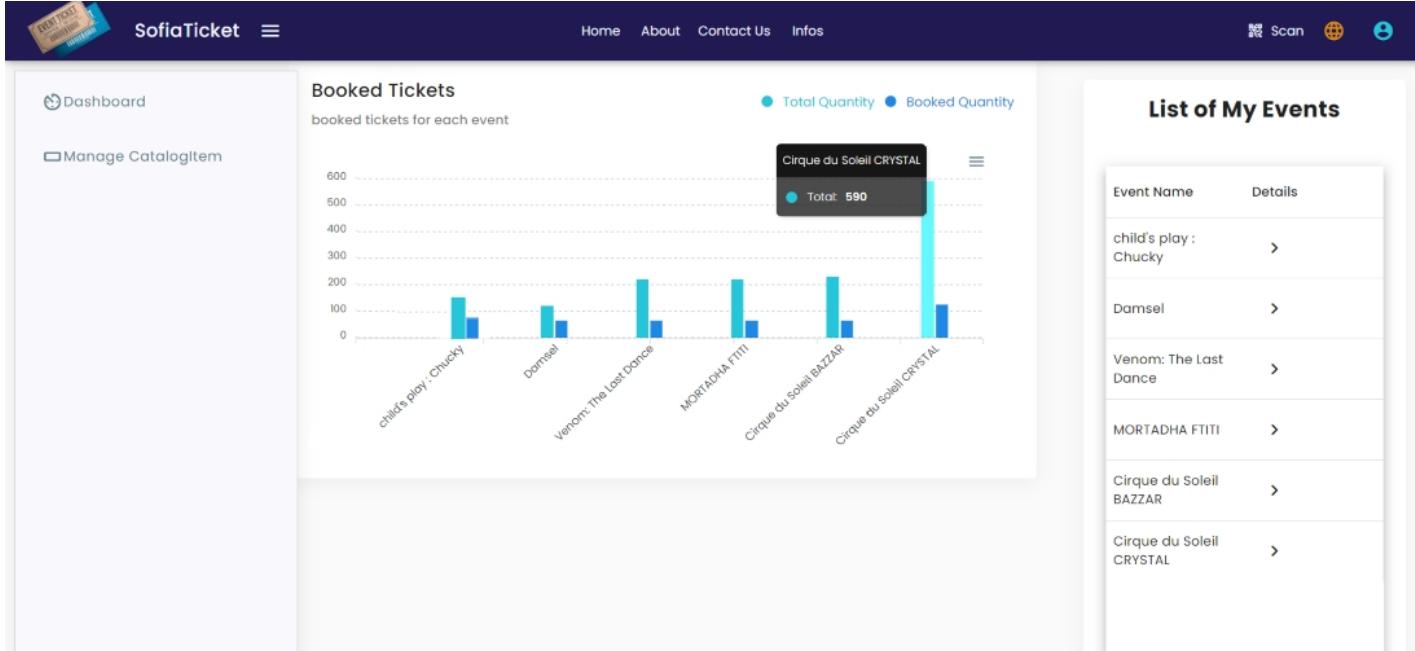


FIGURE 6.2 : Interface “ Statistiques des Événements”

6.1.6 Interface “ Statistiques des sessions”

En sélectionnant un événement, l'organisateur peut voir en détail la quantité des tickets vendues pour chaque type de ticket pour toutes les sessions comme illustré par la figure 6.3.

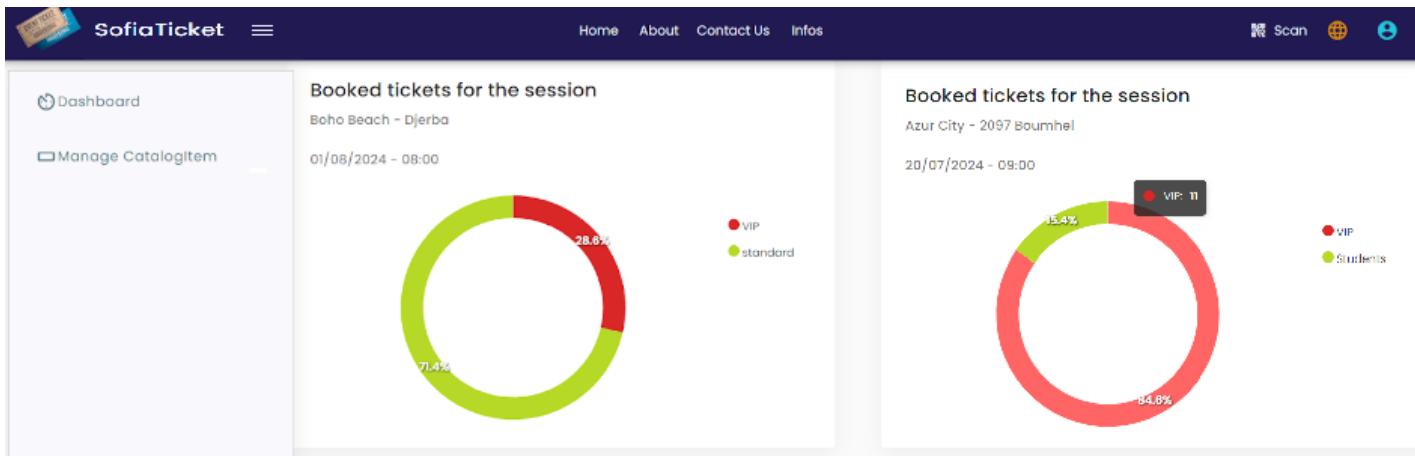
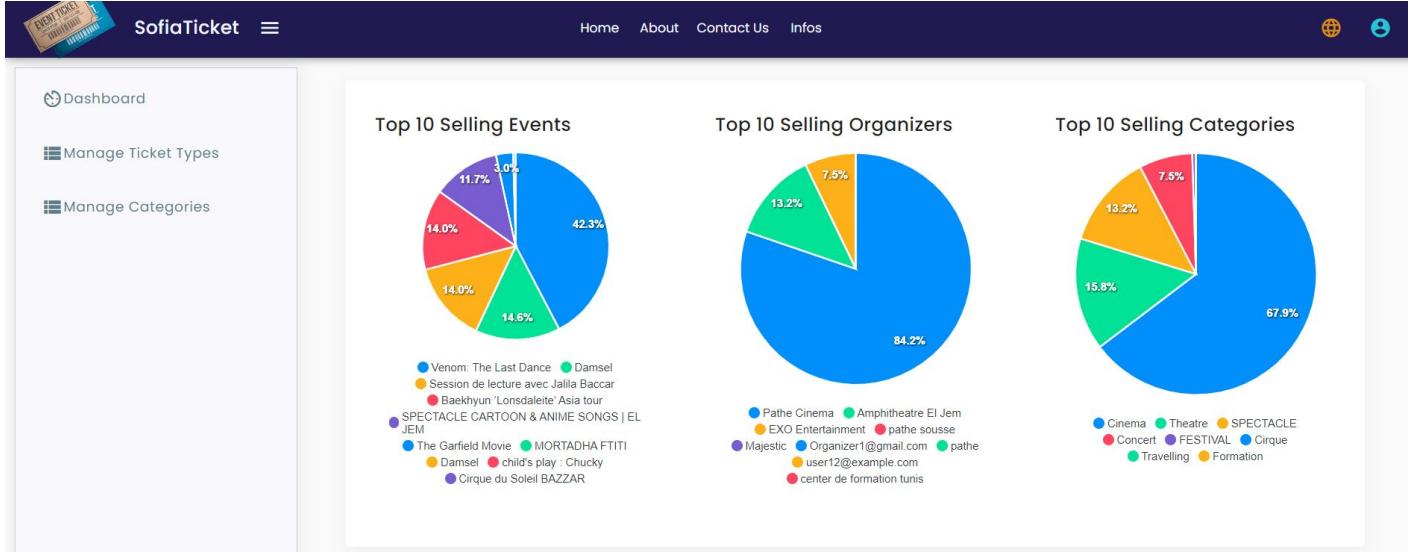


FIGURE 6.3 : Interface “ Statistiques des sessions”

Interface "Tableau de Bord des Performances"

L'interface Tableau de Bord des Performances présentée dans la figure 6.4 permet à l'administrateur de l'application de visualiser les 10 meilleurs événements, catégories et organisateurs ayant réalisé le plus de ventes de billets.

**FIGURE 6.4 :** Interface "Tableau de Bord des Performances"

La figure 6.5 et la figure 6.6 montrent l'interface qui permet à l'administrateur de surveiller les ventes de billets jour par jour pour chaque événement et chaque organisateur, ainsi que d'accéder aux informations de contact des organisateurs.

**FIGURE 6.5 :** Interface réservations par événement

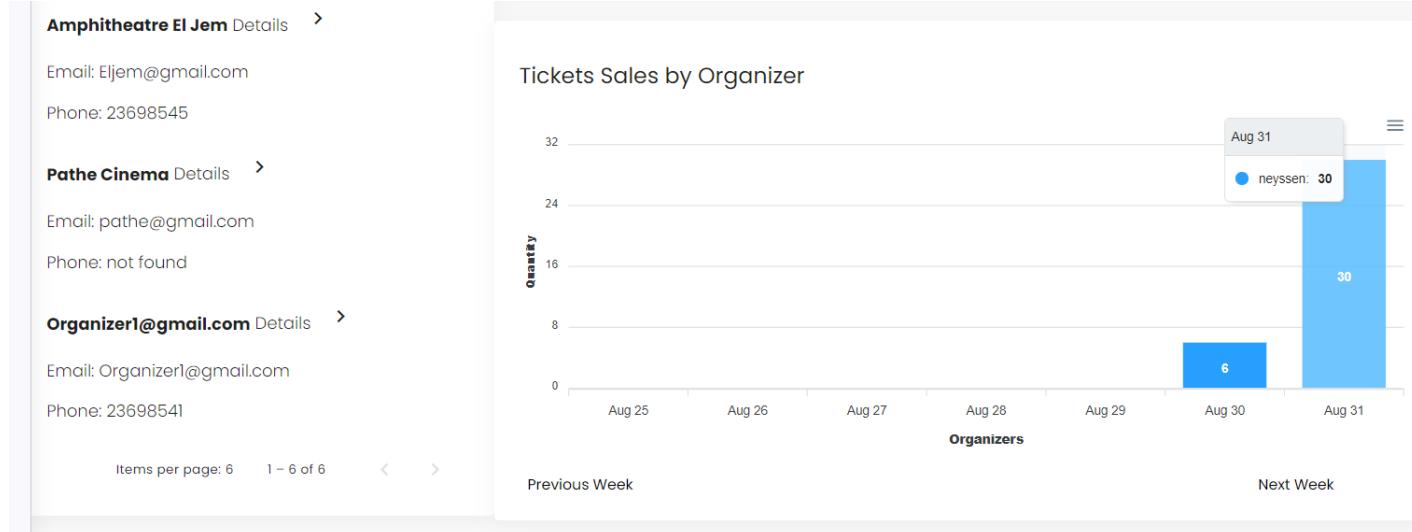


FIGURE 6.6 : Interface réservations par organisateur

6.2 Sprint 9 : Scanner les billets et Interface utilisateur multilingue

Dans cette section nous allons présenter les différentes étapes de la réalisation du Sprint 9 "Scanner les billets et Interface utilisateur multilingue".

6.2.1 Objectifs de Sprint 9

L'objectif de ce sprint est de permettre aux organisateurs de scanner les billets à l'entrée de l'événement pour vérifier leur validité.

De plus, notre application démontre sa capacité à afficher du contenu dans plusieurs langues, offrant ainsi une expérience utilisateur personnalisée et adaptée à différentes audiences internationales.

ID	Fonctionnalités (User Stories)	Priorité	Estimation (Jours)
2	En tant qu'organisateur, je peux scanner les billets à l'entrée de l'événement pour vérifier leur validité.	1	7
1	En tant qu'utilisateur de "SofiaTicket", je peux choisir la langue de l'application pour naviguer dans ma langue préférée.	2	8

TABLEAU 6.2 : Backlog de Sprint 9

6.2.2 Technologies utilisées

- **Internationalisation (i18n)** : Le processus d'adaptation d'un logiciel pour prendre en charge plusieurs langues et cultures. [42]
- **IoFile** : Un service de partage de fichiers temporaire qui permet aux utilisateurs de télécharger

des fichiers et de partager un lien unique. Les fichiers sont automatiquement supprimés après avoir été téléchargés ou après une période d'expiration définie, même s'ils n'ont jamais été téléchargés. Les fichiers sont également cryptés pour assurer leur sécurité. [43]

- **ngx-translate** : Une bibliothèque populaire qui aide les développeurs à ajouter la prise en charge d'i18n à leurs projets. En utilisant ngx-translate, les développeurs peuvent facilement traduire du texte statique dans leurs applications, ainsi que du contenu dynamique généré par les interactions des utilisateurs. Avec son API simple et sa documentation complète, ngx-translate permet aux développeurs d'ajouter facilement l'internationalisation à leurs projets Angular et de garantir que leurs applications sont accessibles aux utilisateurs du monde entier. [44]
- **ngx-scanner-qrcode** : Cette bibliothèque est conçue pour fournir une solution de scanner de code QR. Elle prend des images brutes et localise, extrait et analyse tout code QR trouvé à l'intérieur. [40]

6.2.3 Conception

6.2.3.1 Diagramme d'activité de la procédure de scan d'un billet

La figure 6.7 illustre le diagramme de la procédure de scan d'un billet.

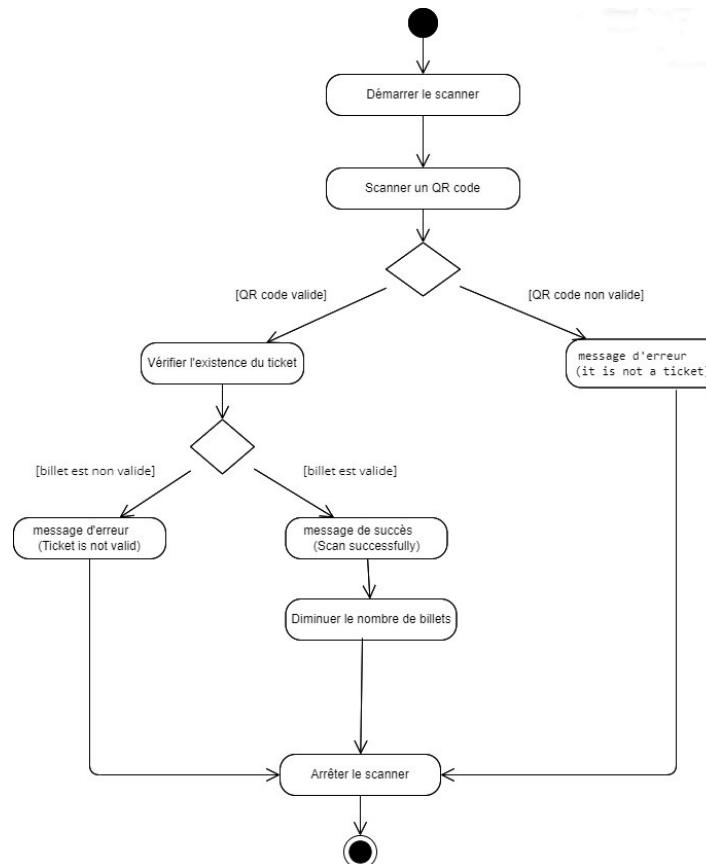


FIGURE 6.7 : diagramme de la procédure de scan d'un billet

6.2.3.2 Diagramme de Séquence pour le Support Multilingue

La figure 6.8 illustre le diagramme de Séquence pour le Support Multilingue.

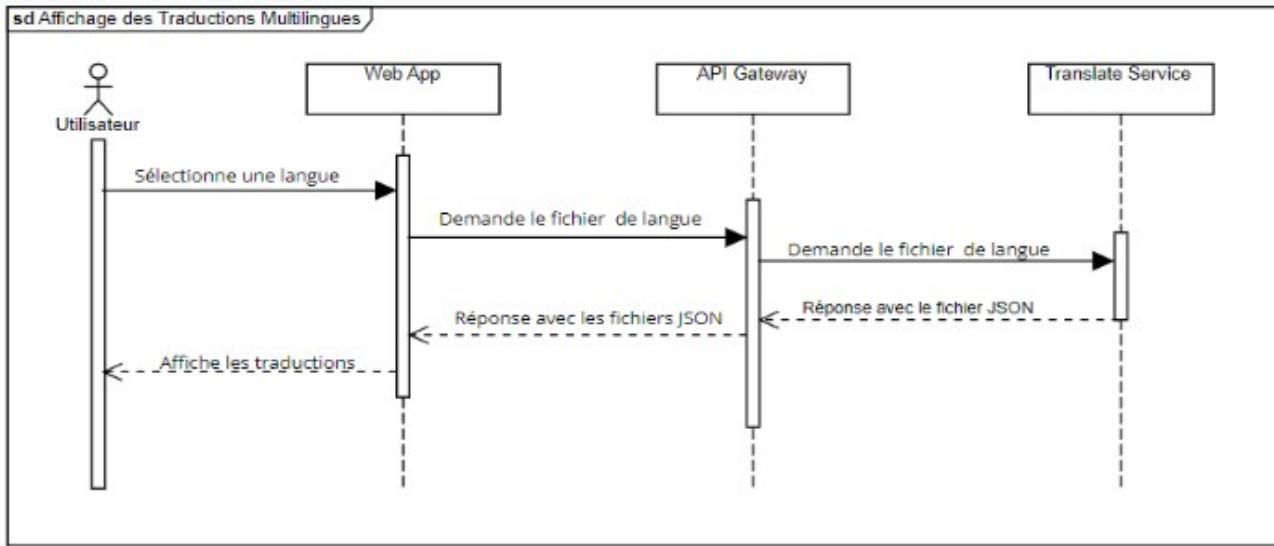


FIGURE 6.8 : diagramme de Séquence pour le Support Multilingue

6.2.4 Réalisation

En cliquant sur le bouton scan, la caméra de l'organisateur s'ouvre en mode scan et en approchant le billet, une analyse du QrCode est faite : si le billet est encore valable alors un message “ticket is valid” s’affiche sinon “ticket is not valid” comme le montre les figures 6.9 , 6.10 , et 6.11. suivantes.

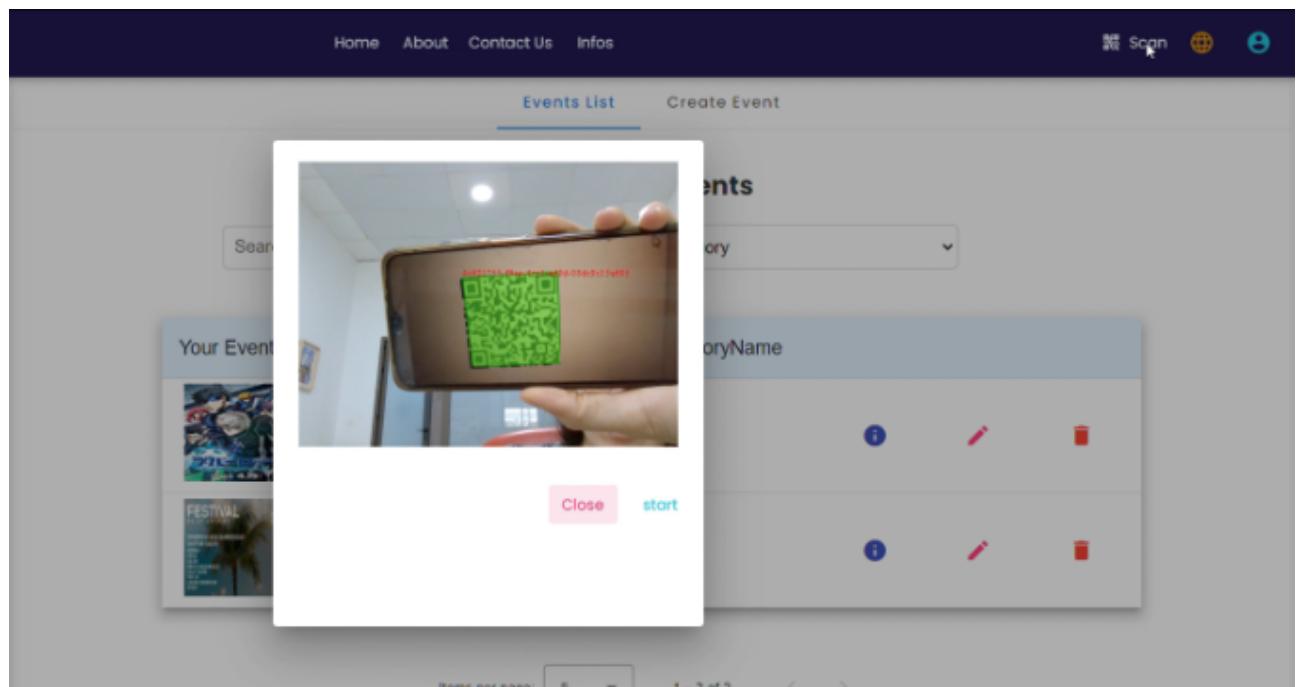


FIGURE 6.9 : Interface Scanner un billet

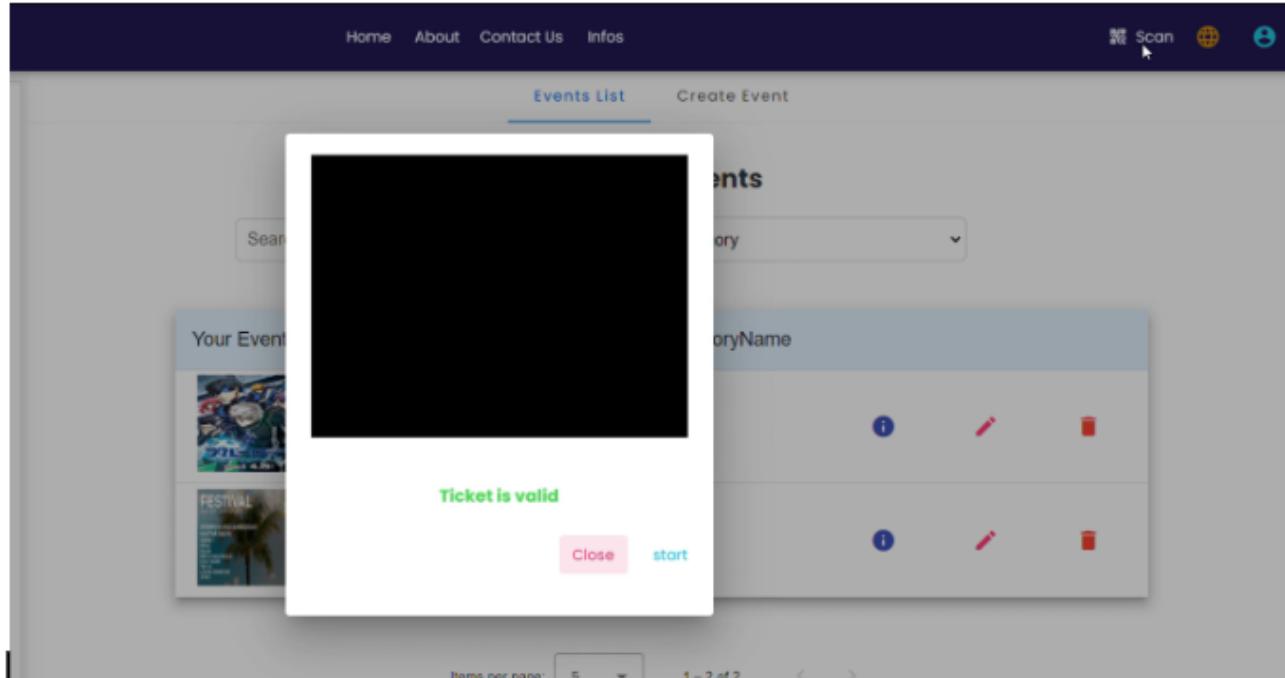


FIGURE 6.10 : Interface Un billet est valide

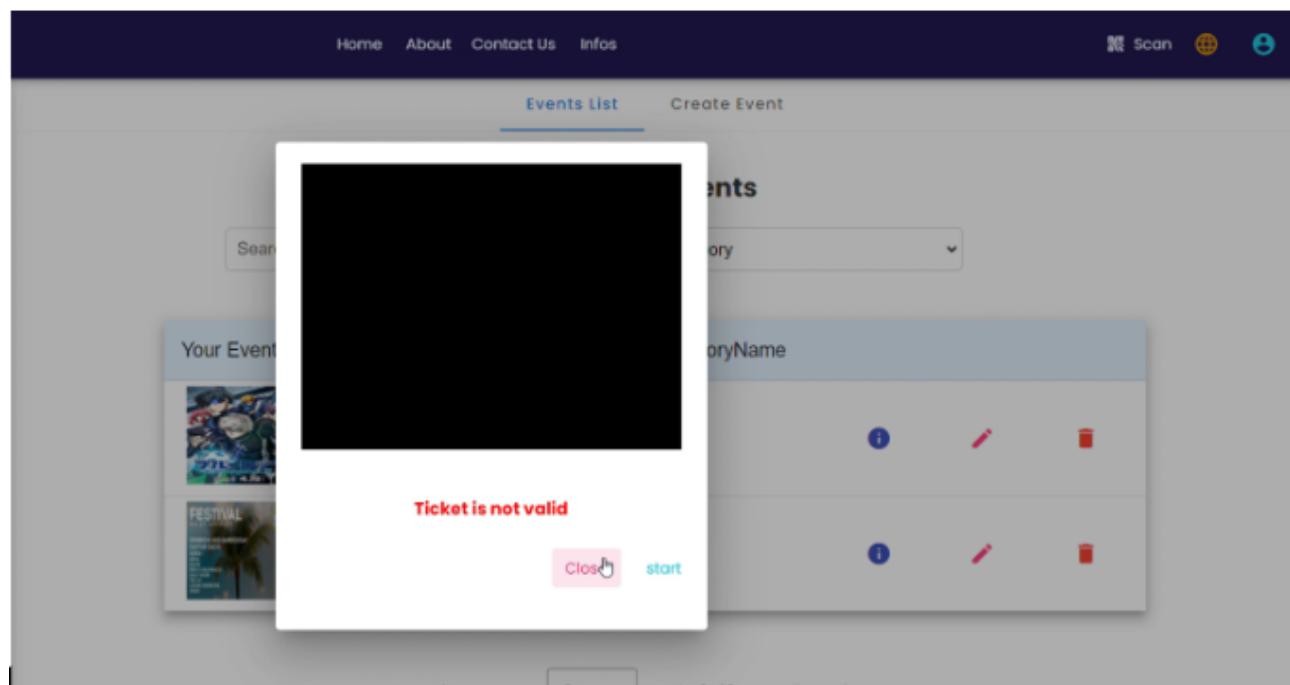


FIGURE 6.11 : Interface Un billet est non valide

6.2.4.1 Interfaces Multilingues

L'application ‘SofiaTicket’ permet aux utilisateurs de choisir leur langue préférée parmi les options disponibles, telles que l'anglais, le français, le coréen, l'arabe, et le chinois. Une fois la langue sélectionnée, l'application traduit automatiquement le contenu de la page dans la langue choisie. Les figures 6.12 , 6.13, 3.14, 6.15 ,6.16 et 6.17 montrent les différentes interfaces de l'application traduites

dans les langues sélectionnées. Chaque figure illustre comment l'application adapte son contenu pour offrir une expérience utilisateur cohérente dans les langues choisies.

The screenshot shows the SofiaTicket website's shopping basket page in English. At the top, there is a navigation bar with links for Home, About, Contact Us, and Infos. On the right side of the header, there is a language selection dropdown menu with options: English, French, Deustch, Chinois, Arabic, and Korean. The main content area is titled "Your basket" and contains a table with one item: "Exo Planet #5 – Exploration" with a price of "TND20.000". Below the table, it says "Total: 21.5 TND" and has a "Confirm" button. The footer of the page includes sections for Help? (with contact information), About (links to Home, About, Infos, Login, Create an account, and Contact Us), and Categories (links to Theatre, Cirque, SPECTACLE, Concert, Cinema, and Travelling). There is also a decorative image of a ticket stub on the right side of the footer.

FIGURE 6.12 : Interface du panier en anglais

The screenshot shows the SofiaTicket website's shopping basket page in Korean. The layout is identical to the English version, with a navigation bar at the top and a language selection dropdown showing Korean as the selected option. The main content area, titled "장바구니", displays the same single-item basket entry and total amount. The footer features Korean versions of the Help?, About, and Categories sections, along with the same decorative ticket stub image.

FIGURE 6.13 : Interface du panier en coréen

The screenshot shows the 'Mon Profil' (My Profile) section of the SofiaTicket website. At the top, there's a navigation bar with links for Accueil, À propos, Nous contacter, Infos, and a user icon. Below the navigation is a form titled 'Mes Commandes' (My Orders) and 'Info utilisateur' (User Information). The user information fields include 'Prénom' (First Name) set to 'neyssen', 'Nom' (Last Name) set to 'Ben romdhane', 'Email' set to 'user1@gmail.com', and 'Numéro de téléphone' (Phone Number) set to '53654821'. A pink button labeled 'mettre à jour profil' (Update profile) is at the bottom of the form. To the right, there's a decorative graphic of two event tickets. On the left side of the main content area, there's a sidebar with 'Besoin d'aide?' (Need help?) and contact information: phone numbers +216 53 308 637 and +216 53 696 823, and an email address contact@SofiaTicket.tn. In the center, there's a 'À propos' (About) section with links to Accueil, À propos, Infos, se connecter, créer un compte, and Nous contacter. To the right, there's a 'catégories' (Categories) section listing Theatre, Cirque, SPECTACLE, Concert, Cinema, Travelling, FESTIVAL, and Formation.

FIGURE 6.14 : Interface mon profil en français

The screenshot shows the 'Mein Profil' (My Profile) section of the SofiaTicket website in German. The layout is similar to Figure 6.14, with a navigation bar, a form for 'Meine Bestellungen' (My Orders), and a 'Nutzerinformation' (User Information) section. The 'Meine Bestellungen' section is titled 'Meine Bestellungen' and lists two orders for 'Exo Planet #5 – Exploration'. Each order has a thumbnail image of a ticket stub, the event name, the quantity (2), and the total amount (43). A pink button labeled 'Zeig mir' (Show me) is visible on the right. The sidebar on the left includes 'Befehl n° #123456', 'Reservierungsdatum : 30/08/2024', 'nº der Reservierungen : 3', and 'Gesamtmenge: 129'.

FIGURE 6.15 : Interface mon profil en Allemagne

ملفي الشخصي

معلومات المستخدم

طلباتي

نº طلب: #123456
تاريخ الحجز: 30/08/2024
نº حجوزات: 3
إجمالي: 129

حدث	عدد التذاكر	إجمالي
	2	43
	2	43

FIGURE 6.16 : Interface mes réservations en Arabe

我的个人资料

我的订单

订单 n° #123456
预订日期: 30/08/2024
nº 预订数量: 3
总金额: 129

事件	票数	总金额
	2	43
	2	43

FIGURE 6.17 : Interface mes réservations en chinois

Conclusion

La release 4 a permis d'introduire des améliorations significatives à notre application, notamment avec l'ajout de la fonctionnalité de scanner de billets, la prise en charge multilingue et l'implémentassions de tableaux de bord. Nous présentons maintenant le dernier chapitre, qui aborde la phase finale de tests et de déploiement, afin de garantir le bon fonctionnement de toutes les fonctionnalités avant la mise en production.

RELEASE 5 : TESTS ET DÉPLOIEMENT

Plan

1	Introduction	100
2	Sprint 10 : Tests unitaires et tests d'intégration	100
3	Sprint 11 : Déploiement de l'application	104

7.1 Introduction

Pour assurer la maintenabilité de notre application et simplifier la gestion des erreurs potentielles, il est crucial d'établir un processus de test et de déploiement rigoureux. Cette section décrit les différentes étapes de test ainsi que la configuration de notre pipeline CI/CD pour garantir un flux de développement fiable et automatisé.

7.2 Sprint 10 : Tests unitaires et tests d'intégration

7.2.1 Objectifs de Sprint 10

Les tests jouent un rôle fondamental dans la qualité et la robustesse de notre application. Ils permettent de détecter les erreurs et les dysfonctionnements avant le déploiement en production. D'où faire le bon choix du Framework de test est une étape cruciale.

7.2.2 Choix du framework xUnit

Caractéristiques	NUnit	xUnit	MSTest
Attribut principal pour test	[Test]	[Fact] pour tests simples, [Theory] pour tests paramétrés	[TestMethod]
Paramétrage des tests	Supporte des tests paramétrés avec [TestCase]	Utilise [Theory] et [InlineData] pour les tests	Paramétrage possible mais moins flexible avec [DataRow]
Découverte des tests	Basé sur l'attribut [TestClass]	Pas besoin d'attribut [TestClass]	Basé sur l'attribut [TestClass]
Extensibilité	Moyenne	Très extensible	Moins extensible
Support des tests parallèles	Partiel, support ajouté	Support natif	Partiel, ajouté ultérieurement
Support multi-plateformes	Oui	Oui	Oui
Popularité et communauté	Large communauté, bien établi	En croissance rapide, recommandé	Traditionnel mais en perte de popularité

		pour les nouveaux projets	face à NUnit et xUnit
Compatibilité avec les outils CI/CD	Très bonne	Très bonne	Très bonne
Facilité d'écriture des tests	Similaire à MSTest, mais plus flexible	Plus fluide, moins de formalités (ex : pas besoin de [TestClass])	Simplicité mais plus formel

TABLEAU 7.1 : Comparaison des frameworks de test .NET

Après avoir comparé les frameworks, **xUnit** se démarque par sa flexibilité et son extensibilité, ce qui le rend idéal pour les projets qui nécessitent une approche modulaire et fluide des tests. L'absence de l'attribut [TestClass] et l'usage de [Fact] et [Theory] simplifient la structure des tests tout en permettant un meilleur contrôle sur les tests paramétrés.

Dans notre projet, qui se concentre principalement sur les tests unitaires et d'intégration, **xUnit** est un choix naturel pour sa capacité à s'adapter aux tests complexes et paramétrés, tout en assurant une excellente intégration dans les pipelines CI/CD. Ce framework offre la flexibilité et les performances nécessaires pour répondre à nos exigences de test, en particulier grâce à son support natif des tests parallèles et sa modularité.

7.2.3 Tests unitaires

Dans le domaine du développement web, les tests unitaires jouent un rôle fondamental pour garantir la qualité et la fiabilité des applications.

Un **test unitaire** est une pratique de test dans laquelle des sections de code, appelées “unités”, sont testées individuellement pour vérifier leur bon fonctionnement. Ces unités peuvent être des fonctions, des méthodes ou même des classes.

L’objectif principal d’un test unitaire est de s’assurer que chaque unité fonctionne comme prévu, indépendamment d’autres parties de l’application. Comme il a plusieurs intérêts dont nous citons :

- **Confiance dans le code** : Les tests unitaires offrent une couche de sécurité en vérifiant que chaque unité de code fonctionne correctement. Ils contribuent à renforcer la confiance des développeurs, des équipes de test et des utilisateurs dans la qualité du code et la fiabilité de l’application.
- **Vérifier la fonctionnalité** : Les tests unitaires permettent de s’assurer que chaque unité de code

fonctionne correctement, conformément à ses spécifications. Ils aident à valider le comportement attendu de l'unité et à détecter les erreurs de logique ou de calcul.

- **Maintenabilité du code** : En écrivant des tests unitaires, les développeurs créent également une documentation vivante de leur code. Ces tests servent de référence pour comprendre le comportement attendu de chaque unité de code, ce qui facilite la maintenance future et les modifications du code.
- **Détection des erreurs** : Les tests unitaires sont exécutés fréquemment, idéalement à chaque modification du code. Cela permet de détecter rapidement les erreurs et les bugs, facilitant ainsi leur résolution avant qu'ils ne se propagent dans d'autres parties de l'application.
- **Réutilisabilité** : Les tests unitaires encouragent la conception de code modulaire et bien structuré. En isolant chaque unité de code, les tests unitaires favorisent la réutilisabilité, car les unités peuvent être testées indépendamment et facilement intégrées dans d'autres parties de l'application.^[45]

7.2.4 Tests d'intégration

Bien que les tests unitaires soient utilisés pour tester des composants logiciels isolés, tels que des méthodes de classe individuelles, les tests d'intégration évaluent les composants d'une application à un niveau plus large que les tests unitaires.

Les tests d'intégration confirment que deux ou plusieurs composants d'application fonctionnent ensemble pour produire un résultat attendu, y compris éventuellement chaque composant requis pour traiter entièrement une requête.

Ces tests plus larges sont utilisés pour tester l'infrastructure et la structure complète de l'application, y compris souvent les composants suivants :

- Base de données
- Système de fichiers
- Appliances réseau
- Pipeline requête-réponse

ASP.NET Core prend en charge les tests d'intégration à l'aide d'une infrastructure de tests unitaires avec un hôte web de test et un serveur de test en mémoire ^[46].

7.2.5 Implémentation des tests

Pour la structure des projets de tests on considère un dossier de tests pour chaque microservices, constitué d'un projet pour les tests unitaires et un pour les tests d'intégration comme illustré dans la

figure 7.1.

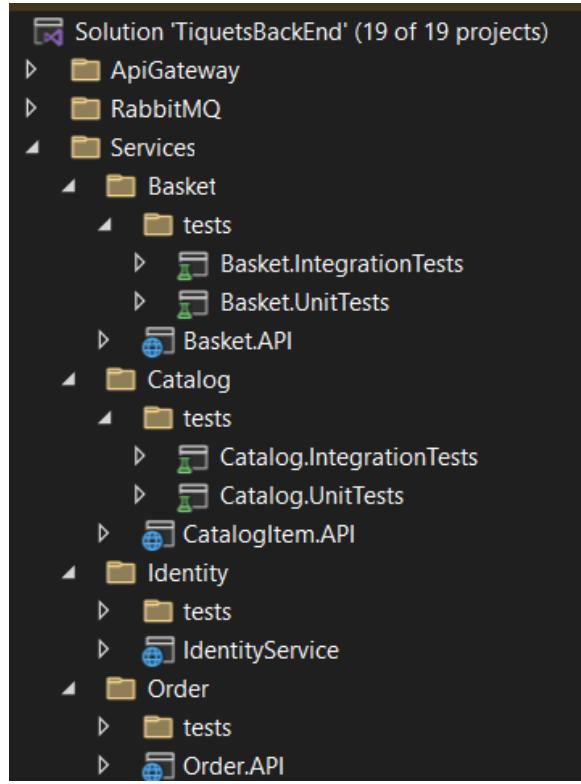


FIGURE 7.1 : la structure des projets

Nous avons utilisé deux attributs de test Fact et Theory.

L'attribut [Fact] est utilisé pour marquer une méthode comme un test qui ne prend aucun paramètre.

Il est utilisé pour des tests simples qui n'ont pas besoin de données d'entrée variées.

L'attribut [Theory] représente une suite de tests qui exécutent le même code, mais qui ont des arguments d'entrée différents.

En total, nous avons développé 33 cas de test qui ont étaient tous compiler avec succès, comme l'illustre la figure 7.2, indiquant ainsi la validité des méthodes développer.

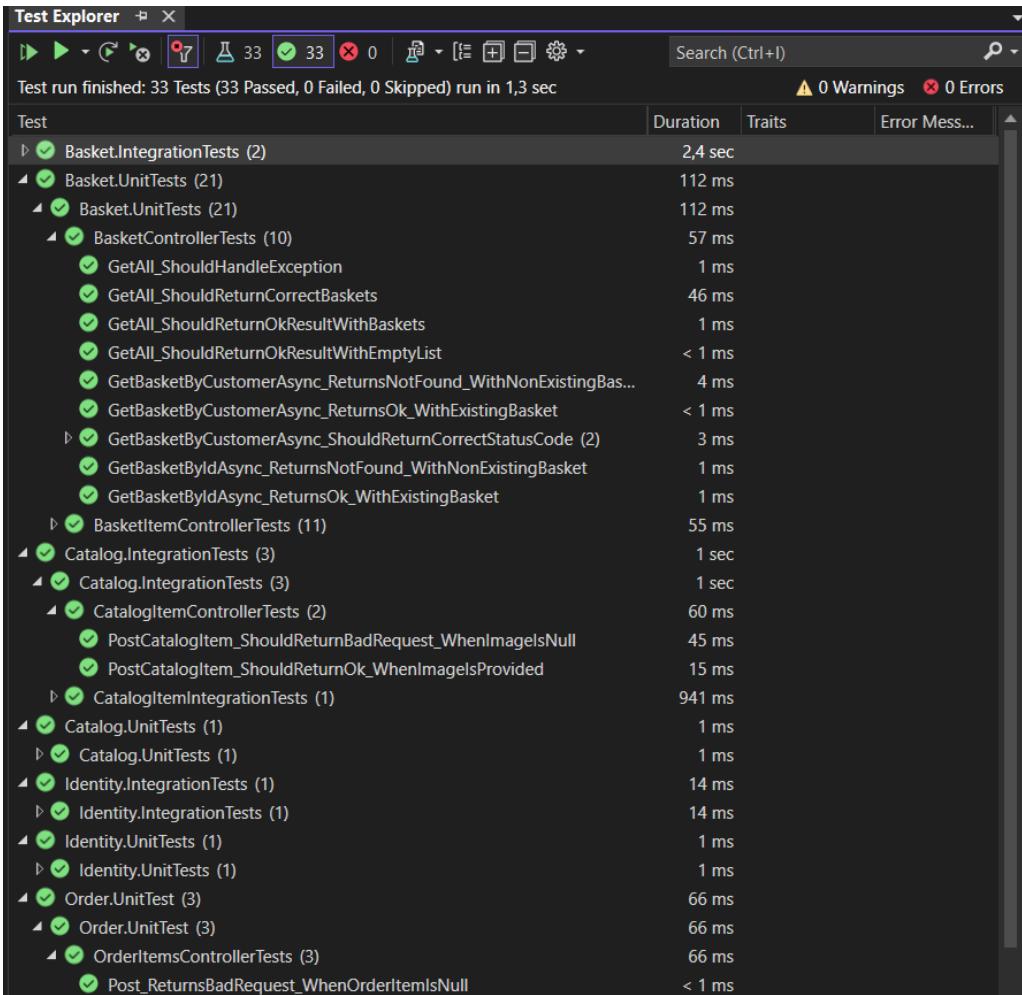


FIGURE 7.2 : Validation des méthodes développées par les tests

7.3 Sprint 11 : Déploiement de l'application

7.3.1 Objectifs de Sprint 11

L'objectif de ce sprint est de garantir une mise en production fluide et stable du Backend de notre application. Nous explorerons l'infrastructure mise en place, les outils utilisés tels que Docker Compose et Jenkins, ainsi que les différentes étapes du pipeline CI/CD.

7.3.2 Environnement de Déploiement

L'environnement de déploiement a été soigneusement configuré pour garantir la stabilité et la performance de l'application. Les composants essentiels de cette configuration incluent :

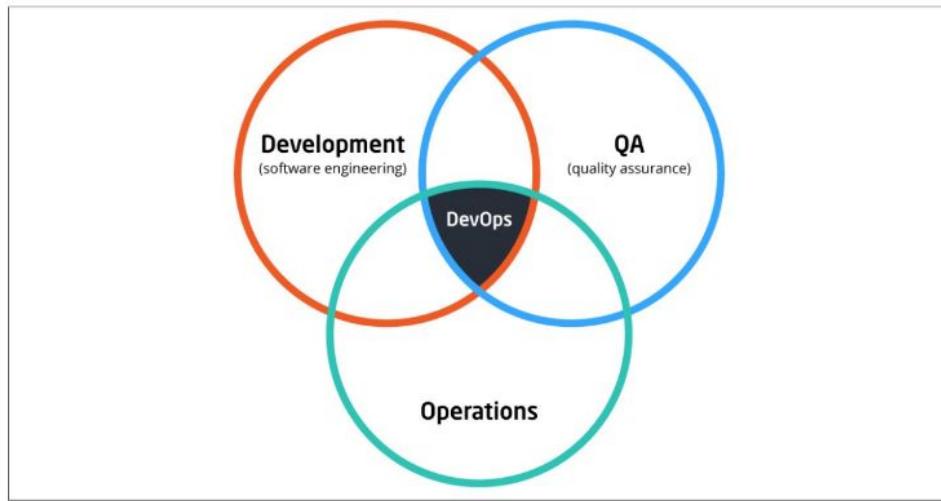
- **Docker & Docker Compose** : Docker a été utilisé pour conteneuriser les microservices, tandis que Docker Compose a orchestré l'ensemble des services définis dans le fichier `docker-compose.yml`.
- **Docker Hub** : Les images Docker des microservices sont stockées sur Docker Hub. Cela permet

de partager et de déployer les images à partir d'un registre centralisé, facilitant l'accès aux versions spécifiques des images pour différents environnements de déploiement.

- **Docker Desktop** : Docker Desktop est utilisé pour visualiser et gérer les conteneurs en cours d'exécution et les images stockées localement. Il fournit une interface graphique qui permet de surveiller l'état des services et des conteneurs, ainsi que d'interagir avec eux de manière plus intuitive.
- **Jenkins** : Jenkins a été configuré pour exécuter la pipeline CI/CD, automatisant les tâches de build, de test, et de déploiement.
- **Conteneur Privilégié Jenkins** : Jenkins a été déployé dans un conteneur Docker configuré en mode privilégié pour permettre l'exécution des commandes Docker directement à partir de Jenkins.
- **.NET 8 et Aspire** : Les microservices ont été développés en .NET 8 et Aspire. Les dépendances nécessaires ont été installées dans l'image Docker Jenkins pour garantir la compatibilité lors des builds et des tests.
- **SQL Server 2022** : L'image Docker de SQL Server 2022 a été utilisée pour conteneurer les bases de données, facilitant ainsi la gestion et l'accès aux données nécessaires pour les microservices.
- **GitHub** : Le code source est hébergé sur GitHub. Jenkins utilise ce dépôt pour récupérer le code et déclencher les builds automatiques à chaque commit.

7.3.3 Introduction sur L'approche DevOps

DevOps est une approche qui réunit les équipes de développement et d'exploitation pour améliorer le processus de développement logiciel, comme le montre la figure 7.3.3. En combinant l'automatisation, la collaboration et l'intégration continue, DevOps vise à accélérer la livraison des logiciels, à améliorer leur qualité et à favoriser une meilleure collaboration entre les équipes. En adoptant DevOps, les entreprises peuvent bénéficier d'une plus grande agilité, d'une réduction des erreurs humaines et d'une meilleure efficacité dans le déploiement des applications. Cela permet de répondre aux exigences croissantes du marché en fournissant des logiciels de haute qualité de manière plus rapide et plus fiable [47].

**FIGURE 7.3 :** L'approche DevOps

Les avantages de DevOps

DevOps joue un rôle essentiel dans le cycle de vie du développement logiciel en apportant plusieurs avantages et en résolvant des défis fréquemment rencontrés par les équipes de développement.

Voici quelques points clés sur l'importance de DevOps :

- **Collaboration améliorée** : DevOps favorise une collaboration étroite entre les équipes de développement, d'exploitation et de qualité. En encourageant la communication et le partage des connaissances, il permet une meilleure compréhension des besoins et des contraintes de chaque équipe. Cette collaboration renforcée conduit à des décisions plus éclairées, à des processus plus efficaces et à une meilleure résolution des problèmes.
- **Livraison continue et rapide** : DevOps vise à automatiser les processus de développement, de test et de déploiement. Cela permet une livraison continue et rapide des logiciels, en réduisant les délais entre les phases de développement et de production. Les équipes peuvent ainsi déployer des versions de logiciels plus fréquemment et de manière plus prévisible, ce qui répond aux attentes des utilisateurs et aux exigences du marché.
- **Qualité et fiabilité** : En intégrant des tests automatisés et des contrôles de qualité tout au long du processus de développement, DevOps améliore la qualité du code et réduit les risques d'erreurs. Les tests continus permettent de détecter les problèmes plus tôt, facilitant ainsi leur résolution et réduisant les coûts liés aux corrections de bugs en aval. La mise en place de mécanismes de surveillance et de rétroaction permet également d'identifier rapidement les problèmes de performance ou de stabilité, garantissant ainsi la fiabilité des applications.
- **Agilité et flexibilité** : DevOps favorise l'adoption de pratiques agiles, telles que la planification

itérative, la livraison itérative et l'adaptation continue. Cela permet aux équipes de répondre plus rapidement aux changements, de s'adapter aux besoins changeants des utilisateurs et du marché, et de prendre des décisions basées sur des données en temps réel. L'agilité et la flexibilité accrues facilitent également l'innovation et la création de nouveaux produits ou fonctionnalités.

- **Gestion efficace des infrastructures :** DevOps permet de gérer les infrastructures de manière plus efficace grâce à l'utilisation de techniques telles que l'infrastructure en tant que code (IaC) et l'orchestration des conteneurs. Cela permet de provisionner et de gérer les ressources d'infrastructure de manière automatisée et reproductible, réduisant ainsi les erreurs liées à la configuration manuelle et améliorant la scalabilité et la résilience des applications.

La chaîne CI/CD

Le DevOps repose sur la chaîne CI/CD (Continuous Integration/Continuous Delivery) qui Constitue l'épine dorsale de la méthodologie DevOps. Cette chaîne permet d'automatiser les processus de développement, de test et de déploiement, favorisant ainsi une livraison continue des logiciels. La CI/CD commence par la Continuous Integration (Intégration Continue) où les développeurs intègrent régulièrement leur code au sein d'un référentiel partagé. À chaque intégration, une suite de tests automatisés est exécutée pour vérifier l'intégrité du code et détecter d'éventuelles erreurs le plus tôt possible. Cela permet d'identifier rapidement les conflits de fusion et de garantir une base de code stable et fonctionnelle.

Une fois l'intégration réussie, la chaîne CI/CD se poursuit avec la Continuous Delivery (Livraison Continue) qui vise à automatiser le déploiement des logiciels dans des environnements de production. À chaque modification du code, l'application est construite, testée et déployée automatiquement dans un environnement de pré-production pour une validation supplémentaire. Si tous les tests sont réussis, l'application est alors déployée en production de manière automatique et fiable.

La chaîne CI/CD permet de réduire les délais entre les développements, les tests et les déploiements, offrant ainsi des avantages significatifs.

7.3.4 Analyse Comparative : Docker Compose vs Docker file

Pour choisir les outils les plus adaptés à la gestion des microservices dans notre projet, une comparaison entre Docker Compose et Dockerfile est présentée dans le tableau 7.2.

Critère	Dockerfile	Docker Compose
Objectif Principal	Créer une image Docker individuelle.	Orchestrer plusieurs conteneurs Docker.
Gestion des Services	Gère la construction d'une seule image Docker.	Gère plusieurs conteneurs et leur interconnexion.
Configuration de Réseaux	Pas de gestion native des réseaux.	Permet la configuration des réseaux entre conteneurs.
Volumes et Stockage	Nécessite des configurations manuelles pour les volumes.	Simplifie la gestion des volumes pour les conteneurs.
Environnement Multi-Services	Ne gère qu'une image Docker à la fois.	Idéal pour des environnements complexes avec plusieurs services.
Déploiement et Orchestration	Pas de fonctionnalités d'orchestration.	Orchestration intégrée pour démarrer, arrêter et gérer les conteneurs.
Facilité d'Utilisation	Configuration plus complexe pour des environnements multi-services.	Fichier "docker-compose.yml" centralise et simplifie la configuration.
Développement Local	Convient pour le développement d'une image unique.	Facilite le développement local en simulant des environnements de production.
Scalabilité	Requiert des configurations supplémentaires pour la scalabilité.	Supporte la scalabilité et la gestion des services multiples de manière native.

TABLEAU 7.2 : Comparaison entre Dockerfile et Docker Compose

Docker Compose a été choisi pour orchestrer les micro-services dans notre environnement de déploiement en raison de ses avantages significatifs pour la gestion des environnements complexes.

Contrairement à Docker file, qui est limité à la création d'images individuelles, Docker Compose offre une solution complète pour l'orchestration de plusieurs conteneurs, la gestion des réseaux et des volumes, ainsi que la simplification de la configuration. Cela rend Docker Compose particulièrement adapté à notre architecture de micro-services.

7.3.5 La processus du pipeline CI/CD Jenkins

Les étapes de notre pipeline CI/CD est conçue pour automatiser les processus de build, de test, et de déploiement, garantissant une livraison continue et sécurisée des modifications de code.

La figure 7.4 représente les étapes de pipeline CI/CD

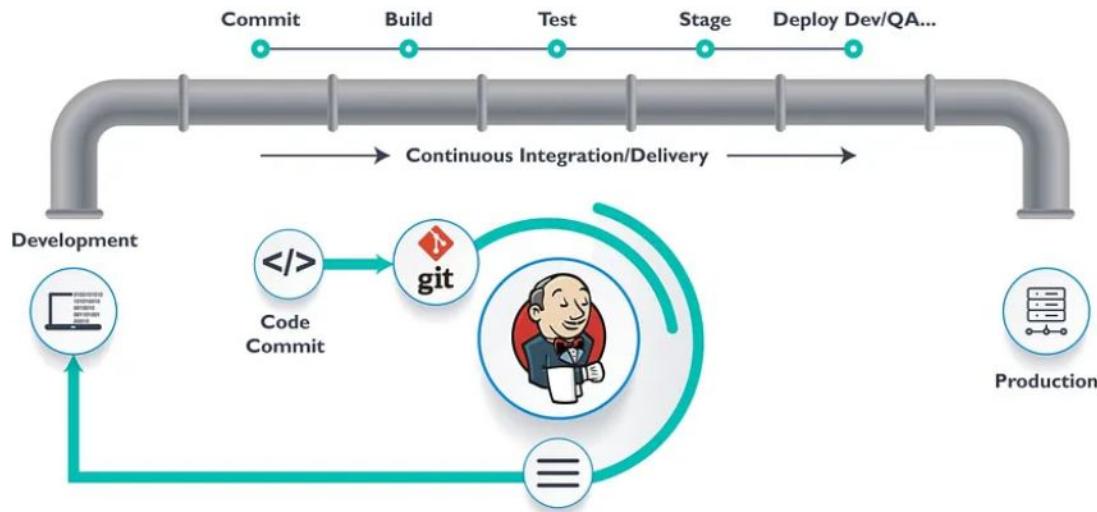


FIGURE 7.4 : les étapes de pipeline CI/CD [48]

La figure 7.5 représente l'architecture de la partie CD (Continuous Delivery) du pipeline qui utilise Docker Compose pour orchestrer les services de notre application microservices. Lorsqu'un développeur pousse du code vers un dépôt GitHub, Jenkins, qui surveille ce dépôt, déclenche un processus de build. Jenkins construit chaque microservice dans un conteneur Docker et crée des images Docker pour chaque service. Docker Compose est ensuite utilisé pour lancer et exécuter tous les services définis dans l'application, permettant ainsi de déployer l'ensemble de l'environnement de l'application, garantissant que les services fonctionnent correctement ensemble avant leur déploiement en production.

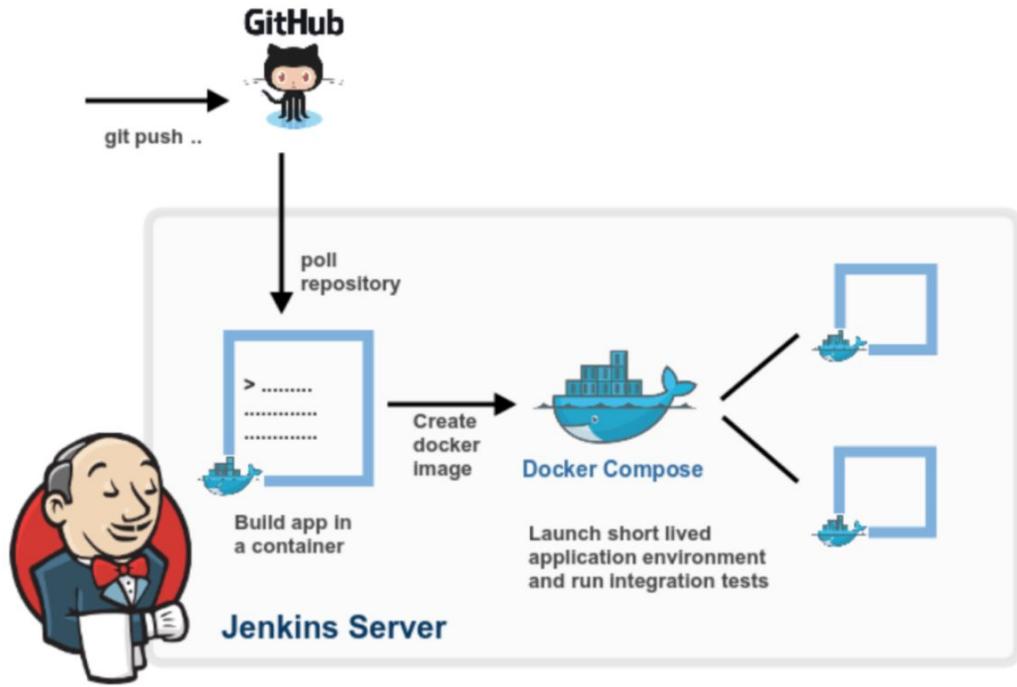


FIGURE 7.5 : CI/CD avec le docker compose [49]

7.3.6 Mise en place du Pipeline CI/CD

Les étapes suivantes décrivent le pipeline CI/CD mis en place et les illustrations associées pour illustrer chaque étape :

Etape 1 : Checkout Stage

Objectif : Cloner le dépôt Git et vérifier la branche AppDeployment.

Actions : Utilise les informations d’identification pour accéder au dépôt et récupérer le code source.

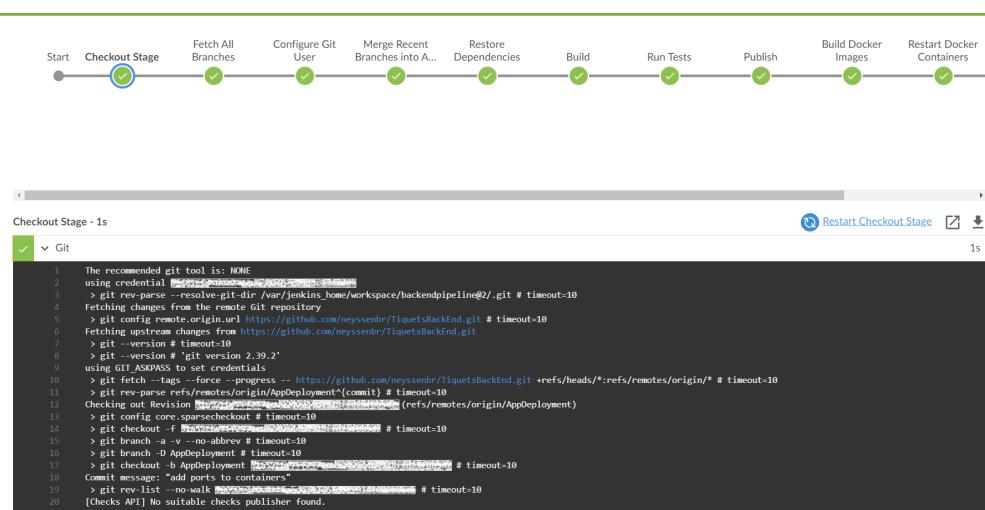


FIGURE 7.6 : Cloner le dépôt Git

Etape 2 : Fetch All Branches

Objectif : Mettre à jour les références des branches distantes.

Actions : Configure les informations d'identification Git, met à jour les références distantes, et récupère toutes les branches.



```
Fetch All Branches - 2s
✓ ✓ git remote set-url origin https://$...@github.com/neyssenbr/TicketsBackEnd.git - Shell Script
  1 + git remote set-url origin https://$...@github.com/neyssenbr/TicketsBackEnd.git
✓ ✓ git fetch --all - Shell Script
  1 + git fetch --all
```

FIGURE 7.7 : Fetch All Branches

Etape 3 : Configure Git User

Objectif : Configurer les informations d'utilisateur Git pour les commit.

Actions : Définir l'adresse email et le nom d'utilisateur pour Git.



```
Configure Git User - <1s
✓ ✓ git config user.email "neyssen98@gmail.com" - Shell Script
  1 + git config user.email neyssen98@gmail.com
✓ ✓ git config user.name "neyssenbr" - Shell Script
  1 + git config user.name neyssenbr
```

FIGURE 7.8 : Configure Git User

Merge Recent Branches into AppDeployment

Objectif : Fusionner les branches récentes dans la branche AppDeployment.

Actions : Identifier les branches modifiées récemment, essayer de les fusionner dans AppDeployment, et gérer les conflits de fusion.

Illustration :

- Succès de la Fusion : La branche a été fusionnée avec succès dans AppDeployment.

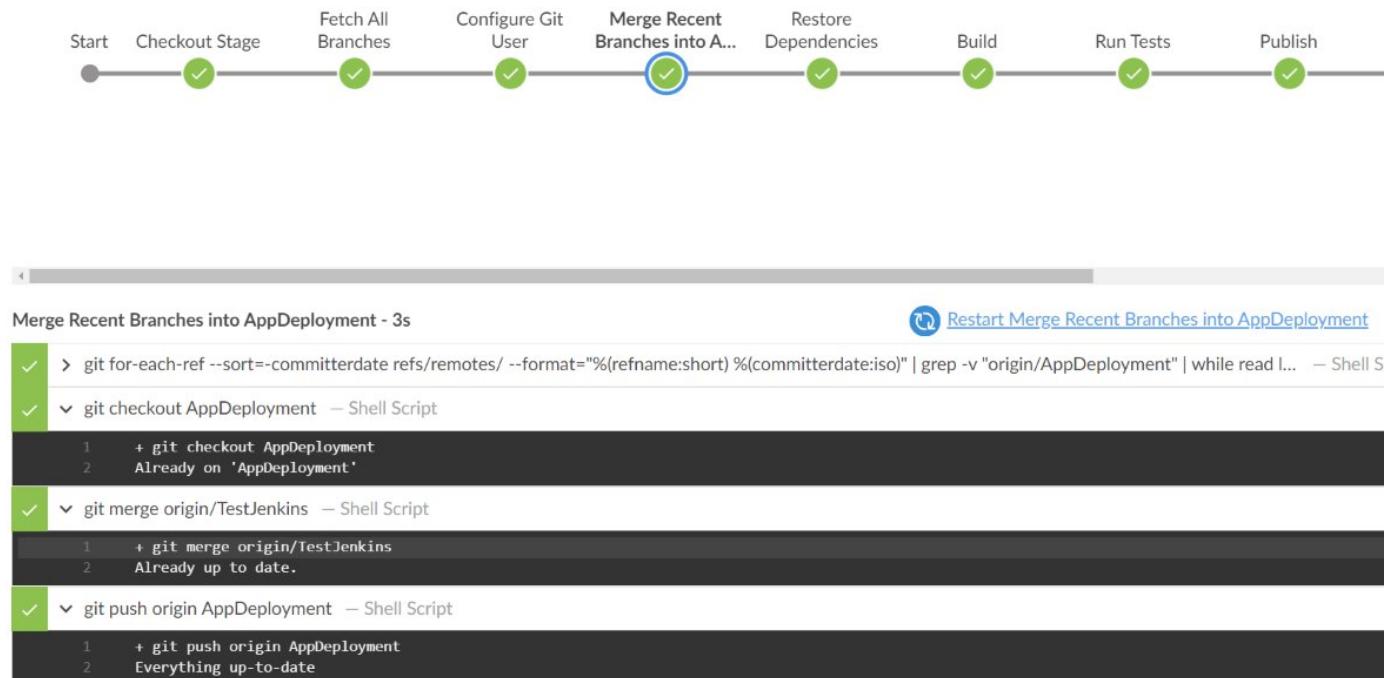


FIGURE 7.9 : Succès de la Fusion

- Etape 4 : Conflit de Fusion : Une tentative de fusion a échoué en raison de conflits entre les branches.

The screenshot shows a Jenkins pipeline stage named 'Merge Recent Branches into AppDeployment - 1s'. The stage has the following steps:

- git for-each-ref --sort=-committerdate refs/remotes/ --format="%{refname:short} %(committerdate:iso)" | grep -v "origin/AppDeployment" | while read I... — Shell Script
- git checkout AppDeployment — Shell Script
- git merge origin/master — Shell Script
 - + git merge origin/master
 - Auto-merging Basket.API/Basket.API.csproj
 - Auto-merging Basket.UnitTests/Basket.UnitTests.csproj
 - Auto-merging Basket.UnitTests/BasketControllerTests.cs
 - Auto-merging Catalog.UnitTests/Catalog.UnitTests.csproj
 - Auto-merging EventCatalog.API/CatalogItem.API.csproj
 - CONFLICT (rename/delete): Order.UnitTesting.UnitTest1.cs renamed to Identity.UnitTesting.UnitTest1.cs in origin/master, but deleted in HEAD.
 - CONFLICT (modify/delete): Identity.UnitTesting.UnitTest1.cs deleted in HEAD and modified in origin/master. Version origin/master of Identity.UnitTesting.UnitTest1.cs left in tree.
 - CONFLICT (modify/delete): Order.UnitTesting.Order.UnitTest.csproj deleted in HEAD and modified in origin/master. Version origin/master of Order.UnitTesting.Order.UnitTest.csproj left in tree.
 - Auto-merging TiquetsBackEnd.sln
 - CONFLICT (content): Merge conflict in TiquetsBackEnd.sln
 - Automatic merge failed; fix conflicts and then commit the result.
- Merge conflict detected with branch master. Please resolve conflicts manually. — Print Message
 - Merge conflict detected with branch master. Please resolve conflicts manually.
- git merge --abort — Shell Script
 - + git merge --abort
- Branches with merge conflicts: master — Print Message
 - Branches with merge conflicts: master

A red circle with a minus sign is placed around the 'git merge origin/master' step, indicating it failed. The entire pipeline stage is marked with a red minus sign.

FIGURE 7.10 : Conflit de Fusion

Etape 5 : Restore Dépendances

Objectif : Restaurer les dépendances du projet.

Actions Exécuter la commande “dotnet restore” pour préparer les dépendances nécessaires.

Stage 'Restore Dependencies'

- ⌚ Started 21 h ago
- ⌚ Queued 0 ms
- ⌚ Took 2.8 s
- ℹ Success
- [View as plain text](#)

⌚ **dotnet restore "TiquetsBackEnd.sln"** 2.7 s

Shell Script

```

0 + dotnet restore TiquetsBackEnd.sln
1 Determining projects to restore...
2 /var/jenkins_home/workspace/backendpipeline/EventCatalog.API/CatalogItem.API.csproj : warning NU1701: Package 'Intercom.Dotnet.Client 2.1.1' was restored using '.NETFramework,Version=v4.6.1, .NETFramework,Version=v4.6.2, .NETFramework,Version=v4.7, .NETFramework,Version=v4.7.1, .NETFramework,Version=v4.7.2, .NETFramework,Version=v4.8, .NETFramework,Version=v4.8.1' instead of the project target framework 'net8.0'. This package may not be fully compatible with your project.
[ /var/jenkins_home/workspace/backendpipeline/TiquetsBackEnd.sln]
3 /var/jenkins_home/workspace/backendpipeline/Order.API/Order.API.csproj : warning NU1903: Package 'MimeKit' 4.6.0 has a known high severity vulnerability, https://github.com/advisories/GHSA-gmc6-fwg3-75m5
[ /var/jenkins_home/workspace/backendpipeline/TiquetsBackEnd.sln]
```

FIGURE 7.11 : Restore Dépendances

Étape 6 : le Construire ‘ Build ‘ du projet .NET

Objectif : Construire le projet .NET pour générer les fichiers exécutables et les bibliothèques nécessaires.

Actions : La commande 'dotnet build "TiquetsBackEnd.sln" --configuration Release ' compile tous les micro-services dans la solution TiquetsBackEnd.sln.

Stage 'Build'

- ⌚ Started 21 h ago
- ⌚ Queued 0 ms
- ⌚ Took 7.1 s
- ℹ Success
- [View as plain text](#)

⌚ **dotnet build "TiquetsBackEnd.sln" --configuration Release** 7.1 s

Shell Script

```

/var/jenkins_home/workspace/backendpipeline/IdentityService/bin/Release/net8.0/IdentityService.dll
32 CatalogItem.API ->
/var/jenkins_home/workspace/backendpipeline/EventCatalog.API/bin/Release/net8.0/CatalogItem.API.dll
33 Basket.API -> /var/jenkins_home/workspace/backendpipeline/Basket.API/bin/Release/net8.0/Basket.API.dll
34 Basket.UnitTests ->
/var/jenkins_home/workspace/backendpipeline/Basket.UnitTests/bin/Release/net8.0/Basket.UnitTests.dll
35 TiquetsBackEnd.AppHost ->
/var/jenkins_home/workspace/backendpipeline/TiquetsBackEnd.AppHost/bin/Release/net8.0/TiquetsBackEnd.AppHost.dll
36
37 Build succeeded.
38
```

FIGURE 7.12 : Construire ‘Build’ du projet

Étape 7 : exécution des Tests

Objectif : Vérifier la fonctionnalité et l'intégrité du code en exécutant les tests définis dans la solution TiquetsBackEnd.sln.

Actions : La commande dotnet test "TiquetsBackEnd.sln" --no-restore est utilisée pour lancer les tests. L'option --no-restore indique que les dépendances ne seront pas restaurées, car elles ont déjà été restaurées dans une étape précédente.

```

Stage 'Run Tests'
① Started 21 h ago
② Queued 0 ms
③ Took 6.7 s
④ Success
⑤ View as plain text

dotnet test "TiquetsBackEnd.sln" --no-restore
Shell Script
6.6 s  ↻ ⌂ ^

0 + dotnet test TiquetsBackEnd.sln --no-restore
1 Catalog.UnitTests -> /var/jenkins_home/workspace/backendlipeline/Catalog.UnitTests/bin/Debug/net8.0/Catalog.UnitTests.dll
2 Test run for /var/jenkins_home/workspace/backendlipeline/Catalog.UnitTests/bin/Debug/net8.0/Catalog.UnitTests.dll (.NETCoreApp,Version=v8.0)
3 Microsoft (R) Test Execution Command Line Tool Version 17.10.0 (x64)
4 Copyright (c) Microsoft Corporation. All rights reserved.
5
6 Starting test execution, please wait...
7 A total of 1 test files matched the specified pattern.
8
9 Passed! - Failed: 0, Passed: 1, Skipped: 0, Total: 1, Duration: < 1 ms - Catalog.UnitTests.dll (net8.0)
10 TiquetsBackEnd.ServiceDefaults -> /var/jenkins_home/workspace/backendlipeline/TiquetsBackEnd.ServiceDefaults/bin/Debug/net8.0/TiquetsBackEnd.ServiceDefaults.dll
11 RabbitMQ -> /var/jenkins_home/workspace/backendlipeline/RabbitMQ/bin/Debug/net8.0/RabbitMQ.dll
12 Basket.API -> /var/jenkins_home/workspace/backendlipeline/Basket.API/bin/Debug/net8.0/Basket.API.dll
13 Basket.UnitTests -> /var/jenkins_home/workspace/backendlipeline/Basket.UnitTests/bin/Debug/net8.0/Basket.UnitTests.dll
14 Test run for /var/jenkins_home/workspace/backendlipeline/Basket.UnitTests/bin/Debug/net8.0/Basket.UnitTests.dll (.NETCoreApp,Version=v8.0)
15 Microsoft (R) Test Execution Command Line Tool Version 17.10.0 (x64)
16 Copyright (c) Microsoft Corporation. All rights reserved.
17
18 Starting test execution, please wait...
19 A total of 1 test files matched the specified pattern.
20
21 Passed! - Failed: 0, Passed: 9, Skipped: 0, Total: 9, Duration: 962 ms - Basket.UnitTests.dll (net8.0)

```

FIGURE 7.13 : Exécution des Tests

Étape 8 : Construction des Images Docker

Objectif : Construire les images Docker pour les micro-services spécifiés dans le fichier 'docker-compose.yml', afin de préparer les conteneurs pour le déploiement et l'exécution.

Actions : La commande 'docker-compose build' est exécutée pour construire les images Docker des différents services définis dans le fichier de configuration.

Illustration : La figure 7.14 représente les logs de la construction des images Docker.

```

Stage 'Build Docker Images'
⌚ Started 21 h ago
🕒 Queued 0 ms
🕒 Took 37 mn
💡 Success
🔗 View as plain text

docker-compose build
Shell Script
37 mn ⌂ ↻ ^

41 #13 DONE 0.2s
42
43 #20 [build 5/10] COPY [TiquetsBackEnd.ServiceDefaults/TiquetsBackEnd.ServiceDefaults.csproj, TiquetsBackEnd.ServiceDefaults/]
44 #20 sha256:dc5e0f98424a86f668862e2aa18ec52082caa4a078c0b3a621f6ec754db1196
45 #20 DONE 0.3s
46
47 #7 [build 6/10] RUN dotnet workload install aspire
48 #7 sha256:686682a2b8b9971afcd80d92edcaa0c892480d41e002b767092ad003bf6618f4
49 #7 0.940
50 #7 3.629 Skipping NuGet package signature verification.
51 #7 4.243 Installing workload manifest microsoft.net.sdk.android version 34.0.113...
52 #7 4.699 Installing workload manifest microsoft.net.sdk.ios version 17.5.8020...
53 #7 5.140 Installing workload manifest microsoft.net.sdk.maccatalyst version 17.5.8020...
54 #7 5.576 Installing workload manifest microsoft.net.sdk.macos version 14.5.8020...
55 #7 6.005 Installing workload manifest microsoft.net.sdk.maui version 8.0.72...
56 #7 6.459 Installing workload manifest microsoft.net.sdk.tvos version 17.5.8020...
57 #7 7.618 Installing workload manifest microsoft.net.sdk.aspire version 8.1.0...
58 #7 8.780 Installing pack Aspire.Hosting.Sdk version 8.1.0...

```

FIGURE 7.14 : logs de construction des images Docker

La figure 7.15 représente les images créées au docker desktop.

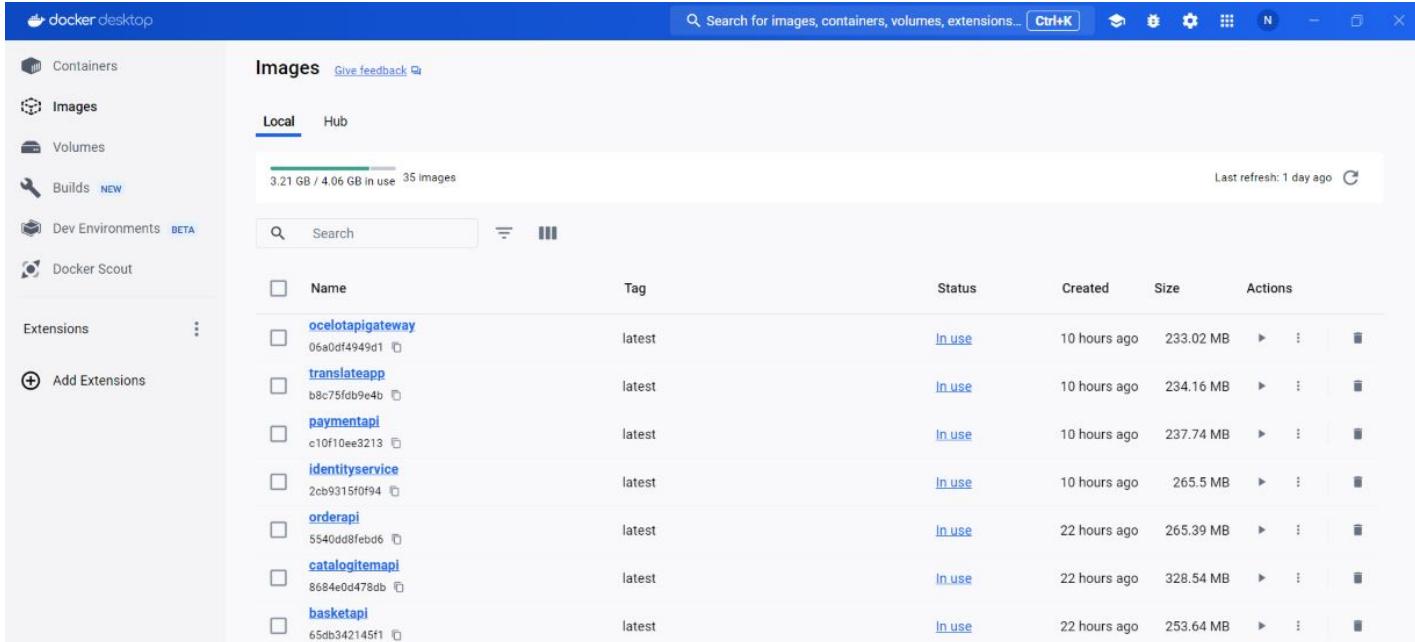


FIGURE 7.15 : Images docker en local

Étape 10 : Poussée les Images au DockerHub

Objectifs : Pousser les images Docker construites vers Docker Hub afin de les rendre disponibles pour le déploiement et le partage.

La commande 'docker push' est exécutée pour envoyer les images Docker des différents micro-services vers Docker Hub. Cette étape assure que les images sont stockées dans un registre centralisé, facilitant ainsi leur déploiement et leur utilisation dans divers environnements.

Illustration : La figure 7.16 montre les logs de la poussée des images Docker.

```

Push Docker Images to Docker Hub - 35m 47s
✓ > echo dckr_pat_R4Whombiac1cQ7j0RbValawvu | docker login -u neyssen -password $stdin - Shell Script 2s
✓ > docker tag identityservice:latest neyssen/identityservice:v1.0 - Shell Script <1s
✓ > docker push neyssen/identityservice:v1.0 - Shell Script 4m 16s
✓ > docker tag paymentapi:latest neyssen/paymentapi:v1.0 - Shell Script <1s
✓ > docker push neyssen/paymentapi:v1.0 - Shell Script 1m 36s
✓ > docker tag translateapp:latest neyssen/translateapp:v1.0 - Shell Script <1s
✓ > docker push neyssen/translateapp:v1.0 - Shell Script 1m 19s
✓ > docker tag ocelotapigateway:latest neyssen/ocelotapigateway:v3.0 - Shell Script <1s
✓ > docker push neyssen/ocelotapigateway:v3.0 - Shell Script 4s
✓ > docker tag basketapi:latest neyssen/basketapi:v1.0 - Shell Script <1s
✓ > docker push neyssen/basketapi:v1.0 - Shell Script 2m 42s
✓ > docker tag catalogitemapi:latest neyssen/catalogitemapi:v1.0 - Shell Script <1s
✓ > docker push neyssen/catalogitemapi:v1.0 - Shell Script 21m 43s
✓ > docker tag orderapi:latest neyssen/orderapi:v1.0 - Shell Script <1s
✓ > docker push neyssen/orderapi:v1.0 - Shell Script 4m 2s

```

FIGURE 7.16 : logs de 'docker push'

La figure 7.17 illustre les images disponibles dans Docker Hub.

Repository	Last pushed	Stars	Downloads	Status
neyssen / orderapi	5 minutes ago	0	4	Public
neyssen / catalogitemapi	9 minutes ago	0	1	Public
neyssen / basketapi	30 minutes ago	0	3	Public
neyssen / ocelotapigateway	33 minutes ago	0	8	Public
neyssen / translateapp	33 minutes ago	0	3	Public
neyssen / paymentapi	35 minutes ago	0	3	Public
neyssen / identityservice	36 minutes ago	0	2	Public

FIGURE 7.17 : Images dans DockerHub

Étape 11 :Création des Conteneurs Docker

Objectif : Démarrer les conteneurs Docker en utilisant les images créées lors de l'étape de 'Construction des Images Docker' afin de déployer les services dans un environnement d'exécution.

Action : La commande 'docker-compose up' est utilisée pour créer et démarrer les conteneurs basés

sur les images définies dans le fichier docker-compose.yml.

Illustration : La figure 7.18 montre les conteneurs en cours d'exécution après l'exécution de la commande docker-compose up.

```

    docker-compose up -d --no-recreate
    Shell Script
    0 + docker-compose up -d --no-recreate
    1 Creating network "backendpipeline_mynetwork" with driver "bridge"
    2 Creating backendpipeline_translateapp_1 ...
    3 Creating backendpipeline_mssqlserver_1 ...
    4 Creating backendpipeline_payment.api_1 ...
    5 Creating backendpipeline_ocelotapigateway_1 ...
    6 Creating backendpipeline_mssqlserver_1 ... done
    7 Creating backendpipeline_catalogitem.api_1 ...
    8 Creating backendpipeline_identityservice_1 ...
    9 Creating backendpipeline_order.api_1 ...
    10 Creating backendpipeline_basket.api_1 ...
    11 Creating backendpipeline_ocelotapigateway_1 ... done
    12 Creating backendpipeline_translateapp_1 ... done
    13 Creating backendpipeline_payment.api_1 ... done
    14 Creating backendpipeline_identityservice_1 ... done
    15 Creating backendpipeline_order.api_1 ... done
  
```

FIGURE 7.18 : Logs de docker-compose up

La figure 3.19 présente l'interface de Docker Desktop après le déploiement avec Docker Compose. Elle montre les différents services actifs, y compris la base de données MSSQL. Les détails de chaque service, tels que les ports exposés et les états des conteneurs, sont également visibles.

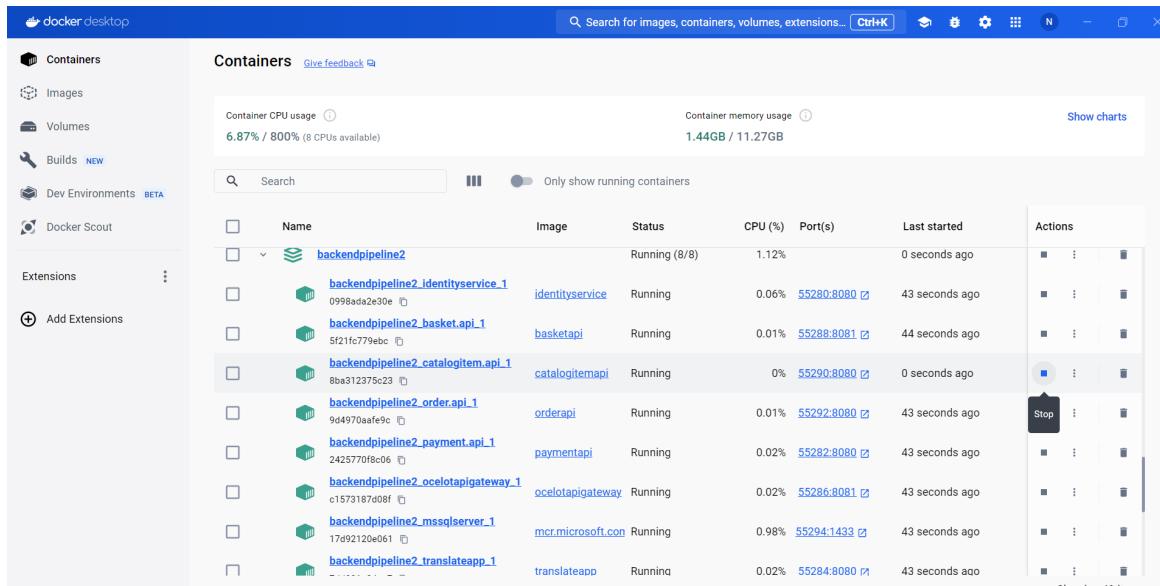


FIGURE 7.19 : docker compose

Artefacts du Pipeline CI/CD

Après la construction et le test de l'application dans le pipeline, les artefacts principaux sont les fichiers publiés. Dans notre pipeline, ces artefacts sont générés à l'étape de publication (Publish). Les fichiers publiés sont stockés dans le répertoire './publish', et ce répertoire est archivé en tant qu'un

artefact. Cela permet de garantir que les fichiers nécessaires pour le déploiement sont disponibles et peuvent être récupérés pour des déploiements ultérieurs ou pour une vérification.

Objectif : Assurer que les fichiers nécessaires au déploiement sont disponibles pour être utilisés dans des environnements de production ou de staging.

Contenu : Les fichiers de sortie de la publication .NET, incluant les exécutables et les dépendances nécessaires pour exécuter l'application.

Illustration La figure 7.20 montre les fichiers archivés à partir du répertoire './publish' après la publication.

NOM	TAILLE
pipeline.log	-
publish/af/Humanizer.resources.dll	16.1 KB
publish/appsettings.Development.json	119 bytes
publish/appsettings.json	142 bytes
publish/ar/Humanizer.resources.dll	21.1 KB
publish/az/Humanizer.resources.dll	16.1 KB
publish/Azure.Core.dll	368.9 KB
publish/Azure.Identity.dll	327 KB
publish/Basket.API	70.7 KB
publish/Basket.API.deps.json	92.3 KB
publish/Basket.API.dll	55.5 KB
publish/Basket.API.pdb	35.6 KB

FIGURE 7.20 : Artefacts du Pipeline

Conclusion

Cette release a permis de valider l'application à travers des tests unitaires et d'intégration, garantissant la stabilité et la fiabilité du code. Ensuite, le déploiement de l'application a assuré une mise en production fluide, confirmant l'efficacité des processus mis en place et la préparation de l'application pour l'utilisation finale.

Conclusion générale

Ce rapport présente notre projet de fin d'études au sein de Tek-Up University et marque l'aboutissement de notre parcours d'enseignement supérieur. Il nous a permis de mettre en pratique les connaissances acquises tout au long de notre cursus universitaire, notamment durant notre stage chez Sofiatech. Cette entreprise nous a confié le développement d'une application de billetterie en ligne.

En conclusion, notre application de billetterie en ligne constitue une avancée majeure dans l'organisation d'événements. Avec des fonctionnalités telles que la personnalisation des billets, nous avons développé une solution innovante qui répond aux besoins des organisateurs tout en anticipant les défis futurs.

Elle simplifie les processus d'organisation, offrant une expérience utilisateur fluide pour tous, ce qui augmente la satisfaction des participants et le succès des événements.

À l'avenir, nous continuerons à améliorer notre plateforme en intégrant les dernières technologies. La prochaine étape consistera à déployer l'application frontend, suivie de tests rigoureux pour garantir une intégration efficace avec le backend et assurer une expérience utilisateur optimale. Nous prévoyons également d'ajouter un système de recommandation personnalisé et un système de reporting mensuel et annuel pour fournir des informations précieuses sur les ventes et les aspects financiers.

De plus, nous souhaitons diversifier notre offre en incluant des options de transport et d'hébergement, créant ainsi une expérience complète pour les utilisateurs.

En somme, cette application marque le début d'une nouvelle ère pour la billetterie en ligne, promettant de transformer l'expérience des utilisateurs et d'apporter une valeur ajoutée significative au secteur.

Bibliographie

- [9] J. RUMBAUGH, M. BLAHA, W. PREMERLANI, F. EDDY et W. LORENSEN, *Object-Oriented Modeling and Design*. Prentice Hall, 2004.
- [10] G. BOOCHE, J. RUMBAUGH et I. JACOBSON, *The Unified Modeling Language User Guide*. Addison-Wesley, 2005.
- [47] G. KIM, P. DEBOIS, D. WILLIS et J. HUMBLE, *The DevOps Handbook : How to Create World-Class Agility, Reliability, & Security in Technology Organizations*. 2016.

Webographie

- [1] SOFIATECH, *Société d'ingénierie et de solutions informatiques*, [Consulté le : Juillet 24, 2024].
adresse : <https://sofia-technologies.com/fr/>.
- [2] PATHÉ, *Source de l'image*, [Consulté le : Juillet 25, 2024]. adresse : <https://www.pathe.tn/fr>.
- [3] TESKERTI, *Source de l'image*, [Consulté le : Juillet 25, 2024]. adresse : <https://teskerti.tn/>.
- [4] TICKETMASTER, *Source de l'image*, [Consulté le : Juillet 25, 2024]. adresse : <https://www.ticketmaster.fr/fr>.
- [5] EVENTBRITE, *Source de l'image*, [Consulté le : Juillet 25, 2024]. adresse : <https://www.eventbrite.fr/>.
- [6] M. F. DAYS, *Source de l'image*, [Consulté le : Juillet 25, 2024]. adresse : <https://www.myfunnydays.com/>.
- [7] ATlassian, *Framework Scrum*, [Consulté le : 28 Juillet 2024]. adresse : <https://www.atlassian.com/fr/agile/scrum>.
- [8] TULEAP, *Image pour comprendre la méthode Agile Scrum*, [Consulté le : 28 Juillet 2024]. adresse : <https://www.tuleap.org/agile/agile-scrum-in-10-minutes>.
- [11] E. COZAC, *Image pour l'architecture de l'angular*, [Consulté le : 30 Juillet 2024]. adresse : <https://eugeniucozac.medium.com/the-mvc-architecture-in-angular-9eeecd586bb94>.
- [12] N. DOCUMENTATION, *Approche NgRx pour la Gestion Efficace de l'État*, [Consulté le : 5 août 2024]. adresse : <https://ngrx.io/docs>.
- [13] N. DECISION, *Définition pour Git*, [Consulté le : 5 août 2024]. adresse : <https://www.next-decision.fr/wiki/qu-est-ce-que-git>.
- [14] L. MAGIT, *Définition pour GitHub*, [Consulté le : 5 août 2024]. adresse : <https://www.lemagit.fr/definition/GitHub>.
- [15] G. SUPPORT, *Définition pour Google Meet*, [Consulté le : 5 août 2024]. adresse : <https://support.google.com/a/users/answer/9282870?hl=fr>.
- [16] TRELLO, *Définition pour Trello*, [Consulté le : 5 août 2024]. adresse : <https://trello.com/>.
- [17] V. STUDIO, *Définition pour Visual Studio*, [Consulté le : 5 août 2024]. adresse : <https://visualstudio.microsoft.com/>.

- [18] V. S. CODE, *Définition pour Visual Studio Code*, [Consulté le : 5 août 2024]. adresse : <https://code.visualstudio.com/>.
- [19] DOCKER, *Définition pour Docker Desktop*, [Consulté le : 5 août 2024]. adresse : <https://www.docker.com/products/docker-desktop>.
- [20] POSTMAN, *Définition pour Postman*, [Consulté le : 5 août 2024]. adresse : <https://www.postman.com/>.
- [21] MICROSOFT, *Définition pour SQL Server Management Studio*, [Consulté le : 5 août 2024]. adresse : <https://learn.microsoft.com/fr-fr/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16>.
- [22] V. PARADIGM, *Définition pour Visual Paradigm*, [Consulté le : 5 août 2024]. adresse : <https://www.visual-paradigm.com/>.
- [23] RABBITMQ, *RabbitMQ*, [Consulté le : 5 août 2024]. adresse : <https://www.rabbitmq.com/>.
- [24] MAILCATCHER, *MailCatcher*, [Consulté le : 5 août 2024]. adresse : <https://mailcatcher.me/>.
- [25] MICROSOFT, *Langage C#*, [Consulté le : 5 août 2024]. adresse : <https://docs.microsoft.com/fr-fr/dotnet/csharp/>.
- [26] TYPESCRIPT, *Langage TypeScript*, [Consulté le : 5 août 2024]. adresse : <https://www.typescriptlang.org/>.
- [27] W3SCHOOLS, *Langage SQL*, [Consulté le : 5 août 2024]. adresse : <https://www.w3schools.com/sql/>.
- [28] M. W. DOCS, *Langage HTML*, [Consulté le : 5 août 2024]. adresse : <https://developer.mozilla.org/fr/docs/Web/HTML>.
- [29] SASS, *Langage SCSS*, [Consulté le : 5 août 2024]. adresse : <https://sass-lang.com/documentation/scss>.
- [30] A. MATERIAL, *Angular Material*, [Consulté le : 5 août 2024]. adresse : <https://material.angular.io/>.
- [31] MICROSOFT, *ASP.NET Core 8*, [Consulté le : 5 août 2024]. adresse : <https://dotnet.microsoft.com/en-us/apps/aspnet>.
- [32] MICROSOFT, *DotNet Aspire*, [Consulté le : 5 août 2024]. adresse : <https://dotnet.microsoft.com/learn/aspnet/aspnet-aspire>.
- [33] XUNIT, *xUnit*, [Consulté le : 5 août 2024]. adresse : <https://xunit.net/>.

- [34] MICROSOFT, *SQL Server*, [Consulté le : 5 août 2024]. adresse : <https://learn.microsoft.com/en-us/sql/>.
- [35] MICROSOFT, *Introduction to ASP.NET Core Identity*, [Consulté le : 5 août 2024]. adresse : <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity>.
- [36] JWT.IO, *What is a JSON Web Token ?* [Consulté le : 5 août 2024]. adresse : <https://jwt.io/introduction>.
- [37] A. DOCUMENTATION, *Route Guards*, [Consulté le : 5 août 2024]. adresse : <https://v16.angular.io/docs>.
- [38] STRIPE, *Online Payment Processing for Internet Businesses*, [Consulté le : 31 août 2024]. adresse : <https://stripe.com>.
- [39] STRIPE.NET, *Official Stripe Library for .NET*, [Consulté le : 31 août 2024]. adresse : <https://docs.stripe.com/apilang=dotnet>.
- [40] ngx-scanner QR CODE, *npm package for ngx-scanner-qrcode*, [Consulté le : 12 septembre 2024]. adresse : <https://www.npmjs.com/package/ngx-scanner-qrcode>.
- [41] APEXCHARTS, *ApexCharts*, [Consulté le : 2 septembre 2024]. adresse : <https://apexcharts.com/>.
- [42] W3C, *i18n Internationalization and Localization*, [Consulté le : 12 septembre 2024]. adresse : <https://www.w3.org/International/questions/qa-i18n>.
- [43] FILE.IO, *Un service de partage de fichiers temporaire permettant de partager des fichiers avec un lien unique*, [Consulté le : 31 Août 2024]. adresse : <https://www.file.io>.
- [44] ngx TRANSLATE, *Internationalization (i18n) library for Angular*, [Consulté le : 12 septembre 2024]. adresse : <https://dev.to/ivannicksimeonov/ngx-translate-internationalization-i18n-library-for-angular-1d4e>.
- [45] YOGOSHA, *Définition pour les tests unitaires*, [Consulté le : 15 septembre 2024]. adresse : <https://yogosha.com/fr/blog/test-unitaire/#:~:text=L'int%C3%A9gration%20des%20tests%20unitaires,-Confiance%20dans%20le&text=V%C3%A9rifier%20la%20fonctionnalit%C3%A9%20Les%20tests,de%20logique%20ou%20de%20calcul.>
- [46] MICROSOFT, *Tests d'intégration dans ASP.NET Core*, [Consulté le : 20 septembre 2024]. adresse : <https://learn.microsoft.com/fr-fr/aspnet/core/test/integration-tests?view=aspnetcore-8.0&source=recommendations>.
- [48] F. KUZHAN, *Image étapes de pipeline CI/CD*, [Consulté le : 24 septembre 2024]. adresse : <https://medium.com/@faisalkuzhan/day-37-90-jenkins-pipeline-4936c3c557eb>.

- [49] R. RINCON, *Image étapes de pipeline CD avec Docker Compose*, [Consulté le : 24 septembre 2024]. adresse : <https://fr.linkedin.com/pulse/d%C3%A9clencher-un-pipeline-jenkins-avec-github-rony-rincon-qhove>.
- [50] JAVATPOINT, *Image pour l'architecture de l'angular*, [Consulté le : 5 août 2024]. adresse : <https://www.javatpoint.com/angularjs-mvc-architecture>.
- [51] N. DOCUMENTATION, *Image pour l'architecture de l'angular*, [Consulté le : 5 août 2024]. adresse : <https://ngrx.io/guide/store>.
- [52] ANGULAR, *Angular 16*, [Consulté le : 5 août 2024]. adresse : <https://angular.io/>.

Annexes

Résumé

Le but de ce document est de fournir un résumé du travail effectué dans le cadre de mon projet de fin d'études pour l'obtention du diplôme national d'ingénierie logicielle au cours de mon stage chez Sofiatech. Le principal objectif de ce projet était de développer une application de billetterie en ligne, permettant aux utilisateurs d'acheter des billets pour divers événements de manière simple et efficace.

Mots clés : billetterie en ligne, architecture microservices, Application web,.NET, Aspire, Angular, NgRx, Ocelot, RabbitMQ, DevOps.

Abstract

The purpose of this document is to provide a summary of the work carried out as part of my final year project for the national degree in software engineering during my internship at Sofiatech. The main objective of this project was to develop an online ticketing application, allowing users to purchase tickets for various events in a simple and efficient manner.

Keywords : online ticketing, microservices architecture, Web application, .NET, Aspire, Angular, NgRx, Ocelot, RabbitMQ, DevOps.