RÉPUBLIQUE DU BÉNIN
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ D'ABOMEY-CALAVI

INSTITUT DE FORMATION ET DE RECHERCHE EN INFORMATIQUE

BP 526 Cotonou Tel : +229 21 14 19 88
http://www.ifri-uac.net Courriel : contact@ifri.uac.bj

MÉMOIRE

pour l'obtention du

Diplôme de Licence en Informatique

Option : Génie Logiciel

Présenté par :
Boris Coffi Marcos **YEHOUESSI**

Plateforme d'évaluation des apprentissages en sécurité informatique.

Sous la supervision :
Dr. Emery Kouassi **ASSOGBA**, PMP

Membres du jury :

Arnaud AHOUANDJINOU	Docteur	IFRI	Président
Miranda GNONLONFOUN	Ingénieur	IFRI	Examinateur
Emery ASSOGBA	Docteur	IFRI	Rapporteur

Année Académique : **2022-2023**

Sommaire

Dédicace ii

Remerciements iii

Résumé iv

Abstract v

Table des figures vi

Liste des acronymes vii

Glossaire 1

Introduction 2

1 Revue de littérature 4

2 Analyse, modélisation, Choix techniques et sécurité 14

3 Prototype de l'application et discussion 23

Conclusion 32

Bibliographie 33

Webographie 33

Table des matières 34

Dédicace

À
ma famille et à tous ceux qui n'ont cessé de me soutenir durant mon cursus universitaire.
Je vous dédie ce document.

Remerciements

C'est avec un immense plaisir que je réserve ces quelques lignes en signe de gratitude et de reconnaissance à tous ceux qui ont contribué, de près ou de loin, à l'élaboration de ce travail. Je souhaite adresser, en premier lieu, mes remerciements les plus sincères à :

- DIEU le Père Tout Puissant, pour nous avoir donné la santé, la force et la persévérance pour accomplir ce travail ;
- Mr. Eugène C. EZIN, Directeur de l'Institut de Formation et de Recherche en Informatique qui m'a permis de travailler sur ce thème de mémoire ;
- Mon encadreur Dr. Emery Kouassi ASSOGBA pour sa disponibilité, son accompagnement et son soutien dans la réalisation de ce mémoire ;
- Je tiens également à remercier les membres du jury d'avoir consacré une partie de leur temps à la lecture de ce mémoire et pour l'intérêt qu'ils ont porté à ce travail.

Veuillez trouver dans ce travail l'expression de ma profonde reconnaissance et ma sincère gratitude.

Résumé

Notre travail porte sur la conception et le développement d'une plateforme d'évaluation des apprentissages en sécurité informatique. Cette plateforme permet aux utilisateurs d'accéder à un tableau de bord et de suivre des parcours éducatifs structurés, comprenant plusieurs formats de cours, à savoir des vidéos, du texte et des exercices pratiques utilisant la technologie des conteneurs Docker pour tester les vulnérabilités des applications web. Chaque cours se termine par un quiz permettant de valider les connaissances acquises.

Deux tableaux de bord distincts ont été développés pour répondre aux besoins spécifiques des différents types d'utilisateurs : un tableau de bord administrateur, qui permet de gérer tous les utilisateurs et les parcours éducatifs, assurant ainsi une gestion centralisée et efficace de la plateforme, et un tableau de bord client, qui permet aux apprenants d'accéder aux parcours disponibles et de suivre les cours à leur propre rythme.

Cette plateforme vise à offrir une expérience d'apprentissage pratique et interactive, permettant aux apprenants de développer des compétences essentielles en sécurité informatique dans un environnement sécurisé et contrôlé.

Mots-clés : Parcours éducatif structuré, Conteneurs Docker pour tester les vulnérabilités ; test de vulnérabilités Web.

Abstract

Notre travail porte sur la conception et le développement d'une plateforme d'évaluation des apprentissages en sécurité informatique. Cette plateforme permet aux utilisateurs d'accéder à un tableau de bord et de suivre des parcours éducatifs structurés, comprenant plusieurs formats de cours, à savoir des vidéos, du texte et des exercices pratiques utilisant la technologie des conteneurs Docker pour tester les vulnérabilités des applications web. Chaque cours se termine par un quiz permettant de valider les connaissances acquises.

Deux tableaux de bord distincts ont été développés pour répondre aux besoins spécifiques des différents types d'utilisateurs : un tableau de bord administrateur, qui permet de gérer tous les utilisateurs et les parcours éducatifs, assurant ainsi une gestion centralisée et efficace de la plateforme, et un tableau de bord client, qui permet aux apprenants d'accéder aux parcours disponibles et de suivre les cours à leur propre rythme.

Cette plateforme vise à offrir une expérience d'apprentissage pratique et interactive, permettant aux apprenants de développer des compétences essentielles en sécurité informatique dans un environnement sécurisé et contrôlé.

Key words: Structured learning path ; Docker containers for vulnerability testing ; Web vulnerability testing.

Table des figures

1.1 API RESTful 6
1.2 Processus de dockerisation d'une application 7

2.1 Diagramme de cas d'utilisation 16
2.2 Diagramme de classe 17
2.3 Diagramme de séquence du cas d'utilisation **Se connecter** 18
2.4 Diagramme de séquence du cas d'utilisation **Gérer un cours (de type lab)** 19
2.5 Comparaison de quelques architectures de développement web 20
2.6 Comparaison de quelques langages de modélisation 21
2.7 Comparaison des environnements de développement 22

3.1 Interface d'accueuil 24
3.2 Interface d'authentification 25
3.3 Interface des statistiques admin 26
3.4 Interface de gestion des utilisateurs 26
3.5 Présentation d'un parcours 27
3.6 Interface de création de cours de type lab 28
3.7 Interface du contenu d'un lab 29
3.8 Interface du dashboard étudiant 29
3.9 Cours de type conteneur 30

Liste des acronymes

API :

Application Programming Interface 1,

CSS :

Cascading Style Sheets 1,

HTML :

HyperText Markup Language 1,

MERN :

MongoDB, Express.js, React.js, Node.js 1,

OWASP :

Open Web Application Security Project 1,

SQL :

Structured Query Language 1, SQL

UML :

Unified Modeling Language, Langage de Modélisation Unifié 1,

XSS :

Cross-site Scripting 1, XSS

Glossaire

Docker : est une plateforme logicielle open-source qui simplifie le déploiement et la gestion d'applications grâce à la virtualisation légère via des conteneurs logiciels. Les conteneurs Docker encapsulent les applications et leur environnement, offrant ainsi une portabilité et une facilité de déploiement sur divers environnements. 1

JavaScript : est un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web 1

Laboratoire (Lab) : désigne un environnement de test dédié à l'évaluation et à l'exploitation des vulnérabilités d'une application. Dans le domaine de la sécurité informatique, un laboratoire est souvent représenté par un conteneur Docker contenant l'application cible. Ce conteneur permet de réaliser des tests de sécurité de manière isolée et contrôlée, facilitant ainsi l'identification des failles de sécurité et le développement de solutions de protection. 1

Introduction Générale

Contexte et justification

Dans un monde de plus en plus connecté, la sécurité des applications web est devenue une priorité. Les cyberattaques se multiplient, exploitant des vulnérabilités dans les systèmes pour voler des données sensibles, perturber les services et causer des dommages financiers considérables. Face à cette menace, des formations en sécurité informatique s'imposent comme une nécessité pour réduire ces risques d'attaque en fournissant aux entreprises des ressources humaines qualifiées et aptes à faire face à ces défis.

Cependant, l'apprentissage de la sécurité informatique pose des défis uniques. Les concepts théoriques, bien qu'importants, ne suffisent pas à préparer les apprenants aux réalités complexes et dynamiques des menaces en ligne. Il est impératif de compléter ces formations théoriques par des expériences pratiques permettant de tester et de renforcer les compétences dans un environnement contrôlé.

Pour répondre à ces défis, notre projet de développement d'une plateforme d'évaluation des apprentissages en sécurité informatique a été conçu pour offrir une expérience d'apprentissage immersive et interactive, intégrant des cours théoriques et pratiques pour couvrir tous les aspects de la sécurité informatique. Les utilisateurs ont accès à des vidéos éducatives, des contenus écrits détaillés, et surtout, à des exercices pratiques via des conteneurs Docker, qui leur permettent de tester des vulnérabilités réelles sur des applications web. Chaque module de cours se termine par un quiz, garantissant que les apprenants peuvent valider et consolider leurs connaissances acquises.

Problématique

Malgré l'importance de la sécurité informatique des applications web dans le monde actuel, les méthodes traditionnelles d'enseignement de cette discipline ne parviennent souvent pas à préparer adéquatement les apprenants aux défis complexes et dynamiques des cybermenaces réelles. Les cours théoriques, bien que nécessaires, manquent souvent de composantes pratiques qui permettent aux étudiants de tester et d'appliquer leurs connaissances dans des environnements contrôlés mais réalistes.

Comment concevoir et développer une plateforme d'évaluation des apprentissages en sécurité informatique qui offre une formation complète et interactive, intégrant des cours théoriques et pra-

tiques, tout en étant accessible et efficace tant pour les administrateurs que pour les apprenants ? Cette plateforme doit permettre aux utilisateurs de tester des vulnérabilités réelles dans un environnement sécurisé, afin de renforcer leurs compétences en sécurité web de manière autonome et progressive.

Objectifs

Dans ce contexte, le but visé est de concevoir et de développer une plateforme d'évaluation des apprentissages en sécurité informatique qui répond aux besoins de formation pratique en sécurité web. Cette plateforme vise à offrir une solution éducative complète, alliant théorie et pratique, pour permettre aux apprenants de développer des compétences essentielles en sécurité informatique. Les objectifs spécifiques sont les suivants :

- Créer une interface utilisateur intuitive qui regroupe deux tableaux de bord distincts pour les administrateurs et les apprenants.
- Inclure des cours sous forme de vidéos, de contenus écrits, et d'exercices pratiques utilisant des conteneurs Docker. Cette diversité de formats vise à couvrir tous les aspects de la sécurité web de manière exhaustive.
- Utiliser des conteneurs Docker pour créer des environnements de test permettant aux apprenants de détecter et exploiter des vulnérabilités réelles, simulant ainsi des cyberattaques authentiques pour appliquer les connaissances théoriques.
- Permettre aux apprenants de suivre les parcours éducatifs à leur propre rythme, en accédant aux cours et exercices pratiques selon leur emploi du temps et niveau de compétence.

Organisation du document

Le présent travail se présente en trois (03) chapitres. Le premier chapitre présente la revue de littérature. Le deuxième aborde l'analyse, la modélisation, les choix techniques et sécurité opérés en vue de la conception de la solution proposée. Le troisième, quant à lui concerne les résultats issus de la réalisation de l'application.

Chapitre 1

Revue de littérature

Introduction

Pour atteindre les résultats attendus de ce travail, il a été nécessaire d'examiner les différentes approches pédagogiques et les outils disponibles pour l'enseignement de la sécurité informatique, en mettant particulièrement l'accent sur l'intégration de composants pratiques et interactifs. Dans ce chapitre, nous allons aborder les enjeux du sujet, les défis liés à la gestion des environnements de test et analyser les travaux existants sur ce sujet.

1.1 Généralités sur le sujet

1.1.1 Fonctionnement d'une plateforme de gestion

En effet, qu'entend-on par gestion ? [3] Il s'agit de l'action ou de la manière de diriger, d'administrer, d'organiser quelque chose. Dans notre contexte, cela concerne **la gestion des cours de type écrit, vidéo ou pratique dans des conteneurs virtuels**. Les plateformes de gestion fonctionnent en général suivant l'administration de certaines données pour faciliter la gestion des tâches, la collaboration et l'organisation des informations. Elles offrent des moyens efficaces de gérer les flux de données et de rester organisé, que ce soit pour une ou plusieurs personnes.

1.1.2 Vulnérabilités Web

Les vulnérabilités Web [7] représentent une menace persistante pour la sécurité des applications en ligne, exposant les données sensibles des utilisateurs à des risques de piratage et de compromission. Nous allons aborder les principaux types de vulnérabilités Web, les techniques d'attaque associées et les stratégies de prévention.

1. **Typologie des vulnérabilités Web L'Injection de code (injections SQL, XSS)**
   Les injections de code [6] sont parmi les vulnérabilités les plus courantes. Les attaquants exploitent ces failles pour injecter et exécuter du code malveillant sur le serveur ou le navigateur de l'utilisateur, compromettant ainsi la confidentialité et l'intégrité des données.
   Mauvaise gestion des sessions et des cookies

Les applications Web mal configurées peuvent être sujettes à des attaques de fixation de session, de détournement de session ou de falsification de cookies, permettant aux attaquants d'accéder illicitement aux comptes des utilisateurs.

Cross-Site Scripting (XSS)

Les attaques XSS permettent à un attaquant d'injecter du code malveillant dans des pages Web consultées par d'autres utilisateurs. Cela peut entraîner le vol de sessions utilisateur, le vol de cookies ou d'autres formes de compromission de données.

Inclusions de fichiers non sécurisées

Les vulnérabilités d'inclusions de fichiers non sécurisées permettent à un attaquant d'inclure et d'exécuter du code arbitraire à partir de fichiers externes, ouvrant ainsi la porte à une variété d'attaques telles que l'exécution de code à distance.

2. **Techniques d'attaque**

   - Les scanners automatiques qui sont des outils de balayage automatisé, qui identifient les vulnérabilités Web en analysant la structure et le comportement des applications. Ce qui permet aux attaquants de cibler rapidement les points faibles et de les exploiter.
   - Les attaquants utilisent des techniques d'ingénierie sociale pour tromper les utilisateurs et leur faire divulguer des informations sensibles, telles que des identifiants de connexion ou des informations de carte de crédit, qui peuvent ensuite être utilisées pour compromettre les applications Web.
   - Les attaques d'injection de code, telles que les injections SQL et XSS, demeurent parmi les techniques les plus couramment utilisées par les attaquants pour exploiter les vulnérabilités Web et compromettre la sécurité des applications.

3. **Stratégies de prévention :**

   - Les applications Web doivent mettre en œuvre une validation rigoureuse des données entrantes et échapper les sorties pour empêcher les attaques d'injection de code, telles que les injections SQL et XSS.
   - Les applications doivent utiliser des mécanismes robustes de gestion des sessions et des cookies, tels que la génération de jetons anti-CSRF (Cross-Site Request Forgery) et l'utilisation de l'attribut sécurisé pour les cookies.
   - Les applications doivent mettre en place des contrôles d'accès stricts pour les fichiers et répertoires sensibles, en limitant les privilèges d'accès aux seuls utilisateurs autorisés.

1.1.3 Utilité d'une plateforme de gestion dans le contexte des environnements de test de vulnérabilités Web

Les tests de vulnérabilités jouent un rôle crucial dans ce processus, offrant aux apprenants l'opportunité de mettre en pratique leurs connaissances théoriques et de développer leurs compétences en matière de détection et de correction de failles de sécurité. La mise en place des cours de type pratique dans des environnements de test pour les rendre disponibles à plusieurs utilisateurs s'avère complexe, ce qui peut freiner l'efficacité du des cours mis en place. C'est là qu'intervient l'utilité d'une plateforme de gestion dédiée.

Une plateforme de gestion des cours pratiques dans des environnements virtuels de test en sécurité offre de nombreux avantages :

1. **Centralisation et simplification de la gestion :**
    - Création et configuration automatisées de tous les types de cours.
    - Gestion centralisée des utilisateurs, des droits d'accès et des ressources.
    - Suivi et reporting des tests de vulnérabilités.
2. **Amélioration de la sécurité et de la fiabilité :**
    - Environnements de test isolés et sécurisés.
    - Contrôle des accès et des modifications.
    - Conformité aux normes de sécurité.
3. **Gain de temps et d'efficacité :**
    - Réduction du temps de mise en place des environnements de test.
    - Automatisation des tâches répétitives.
    - Partage des environnements de test entre équipes.
4. **Meilleure collaboration et communication :**
    - Plateforme collaborative pour les équipes de sécurité et de développement.
    - Communication centralisée et transparente.
    - Suivi et reporting partagés.

1.1.4 API

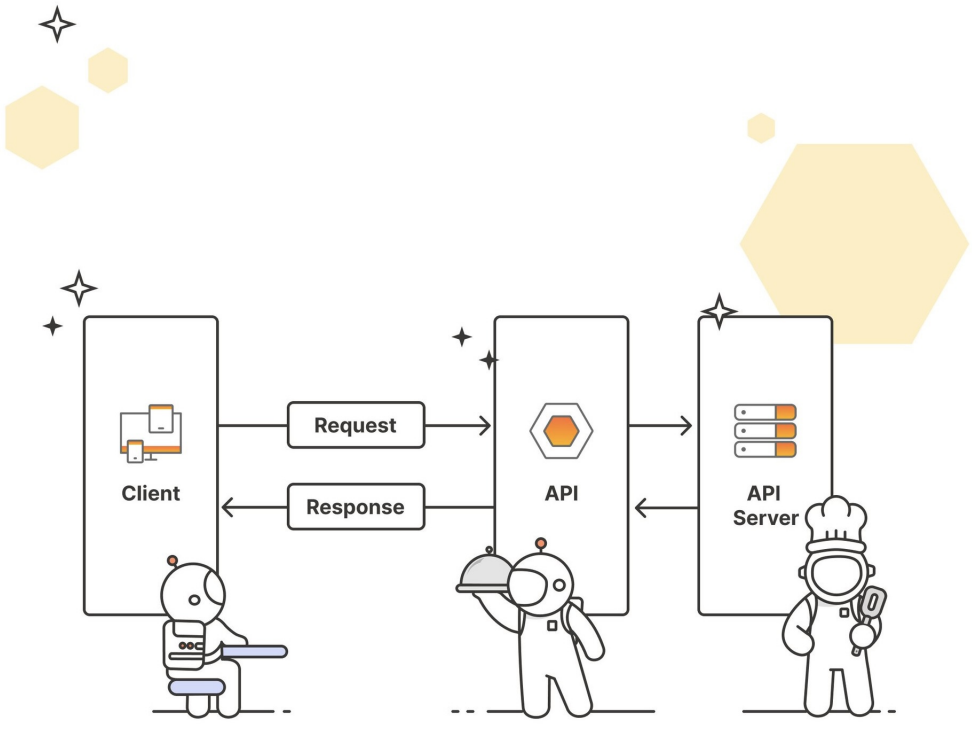


FIGURE 1.1 : API RESTful

Les interfaces de programmation d'applications (API) [1] sont comme des ponts qui permettent à différentes applications de communiquer entre elles. Elles sont très importantes dans le développement de logiciels, surtout pour les applications Web. Elles peuvent être conçues de différentes manières. On a notamment les **API RESTful et les API SOAP**. Les premières sont plus simples et flexibles, tandis que les secondes offrent une sécurité et une fiabilité supérieures. Pour leurs sécurisation il faut s'assurer que seules les bonnes personnes ou applications y ont accès. Pour cela, on utilise des méthodes d'authentification comme les jetons d'accès ou les clés API.

1.1.5 Docker et API Docker

1. **Docker** [2] est une plateforme de conteneurisation populaire qui simplifie le déploiement et la gestion d'applications. On peut l'utiliser pour créer des conteneurs de système dans lesquels on peut effectuer des tests de vulnérabilité. Il utilise la technologie de conteneurisation pour encapsuler les applications et leurs dépendances dans des environnements isolés appelés conteneurs. Cela permet une portabilité accrue et une utilisation plus efficace des ressources. Le processus de dockerisation d'une application avec Dockerfile, image et conteneur peut se faire suivant ce procédé.

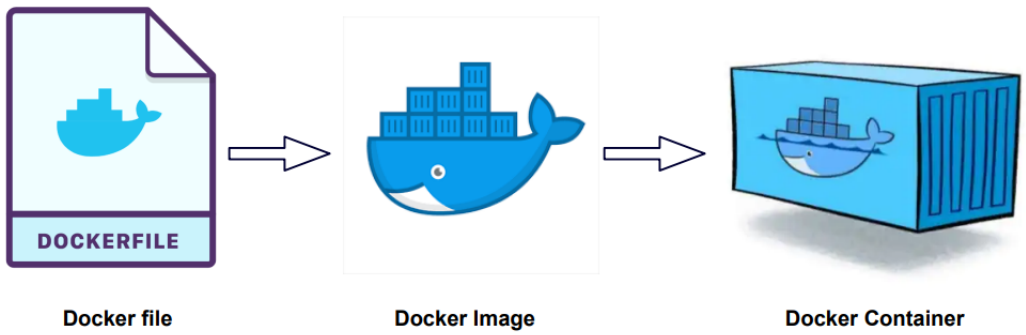


FIGURE 1.2 : Processus de dockerisation d'une application

Le **Dockerfile** est un fichier texte qui contient les instructions pour construire l'image Docker. Il définit l'environnement de construction, les logiciels à installer, les fichiers à copier et les commandes à exécuter pour créer l'image. Ensuite la commande `docker build` est utilisée pour construire l'image Docker à partir du Dockerfile puis `docker run` pour exécuter l'image Docker et créer un conteneur.

Avantages de la dockerisation :

- Les applications conteneurisées peuvent être exécutées sur n'importe quelle plateforme compatible Docker.
- Les conteneurs s'exécutent dans un environnement isolé, ce qui les rend plus sécurisés et plus fiables.
- Les conteneurs peuvent être facilement déployés et mis à l'échelle.
- Les images Docker garantissent un environnement d'exécution cohérent pour vos applications.

2. **L'API Docker** expose un ensemble de fonctionnalités permettant de contrôler et de manipuler les conteneurs Docker. Ces fonctionnalités incluent la gestion des images, des conteneurs des réseaux, ainsi que des opérations système telles que la surveillance des événements et la gestion des ressources.

1.2 Gestion des ressources et sécurité

Assurer une bonne gestion des ressources et garantir leurs sécurité sont essentielles pour améliorer l'expérience d'apprentissage.

1.2.1 Gestion des utilisateurs, des inscriptions et des progrès des apprenants

La gestion des utilisateurs constitue un aspect essentiel de notre plateforme. Nous avons mis en place un processus d'inscription sécurisé et efficace, ainsi qu'un suivi des progrès des apprenants pour garantir une expérience optimale.

- **Processus d'inscription et validation des comptes**
  Lorsqu'un utilisateur s'inscrit sur notre plateforme, il lui est demandé de fournir des informations de base telles que son nom, son adresse e-mail et un mot de passe sécurisé. Une fois le formulaire d'inscription soumis, un e-mail de confirmation est automatiquement envoyé à l'adresse e-mail fournie. Ce courriel contient un lien de connexion unique, accompagné d'un jeton généré de manière aléatoire, qui est valide pendant une période spécifiée. En cliquant sur ce lien, l'utilisateur confirme la validité de son adresse e-mail et active son compte sur la plateforme. Tant que ce lien n'est pas cliqué, le compte reste en attente de validation.
- **Attribution des rôles aux utilisateurs**
  Initialement, chaque nouvel utilisateur est enregistré avec le statut d'apprenant. Cependant, l'administrateur de la plateforme a le pouvoir de modifier le statut de l'utilisateur, lui accordant ainsi les privilèges d'enseignant ou d'administrateur, en fonction de ses responsabilités et de son niveau d'autorisation.
- **Suivi des progrès des apprenants**
  Le suivi des progrès des apprenants est automatiquement enregistré à chaque achèvement d'un laboratoire de test de sécurité ou à la fin d'un cours ou d'un parcours. Ces données de progression sont associées à chaque utilisateur, permettant ainsi de suivre et de visualiser leur évolution dans l'apprentissage.

Cette approche garantit non seulement la sécurité des comptes des utilisateurs en vérifiant les adresses e-mail lors de l'inscription, mais elle permet également une gestion flexible des rôles et un suivi précis des progrès des apprenants tout au long de leur parcours d'apprentissage sur notre plateforme.

1.2.2 Sécurité Web

1.2.2.1 Normes et bonnes pratiques en matière de sécurité Web

La sécurisation des applications Web [4] est devenue un enjeu majeur dans un monde numérique en constante évolution. Pour répondre à cette préoccupation croissante, des **normes et des bonnes**

pratiques ont été établies pour guider les développeurs et les professionnels de la sécurité dans leurs efforts visant à protéger les systèmes et les données contre les attaques malveillantes. Ces normes et bonnes pratiques fournissent un cadre structuré pour identifier, évaluer et atténuer les vulnérabilités potentielles dans les applications Web.

1. **Normes de sécurité**
   L'une des normes les plus influentes dans ce domaine est le Top 10 de l'OWASP (Open Web Application Security Project). Publié régulièrement, ce document répertorie les dix principales vulnérabilités des applications Web, telles que les injections SQL, les attaques par XSS (Cross-Site Scripting), et les problèmes d'authentification et de gestion des sessions. En suivant les recommandations de l'OWASP Top 10, les développeurs peuvent concevoir des applications Web plus résistantes aux attaques. Outre l'OWASP, d'autres normes et cadres de sécurité, tels que ISO/IEC 27001, offrent également des lignes directrices précieuses pour assurer la sécurité des applications Web. ISO/IEC 27001 établit des exigences pour la mise en place, la mise en œuvre, la maintenance et l'amélioration d'un système de gestion de la sécurité de l'information. En conformité avec cette norme, les organisations peuvent déployer des mesures de sécurité cohérentes et efficaces pour protéger leurs actifs informationnels.

2. **Bonnes pratiques pour une sécurité optimale**
   En plus de l'application des normes, l'adoption de bonnes pratiques est essentielle pour minimiser les risques de cyberattaques. Voici quelques exemples :

   - La sécurité doit être intégrée dès la phase de conception comme un élément essentiel du processus de développement logiciel, et non comme un simple ajout.
   - Il est essentiel de valider et filtrer les données des utilisateurs pour prévenir les injections de code et les attaques malveillantes.
   - Gérer les identifiants et les mots de passe de manière sécurisée implique l'utilisation de mots de passe forts et uniques, ainsi que l'authentification multifacteur, pour renforcer la sécurité des comptes utilisateurs.
   - Mettre en œuvre des contrôles de sécurité robustes nécessite des mesures techniques comme le chiffrement des données et le filtrage du trafic réseau, indispensables pour protéger les applications et les infrastructures.

1.2.3 Études de cas de violations de sécurité Web

Les études de cas de violations de sécurité Web offrent des perspectives précieuses sur les failles courantes, les méthodes d'attaque utilisées et les conséquences des vulnérabilités non traitées. Voici quelques exemples notables de violations de sécurité Web qui ont eu un impact significatif :

1. **Violation de données de Yahoo (2013-2014)**
   Une série d'attaques contre Yahoo entre 2013 et 2014 a compromis les données de plus de 3 milliards de comptes d'utilisateurs. Les informations volées comprenaient des données sensibles telles que les noms, les adresses e-mail, les dates de naissance et les mots de passe cryptés.

2. **Piratage de Equifax (2017)**
   Equifax, une importante société d'évaluation du crédit, a été victime d'une faille de sécurité en 2017, entraînant la compromission des données personnelles de près de 147 millions de

personnes. Les informations volées comprenaient les noms, les numéros de sécurité sociale, les dates de naissance et les adresses.

3. **Violation de données de Marriott International (2018)**
   Une cyberattaque contre les systèmes de réservation de Marriott International a exposé les informations personnelles de près de 500 millions de clients. Les données compromises comprenaient des détails tels que les noms, les adresses e-mail, les numéros de téléphone et les numéros de passeport.

4. **Violation de données de Facebook/Cambridge Analytica (2018)**
   L'affaire Cambridge Analytica a révélé que les données personnelles de millions d'utilisateurs de Facebook avaient été collectées de manière inappropriée par l'entreprise de consultation politique. Ces données ont ensuite été utilisées pour influencer les élections et les opinions politiques dans plusieurs pays.

1.2.4 Aspects Juridiques et Éthiques de la Sécurité Web

La sécurité Web ne se résume pas uniquement à la protection technique des systèmes et des données. Elle soulève également d'importantes questions d'ordre juridique et éthique qui doivent être prises en compte par les organisations et les développeurs.

1. **Aspects juridiques**

   - Les organisations doivent se conformer à un ensemble de lois et réglementations relatives à la sécurité des données, à la vie privée et à la protection des consommateurs.
   - Il est important de respecter les droits de propriété intellectuelle et les droits d'auteur des logiciels et des contenus utilisés dans les applications Web.
   - Cybercriminalité et responsabilité : Les organisations peuvent être tenues responsables des cyberattaques qui surviennent sur leurs plateformes.

2. **Aspects éthiques**

   - Les organisations doivent respecter la vie privée des utilisateurs et protéger leurs données personnelles contre l'utilisation abusive.
   - Les organisations doivent être transparentes quant à leurs pratiques de sécurité et être responsables de leurs actions.
   - Les applications Web doivent être accessibles à tous les utilisateurs, sans discrimination.

1.2.5 Contrôle d'accès et sécurité des environnements de test

Le contrôle d'accès et la sécurité des environnements de test revêtent une importance capitale dans notre plateforme d'apprentissage en sécurité Web. Nous avons mis en place des mesures robustes pour garantir que seuls les utilisateurs autorisés ont accès aux environnements de test et que ces environnements sont sécurisés contre les menaces potentielles.

Accès aux environnements de test

L'accès aux environnements de test est strictement contrôlé pour garantir la sécurité et l'intégrité des données. Seuls les utilisateurs ayant les autorisations appropriées peuvent accéder aux laboratoires

de test et aux configurations d'environnement associées. Ce contrôle d'accès est géré par un système d'authentification robuste qui vérifie l'identité de l'utilisateur à chaque tentative de connexion.

Sécurité des environnements de test

Nous accordons une attention particulière à la sécurité des environnements de test pour garantir qu'ils sont résistants aux attaques et aux vulnérabilités potentielles. Nous utilisons des techniques de virtualisation sécurisées, telles que les conteneurs Docker, pour isoler chaque environnement de test et prévenir les interférences entre les différentes instances. De plus, nous appliquons des pratiques de sécurité recommandées, telles que la mise à jour régulière des logiciels, la limitation des privilèges d'accès et la surveillance continue des activités suspectes. Des mécanismes de détection d'intrusion sont également en place pour détecter et répondre rapidement à toute tentative d'attaque ou de compromission des environnements de test.

Cette approche globale de contrôle d'accès et de sécurité garantit que nos environnements de test sont accessibles uniquement aux utilisateurs autorisés et qu'ils offrent un environnement sûr et protégé pour la réalisation des exercices de sécurité Web et des tests de vulnérabilités.

1.3 Analyse des travaux existants

1.3.1 Présentation des plateformes de sécurité

Le marché des plateformes de gestion en sécurité est en pleine expansion, avec de nombreux acteurs proposant des solutions variées. Voici quelques exemples de plateformes populaires :

- **OWASP** qui est une plateforme d'apprentissage gratuite et open-source qui propose des modules sur différents aspects de la sécurité informatique, y compris la gestion des vulnérabilités.
- **SecureFlag** qui propose des défis et des scénarios réalistes pour la pratique de la gestion des vulnérabilités.
- **Hack The Box** qui est une plateforme d'apprentissage pratique de la sécurité informatique, offrant aux utilisateurs la possibilité de tester leurs compétences en hacking et en sécurité réseau dans un environnement contrôlé. La plateforme propose une variété de machines virtuelles vulnérables, appelées "boxes", que les utilisateurs doivent pirater pour accéder à des données sensibles ou obtenir un accès administrateur.
- **SafeStack** qui propose des exercices et des tutoriels interactifs permettant aux développeurs de comprendre et d'appliquer les principes de sécurité lors de la conception et de la programmation d'applications web.

1.3.2 Étude comparative

Bien que les plateformes de sécurité telles qu'OWASP, SecureFlag, Hack The Box et SafeStack offrent des ressources précieuses pour l'apprentissage et la gestion des vulnérabilités, elles présentent toutes des limites qui peuvent entraver leur efficacité.

OWASP, bien qu'elle soit une référence en matière de ressources éducatives open-source, peut parfois manquer de contenu à jour sur les menaces émergentes et les techniques d'attaque récentes.

De plus, son approche théorique peut ne pas suffire pour ceux qui recherchent des expériences pratiques.

SecureFlag propose des défis réalistes, mais ces scénarios peuvent être trop spécifiques et ne pas couvrir l'éventail complet des vulnérabilités que les professionnels rencontrent sur le terrain. Cela peut conduire à un manque de préparation face à des situations variées.

Hack The Box, avec son environnement pratique, est excellent pour tester des compétences techniques, mais son accès nécessite un niveau de compétence préalable qui peut exclure les débutants. De plus, la complexité des machines virtuelles peut être décourageante pour certains utilisateurs.

Enfin, **SafeStack**, bien que fournissant des exercices interactifs pour les développeurs, peut ne pas aborder suffisamment les aspects pratiques de la gestion des vulnérabilités dans des environnements de production. Cela peut limiter la capacité des développeurs à appliquer ces principes dans des situations réelles.

Chaque plateforme présente des avantages, mais leurs limites respectives soulignent la nécessité d'une approche complémentaire et intégrée pour une formation complète en matière de sécurité.

1.3.3 Perspectives d'amélioration

L'analyse des travaux existants a révélé plusieurs besoins non satisfaits dans les plateformes de gestion en sécurité web actuelles, mettant en évidence les opportunités d'innovation et d'amélioration dans ce domaine. Parmi ces besoins, nous pouvons identifier les points suivants :

- **Apprentissage interactif et pratique**
  Les plateformes de gestion en sécurité web existantes offrent souvent des cours théoriques, mais peu d'opportunités d'apprentissage pratique et interactif. La solution proposée introduit des cours de type laboratoire (Lab) qui permettent aux apprenants de mettre en pratique leurs connaissances en effectuant des tests de sécurité réels dans des environnements virtuels sécurisés. Cette approche interactive favorise un apprentissage plus efficace et immersif, permettant aux apprenants de développer leurs compétences en sécurité web de manière pratique.

- **Coût abordable**
  Une autre lacune des plateformes de gestion en sécurité web existantes réside dans leur coût élevé, limitant l'accès à l'apprentissage en sécurité web à un public restreint. La solution proposée vise à démocratiser l'accès à l'éducation en sécurité web en offrant des cours et des environnements de test à un coût abordable. En utilisant des conteneurs Docker pour créer des environnements virtuels sécurisés, la solution minimise les coûts d'infrastructure et permet une diffusion plus large des cours de sécurité web.

- **Facilité d'utilisation et d'accès**
  Les plateformes de gestion en sécurité web traditionnelles peuvent parfois être complexes à utiliser et nécessiter une installation et une configuration compliquées. La solution proposée simplifie l'accès à l'apprentissage en sécurité web en permettant aux apprenants de démarrer facilement des environnements de test dans leur navigateur Web à l'aide d'un simple bouton

"run". Cette approche intuitive et conviviale rend l'apprentissage en sécurité web plus accessible à un large public, y compris aux débutants et aux non-initiés.

En mettant en œuvre ces innovations clés, la solution proposée adresse les lacunes et les besoins non satisfaits dans les plateformes de gestion en sécurité web existantes, offrant ainsi une expérience d'apprentissage en sécurité web plus interactive, abordable et accessible à tous.

Conclusion

Dans ce chapitre, nous avons exploré les divers aspects de la gestion des types de cours, et surtout ceux de type conteneur pour des tests pratiques en sécurité informatique, en mettant en lumière l'importance de notre plateforme d'évaluation des apprentissages dans le domaine de la cybersécurité. Nous avons vu que les plateformes de gestion en sécurité informatique offrent des solutions essentielles pour faciliter les tests de vulnérabilités et assurer un environnement contrôlé et sécurisé pour les apprenants. Le prochain chapitre présentera notre solution en détail, mettant en avant sa conception basée sur les principes de la sécurité informatique et de l'apprentissage interactif. Nous décrirons les fonctionnalités clés de notre plateforme et expliquerons comment elle répond aux défis identifiés dans ce chapitre

Chapitre 2

Analyse, modélisation, Choix techniques et sécurité

Introduction

Dans ce chapitre, nous aborderons comment garantir une bonne gestion pour un accès efficace aux cours de formation en commençant par une analyse approfondie des besoins, la conception de la plateforme et les choix techniques effectués pour y parvenir. Nous examinerons également les considérations de sécurité intégrées à chaque étape du processus.

2.1 Analyse des besoins

Dans la conception de notre plateforme l'analyse des besoins est la partie qui définira les différentes fonctionnalités dont nous avons besoins dans la réalisation du projet pour atteindre les objectifis visés. Nous avons deux types d'utlisateurs impliqués dans l'utilisation de la plateforme. L'administrateur et l'étudiant.

2.1.1 L'administrateur

L'admin peut :

- s'authentifier pour accéder à son tableau de bord
- voir le nombre d'étudiants de la plateforme sur le tableau de bord
- Faire la gestion des parcours qui contiennent de cours et de labs (conteneurs docker de test).
- modifier, supprimer ou changer le rôle d'un utilisateur
- supprimer les conteneurs
- modifier son propre profil

2.1.2 L'étudiants

L'étudiants peut :

- s'inscrire sur la plateforme et recevoir un lien d'authentification par mail pour validé son compte.
- consulter le nombre de parcours, de Labs et trophés gagné suite l'exécution d'un lab.
- voir la liste des parcours et cours
- suiivre un parcours ou un cours.
- terminé un parcours avec certificat à l'appui
- voir les statistiques graphique des Labs sur les vulnérabilités

2.2 Conception

Pour concevoir la plateforme, nous avons choisi le langage de modélisation UML (Unified Modeling Language). En utilisant UML, nous serons en mesure de définir les différents aspects de notre solution, y compris les acteurs impliqués, les interactions entre les différents modules et la structure globale de la plateforme.

2.2.1 Identification des acteurs

On a deux types de d'acteur impliqués dans l'utilisation de plateforme, il s'agit :

- des admins
- des étudiants

2.2.2 Le diagramme de cas d'utilisation

Le diagramme de cas d'utilisation est un diagramme qui illustre les interactions entre les acteurs (utilisateurs ou systèmes externes) et un système logiciel. Son rôle est de définir les fonctionnalités offertes par le système du point de vue de l'utilisateur. Dans notre plateforme, deux (02) types d'utilisateurs sont impliqués dans les interactions : l'administrateur et l'étudiant.
La figure 2.1 présente le diagramme de cas d'utilisation de notre plateforme

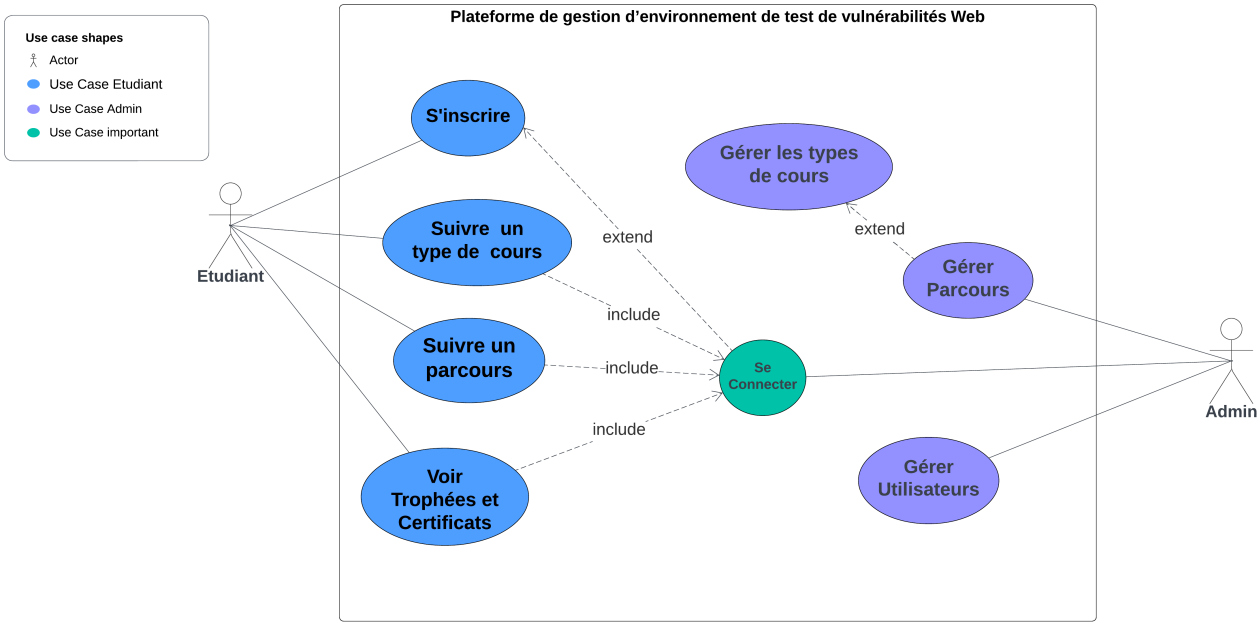


FIGURE 2.1 : Diagramme de cas d'utilisation

Les acteurs principaux sont les Apprenants et l'Admin agit en tant qu'acteur secondaire. Pour l'Admin, les cas d'utilisation incluent des fonctionnalités telles que l'authentification, la gestion des utilisateurs, la création et la gestion de parcours ainsi que la gestion des cours de tous types. Les Étudiants peuvent s'inscrire, consulter les parcours et les cours disponibles, suivre des parcours, terminer des parcours avec des certificats en validant leurs connaissances avec des questionnaires.

2.2.3 Le diagramme de classe

Il illustre la structure d'un système en identifiant les classes, leurs attributs, leurs méthodes et les relations qui existent entre elles.
Le diagramme de classe de notre plateforme se présente comme suit :

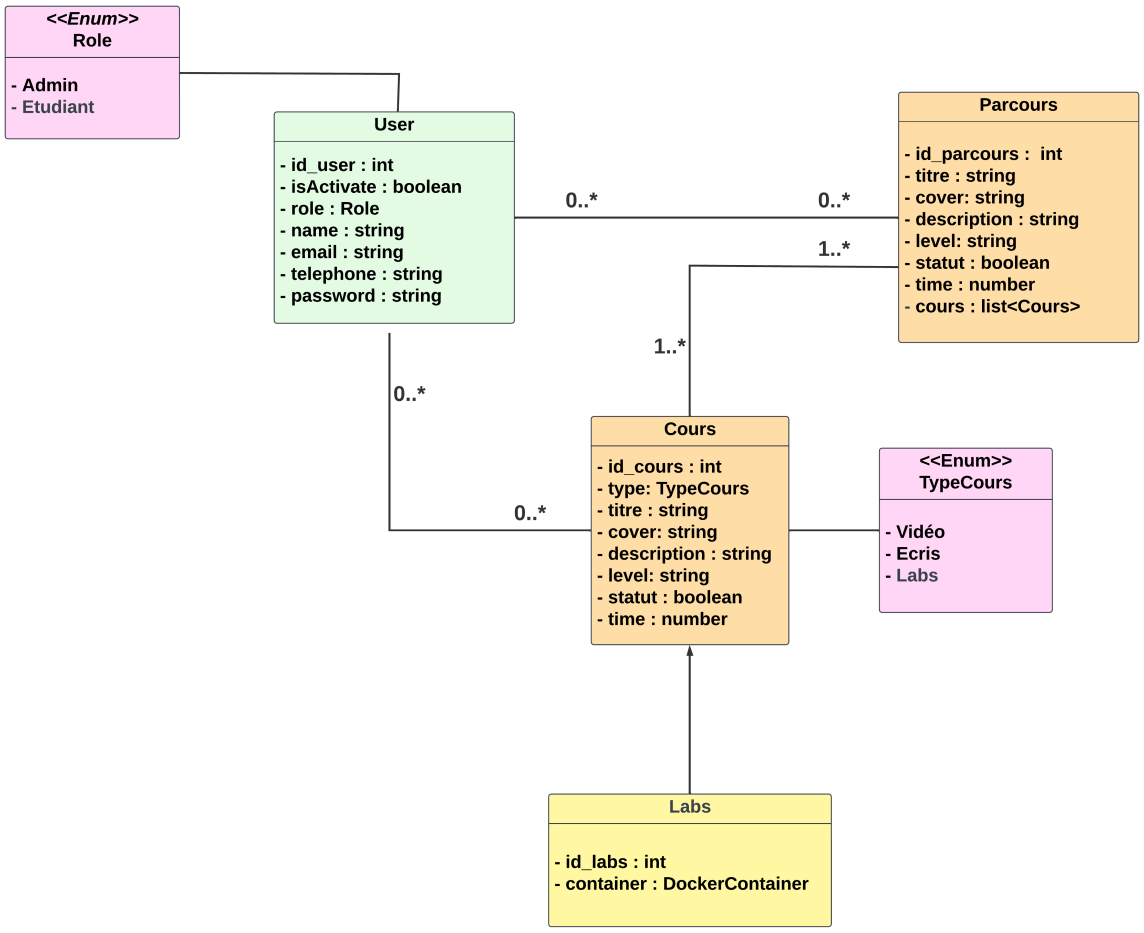


FIGURE 2.2 : Diagramme de classe

Explication des relations du diagramme de classe

Le diagramme de classe représente les entités principales de la plateforme. Il décrit les relations entre les utilisateurs, les parcours, les cours et les laboratoires. Chaque Utilisateur peut être associé à zéro ou plusieurs parcours, tandis que chaque parcours peut contenir plusieurs cours. Les cours, à leur tour, peuvent être liés à plusieurs laboratoires. Chaque utilisateur est également associé à un rôle spécifique dans le système, tel que Administrateur ou Apprenant, ce qui détermine ses autorisations et ses privilèges.

2.2.4 Le diagramme de séquence

Le diagramme de séquence montre les séquences d'interactions entre les différents objets ou acteurs dans un système logiciel. A notre niveau il va nous permettre de modéliser les interactions entre les utilisateurs et les composants du système lors de l'exécution des scénarios d'utilisation. Dans notre contexte nous allons illustistré le diagramme de séquence de deux scénarios, l'authentification et la création d'un cours de type labs.

Diagramme de séquence du cas d'utilisation **Se connecter** :

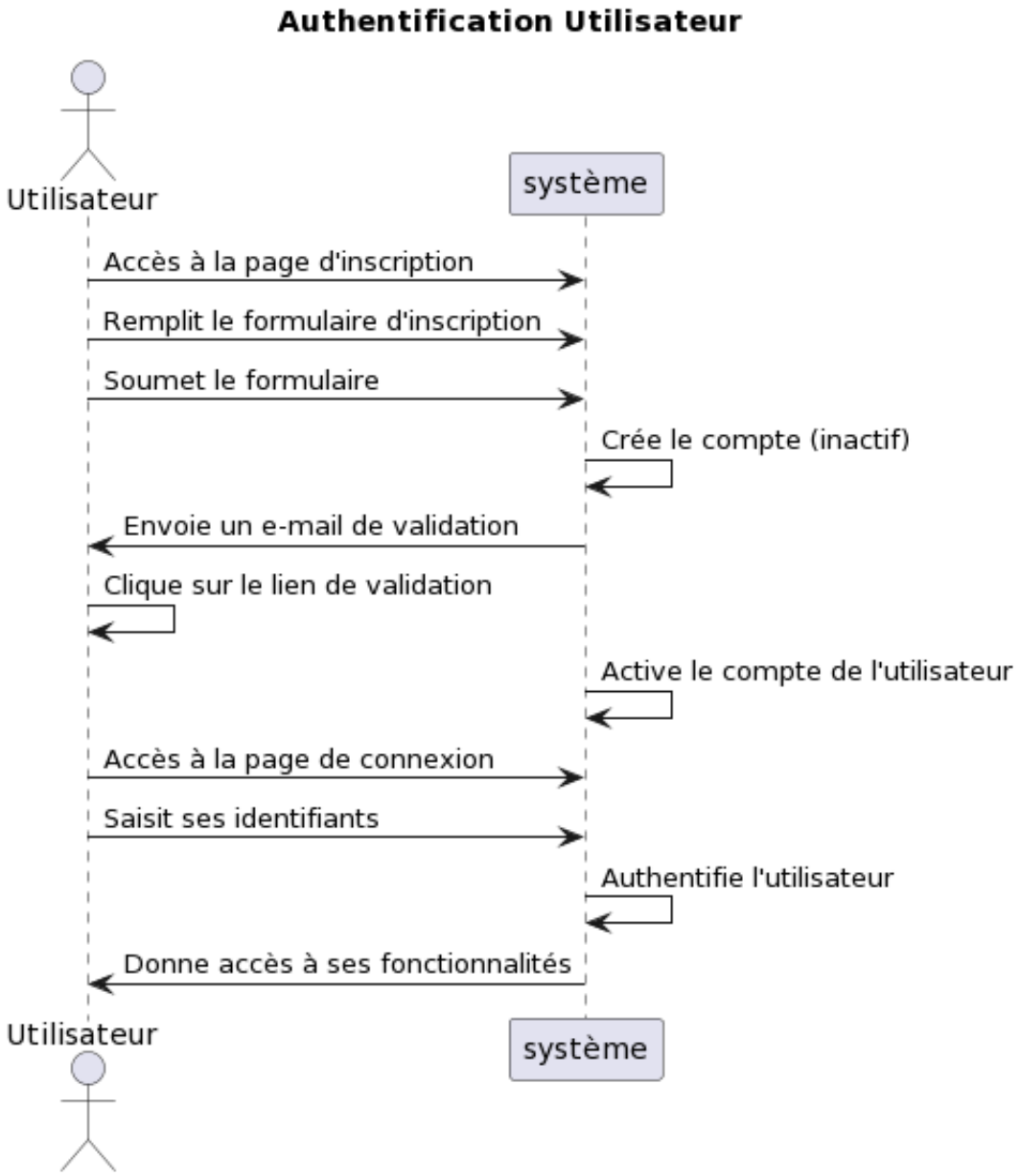


FIGURE 2.3 : Diagramme de séquence du cas d'utilisation **Se connecter**

Pour s'authentifier il faut s'inscrire, le système crée le compte mais il est pas activé. Le système envoi Un mail de validation de compte qui contient un lien qui une fois cliqué active le compte créé et le dirige vers une page de connection pour qu'il se connecte. Et ainsi il s'authentifie.

Diagramme de séquence du cas d'utilisation **Gérer un cours (de type lab)** :

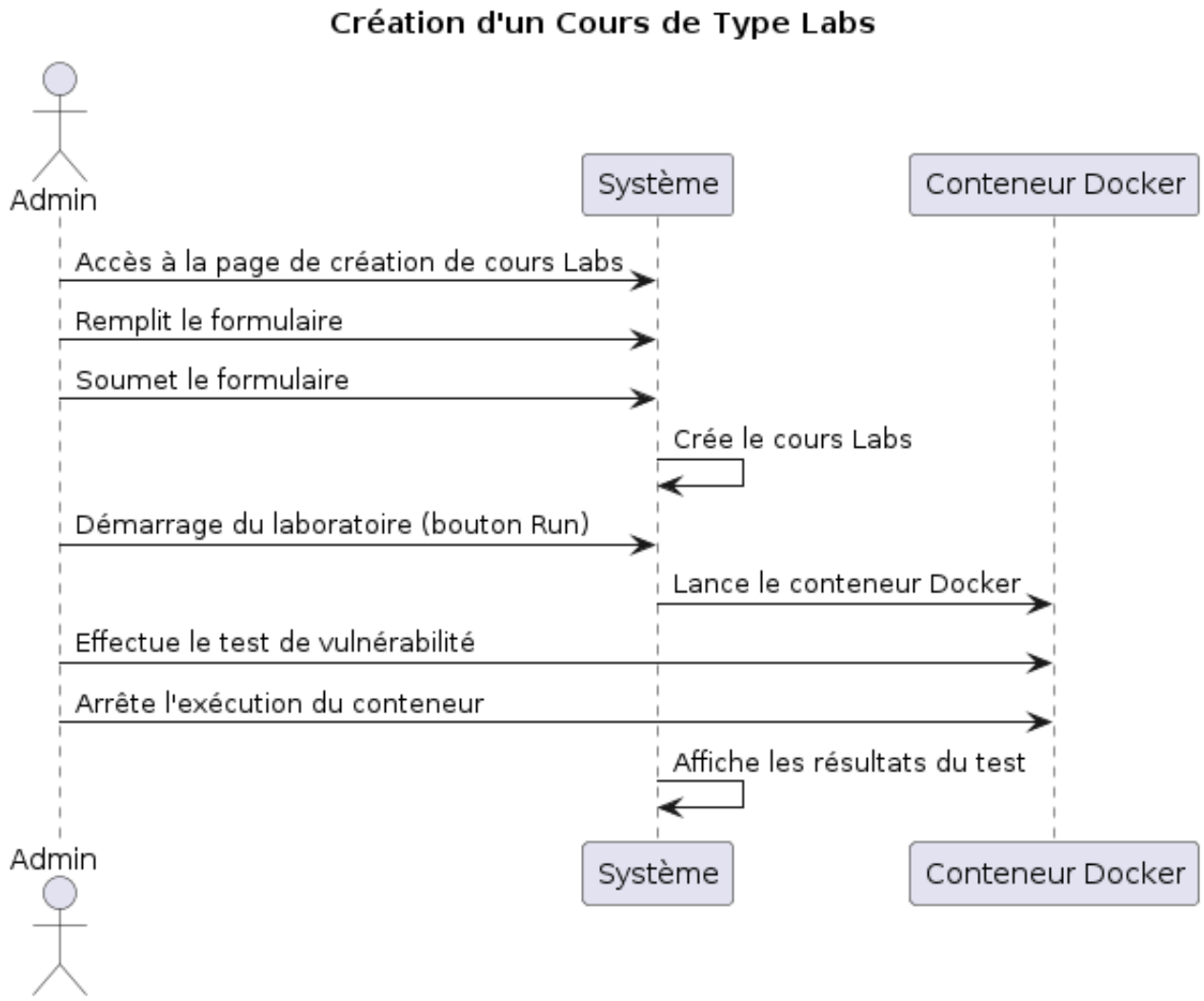


FIGURE 2.4 : Diagramme de séquence du cas d'utilisation **Gérer un cours (de type lab)**

Pour créer un cours de type lab, il faut renseigner les informations nécessaires (comme le titre du lab, la technologie, le type de lab, le service, le type d'accès et le nom de l'image du lab) dans un formulaire. Le système crée le lab qui peut être démarré avec un bouton run, une fois démarré, on se trouve dans un conteneur Docker qui contient un test d'évaluation de vulnérabilité. Une fois le test terminé, on peut arrêter l'exécution du conteneur et voir si le test est réussi ou pas.

2.3 Choix techniques

Pour le développement de la plateforme, certains choix techniques ont été fait, notamment en ce qui concerne l'architecture, le langage de modélisation et les technologies utilisées pour le frontend, le backend et les tests d'API.

2.3.1 Choix de l'architecture

La table ci dessous compare cinq architectures populaires de développement web : MERN, LAMP, MEAN, Django et .NET. Les critères d'évaluation incluent la facilité d'utilisation, la sécurité, la modularité, la performance, la communauté et le support, la gestion des routes, la flexibilité, ainsi que

l'année d'ancienneté. Ces critères permettent de mieux comprendre les forces et les faiblesses de chaque architecture, facilitant ainsi le choix en fonction des besoins spécifiques d'un projet.

Critères	MERN (MongoDB, Express, React, Node.js)	LAMP (Linux, Apache, MySQL, PHP)	MEAN (MongoDB, Express, Angular, Node.js)	Django (Django)	.NET (.NET Framework)
Facilité d'utilisation	Très bonne	Bonne	Bonne	Très bonne	Moyenne
Sécurité	Très bonne	Moyenne	Bonne	Excellente	Excellente
Modularité	Excellente	Moyenne	Bonne	Bonne	Très bonne
Performance	Très bonne	Moyenne	Très bonne	Bonne	Excellente
Communauté et support	Excellente	Bonne	Bonne	Très bonne	Excellente
Gestion des routes	Excellente	Moyenne	Bonne	Bonne	Bonne
Flexibilité	Excellente	Moyenne	Bonne	Moyenne	Bonne
Année d'ancienneté	2013	1995	2013	2005	2002

FIGURE 2.5 : Comparaison de quelques architectures de développement web

L'architecture MERN [5] offre une combinaison de technologies modernes, parfaitement adaptée au développement de notre plateforme de gestion.
Elle est basée sur MongoDB, Express.js, React.js et Node.js et choisie pour notre plateforme de gestion d'environnements de test de vulnérabilités Web en raison de ses nombreux avantages.
MongoDB offre une flexibilité et une scalabilité pour le stockage des données, tandis qu'**Express.js** simplifie le développement backend avec sa modularité et sa gestion des routes.**React.js** facilite la création d'interfaces utilisateur interactives et réactives, tandis que **Node.js** permet une cohérence et une productivité globales avec JavaScript sur le frontend et le backend. **L'architecture MERN** que nous avons choisie offre une solution complète et harmonieuse pour développer une plateforme, réactive et facilement maintenable dans le temps.

2.3.2 Le langage de modélisation

La table ci dessous compare quatres outils de langages de modélisation populaires tels que UML (Unified Modeling Language), SysML (Systems Modeling Language), BPMN (Business Process Model and Notation) et ERD (Entity-Relationship Diagram). Les critères d'évaluation incluent la clarté de la représentation, la flexibilité, le support de la communauté, la capacité à modéliser des systèmes complexes, l'intégration avec les outils de développement, et la popularité. Ces critères permettent de comprendre les avantages et les inconvénients de chaque langage de modélisation dans le contexte du développement logiciel.

Critères	UML	SysML	BPMN	ERD
Clarté de la représentation	Très bonne	Bonne	Très bonne	Excellente
Flexibilité	Excellente	Bonne	Moyenne	Moyenne
Support de la communauté	Excellente	Bonne	Moyenne	Bonne
Capacité à modéliser des systèmes complexes	Excellente	Très bonne	Moyenne	Moyenne
Intégration avec les outils de développement	Très bonne	Bonne	Excellente	Bonne
Popularité	Très populaire	Populaire	Moins populaire	Populaire

FIGURE 2.6 : Comparaison de quelques langages de modélisation

Le langage de modélisation UML (Unified Modeling Language) a été choisi comme langage de modélisation pour notre plateforme d'évaluation des apprentissages en sécurité informatique en raison de sa polyvalence et de sa capacité à représenter efficacement la structure et le comportement d'un système logiciel.
UML offre une variété de diagrammes et de notations pour modéliser différents aspects d'un système logiciel, tels que la structure statique, le comportement dynamique, les interactions entre les composants, et bien plus encore. Ces diagrammes, tels que les diagrammes de cas d'utilisation, de classes, de séquence, d'activité et d'état, permettent de visualiser et de documenter efficacement les différents aspects de notre plateforme, facilitant ainsi la compréhension, la conception et la communication. Dans notre cas, il nous a permis de réaliser le diagramme de cas d'utilisation pour illustrer de façon brève les acteurs impliqués dans notre système et les fonctionnalités qui leur sont liées. À base de cela, nous avons réalisé les diagrammes de classe et de séquence pour mieux avoir un aperçu de l'implémentation des fonctionnalités principales du système.

2.3.3 Environnements de développement

Le tableau ci dessous compare quatres Environnements de Développement (IDE) populaires : Visual Studio Code, IntelliJ IDEA, Eclipse et NetBeans. Les critères d'évaluation incluent la convivialité, la compatibilité, les fonctionnalités, le support de la communauté, la popularité, et la performance.

Critères	Visual Studio Code	IntelliJ IDEA	Eclipse	NetBeans
Convivialité	Excellente	Très bonne	Bonne	Bonne
Compatibilité	Excellente	Bonne	Excellente	Excellente
Fonctionnalités	Très bonne	Excellente	Bonne	Bonne
Support de la communauté	Excellente	Bonne	Excellente	Bonne
Popularité	Très populaire	Populaire	Populaire	Moins populaire
Performance	Très bonne	Excellente	Bonne	Moyenne

FIGURE 2.7 : Comparaison des environnements de développement

Visual Studio Code a été choisi comme environnement de développement intégré (IDE) en raison de sa convivialité et de sa flexibilité. Il offre une interface intuitive qui facilite la prise en main, même pour les développeurs moins expérimentés. Grâce à sa large gamme d'extensions et à sa compatibilité avec divers langages de programmation, Visual Studio Code permet une personnalisation poussée pour s'adapter aux besoins spécifiques du projet. De plus, sa performance exceptionnelle et son support actif de la communauté garantissent des mises à jour régulières et des améliorations constantes. Ces caractéristiques en font un outil idéal pour le développement d'applications web dans un environnement collaboratif.

Conclusion

Dans ce chapitre, nous avons étudié les besoins de notre plateforme, présenté les diagrammes UML utilisés ainsi que les choix techniques effectués. Le chapitre 3 exposera le prototype de l'application et formulera quelques critiques.

Chapitre 3

Prototype de l'application et discussion

Introduction

En réunissant l'ensemble des besoins pour la réalisation de notre solution nous sommes parvenu a un résutat que nous illustrerons dans ce chapitre. Nous allons présenter les différentes interfaces de la plateforme en les décrivants et exposer quelques limites de notre solution.

3.1 Présentation du prototype de la plateforme

La plateforme de gestion d'environnement de test de vulnérabilités Web permettra de passé des tests de sécurité sur des systèmes, dans des conteneurs qui seront exécutés dans le navigateur en toute liberté sans exposé des données réelles.

3.1.1 Page d'accueil de notre plateforme



FIGURE 3.1 : Interface d'accueuil

Nous avons l'interface d'accueil sur laquelle tombent tous les types d'utilisateurs initialement avant l'authentification. Dessus, nous exposons quelques cours de type écrit et de type environnement de test à titre illustratif. Mais ces cours ne sont accessibles qu'après authentification.

3.1.2 Authentification

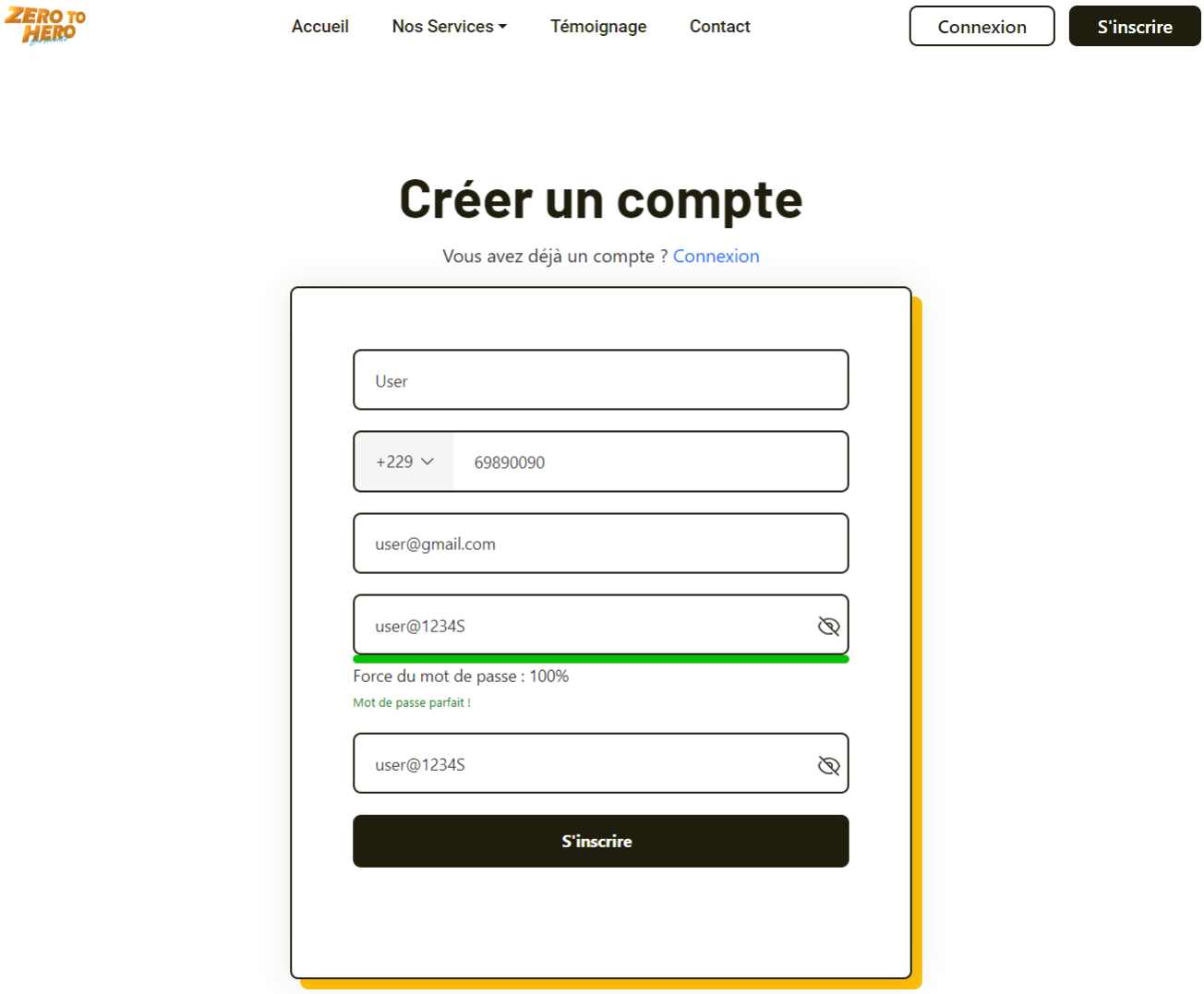


FIGURE 3.2 : Interface d'authentification

Pour l'authentification il faut s'inscrire en soumettant un formulaire qui envoie le lien de conexion par email pour validé le compte crée. Par défaut l'utilisateur créé est un étudiant et tombe sur le dashbord étudiant. C'est a l'admin de décider de son statut si il doit etre changé.

3.1.3 Affichage du dashboard admin

3.1.3.1 Affichage de la page des statistiques

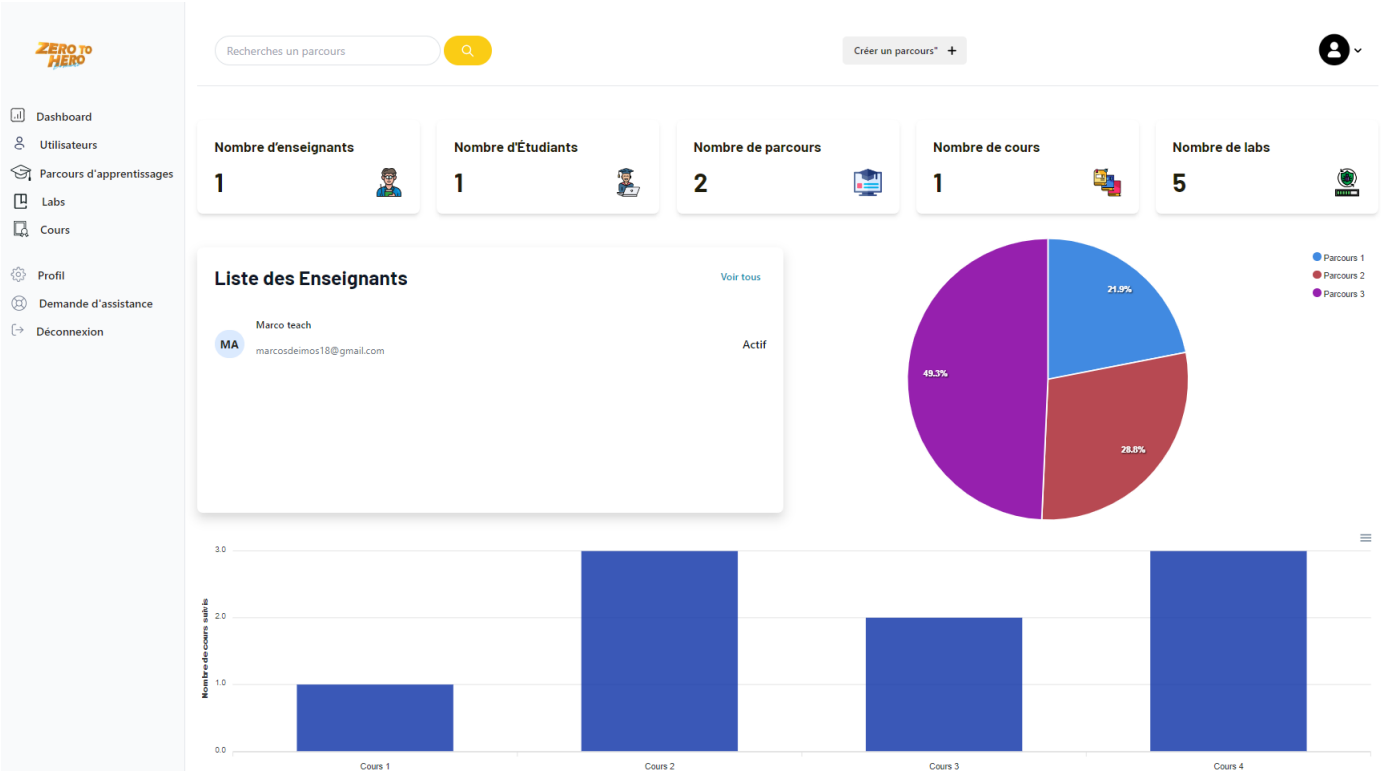


FIGURE 3.3 : Interface des statistiques admin

Une fois authentifié, si l'utilisateur a le rôle d'administrateur, il sera redirigé vers le tableau de bord administrateur où il pourra consulter les statistiques des parcours et des utilisateurs de la plateforme.

3.1.3.2 Gestion des utilisateurs

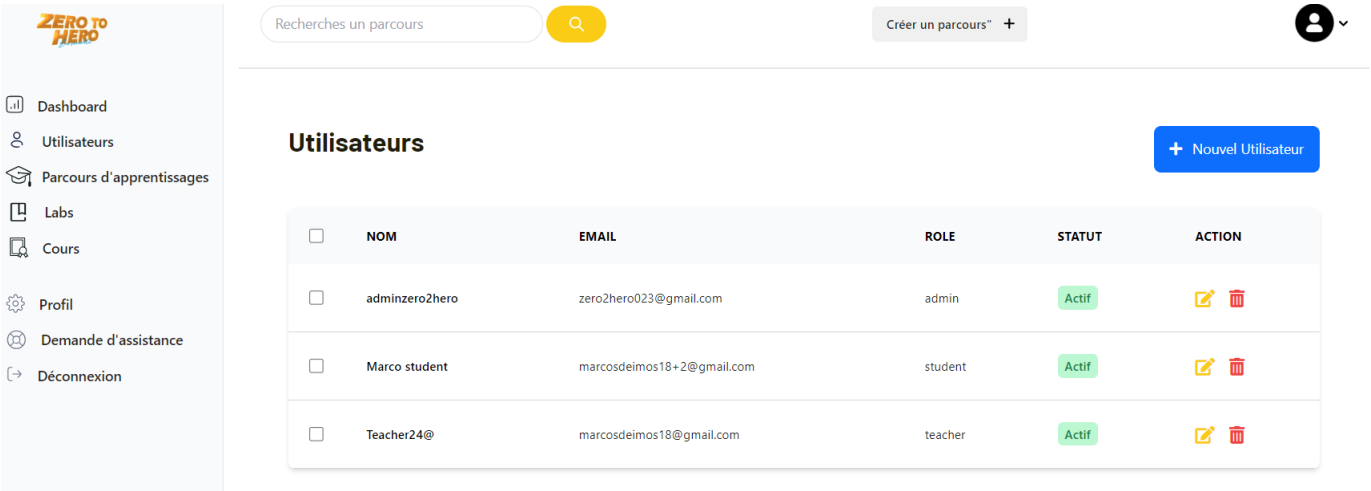


FIGURE 3.4 : Interface de gestion des utilisateurs

Chaque utilisateur se voit attribuer un rôle qui peut être administrateur, enseignant ou étudiant, ainsi qu'un statut qui définit si son compte est actif ou inactif. Le **contrôle d'accès** et **les permissions** sont déterminés en fonction de ce rôle.

En effet, l'identifiant de l'utilisateur connecté et son rôle sont encryptés dans le token qui maintient la session active. Ce token est inclus dans chaque requête envoyée au système pour prouver l'identité de l'utilisateur et lui permettre d'accéder aux ressources autorisées. Pour des raisons de sécurité, ce token a une durée d'expiration de 1 heure. Afin d'éviter que la session ne se déconnecte immédiatement après l'expiration du token, nous avons mis en place un refresh token, également généré de manière aléatoire. Ce refresh token permet d'obtenir un nouveau token valide après l'expiration du token d'accès. Ainsi, le système utilise le refresh token pour renouveler la session sans nécessiter une nouvelle authentification de la part de l'utilisateur. Le refresh token a une durée de validité de 30 jours.

3.1.3.3 Affichage d'un exemple de parcours.

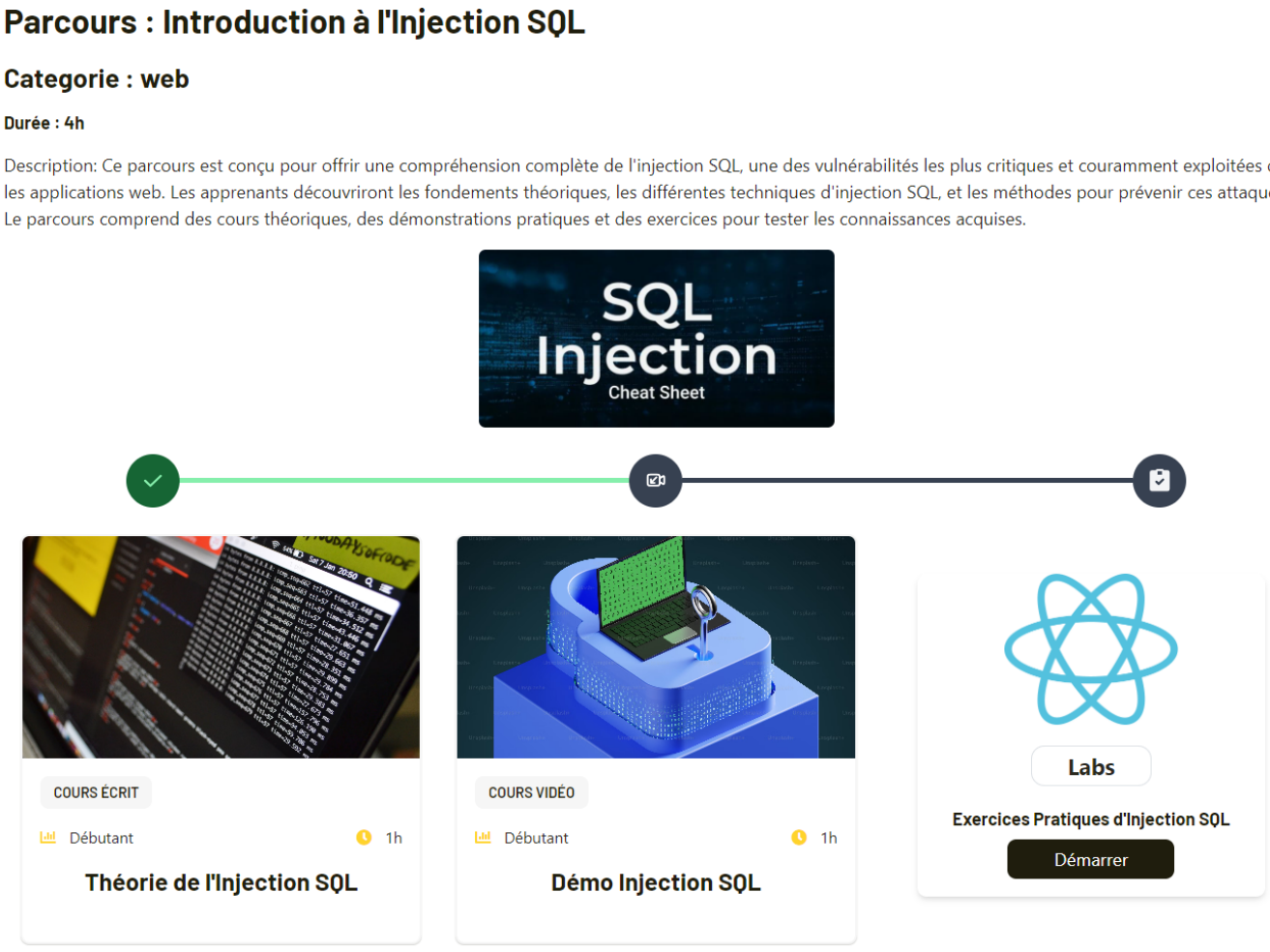


FIGURE 3.5 : Présentation d'un parcours

Seule l'administrateur est responsable de la création des parcours. Ainsi, il crée des parcours qui regroupe des cours de même catégorie de tous types (ecrit, vidéo ou conteneur), formant ainsi un parcours à suivre par l'étudiant pour obtenir une certification à la fin après avoir répondu a une série de question pour valider ses compétences.

3.1.3.4 Affichage des formulaires de création de cours de type lab.

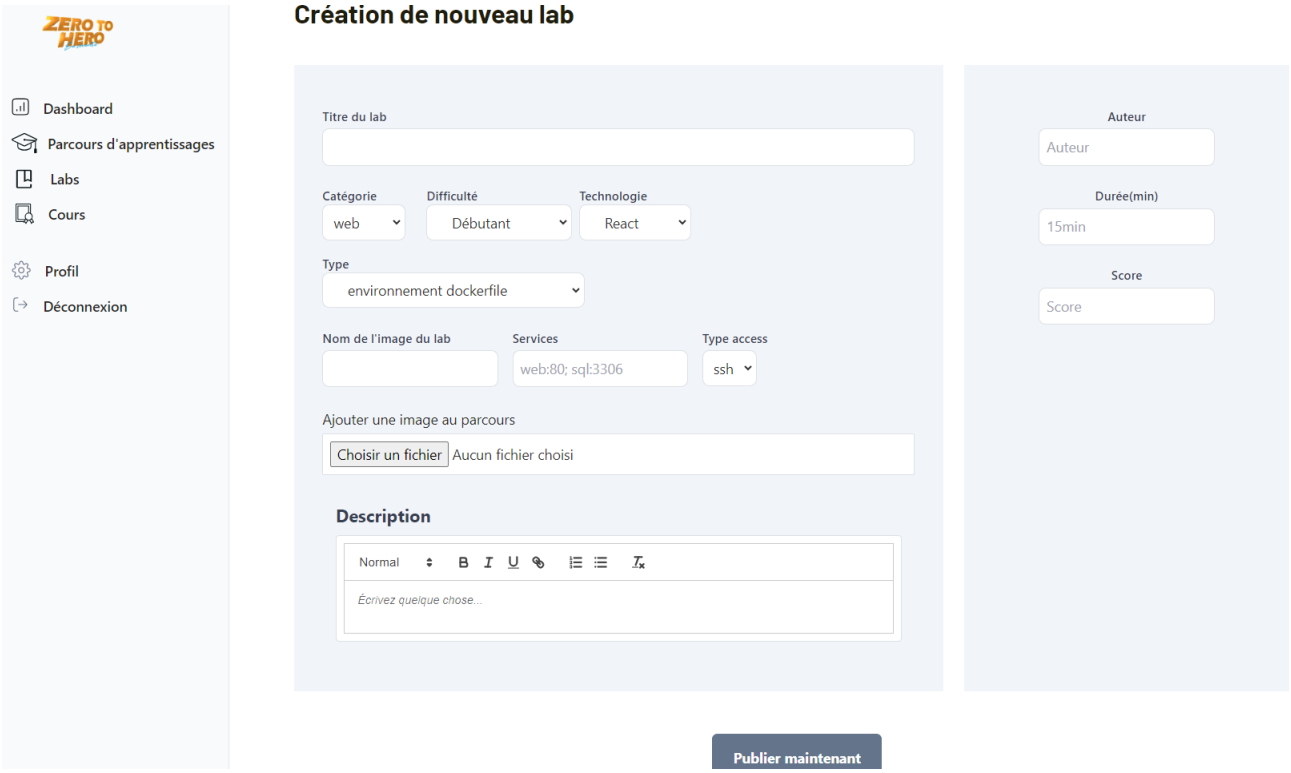


FIGURE 3.6 : Interface de création de cours de type lab

Pour la **création d'un laboratoire (lab)** permettant d'effectuer des tests de vulnérabilité sur diverses technologies, plusieurs informations doivent être renseignées. Cela inclut la technologie de l'application du lab à tester (React, Vue.js ou Angular), le type de lab spécifiant les environnements (comme Dockerfile ou Docker Compose) ou s'il s'agit d'une revue de code. Ces informations permettent de paramétrer le conteneur de manière adéquate. De plus, il est nécessaire de spécifier le type de service avec le port correspondant, l'auteur du lab et sa durée. Toutes ces informations sont envoyées au backend via une API Docker lors de la soumission du formulaire.

3.1.3.5 Contenu d'un lab.



FIGURE 3.7 : Interface du contenu d'un lab

Dans ce cours de type laboratoire (lab), le boutton run une fois cliqué envoie une requête vers le backend avec un corps contenant les informations du lab pour lancer un container.

3.1.4 Affichage du dashboard étudiant

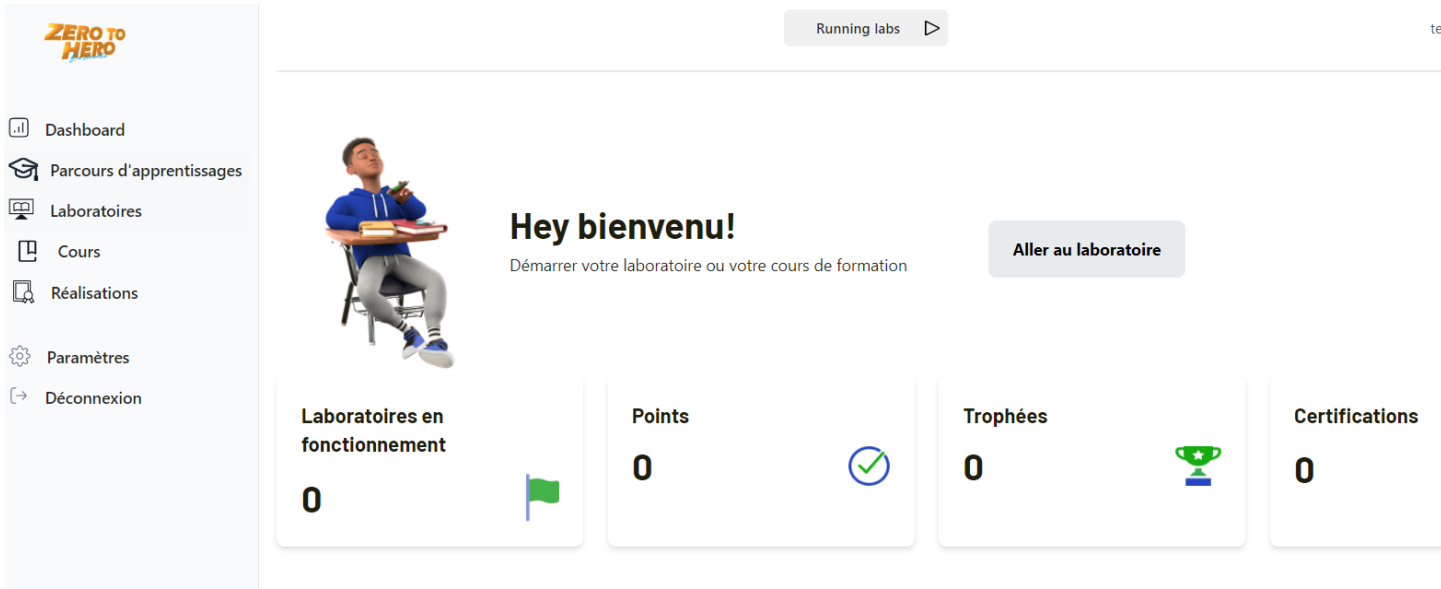


FIGURE 3.8 : Interface du dashboard étudiant

Lorsqu'un étudiant se connecte, il accède à une interface d'apprentissage conviviale qui lui permet de consulter les cours et parcours disponibles en fonction des catégories qui l'intéressent. Il peut également visualiser ses réalisations passées et suivre l'avancement des cours qu'il a commencés.

3.1.4.1 Laboratoire et test de vulnérabilités.

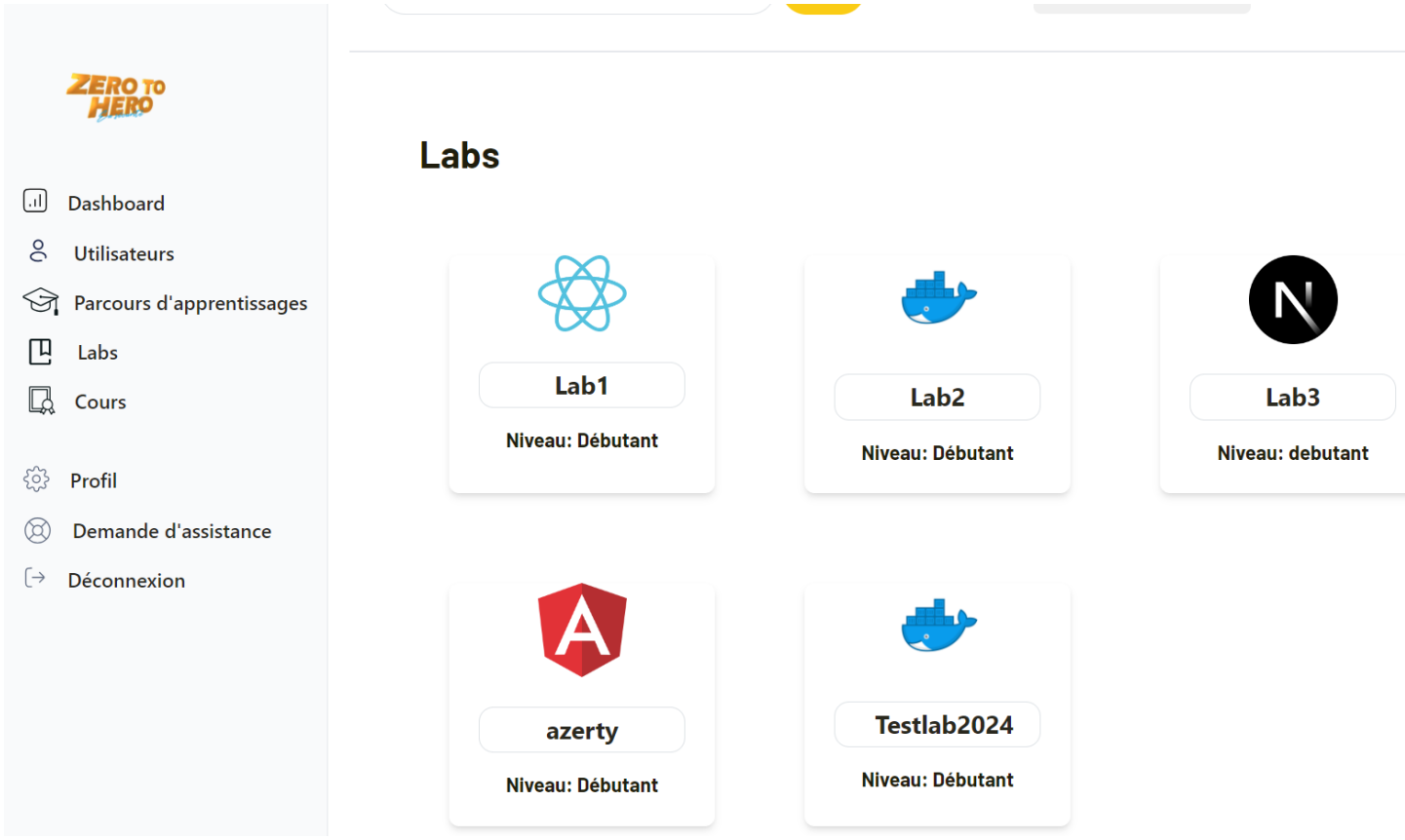


FIGURE 3.9 : Cours de type conteneur

Au niveau de la session des laboratoires, l'étudiant accède au laboratoire disponible et peut effectuer ses tests de vulnérabilités. Le démarrage d'un laboratoire fait accéder à un conteneur grâce à la commande `docker run`, permettant ainsi d'accéder à l'application du conteneur dans un navigateur.

3.2 Sécurité

Pour garantir la **sécurité des données**, nous avons mis en place un contrôle d'accès strict. Seul l'utilisateur dont l'identifiant est reconnu et valide peut accéder aux données liées à son profil.
De plus, nous avons implémenté le standard JSON Web Token (JWT) au niveau de nos API. Cela permet l'échange sécurisé des jetons d'accès qui gèrent la maintenance des sessions de chaque utilisateur. Le contenu du jeton est inclus dans l'en-tête des requêtes que chaque utilisateur effectue, ce qui permet de suivre plus facilement ses actions.
L'achitecture MERN mis en place aussi permet de sécurisé les donné et de limité les pertes d'information par injection SQL. En effet contrairement aux bases de données relationnelles traditionnelles, MongoDB n'utilise pas le langage SQL pour interagir avec la base de données. Par conséquent, les attaques par injection SQL, qui exploitent les vulnérabilités du langage SQL, ne sont pas applicables à MongoDB.Il utilise du BSON (Binary JSON) pour stocker les données, ce qui élimine le risque de manipulation des requêtes par les attaquants. De plus, l'utilisation de requêtes préparées (ou paramétrées) dans les bibliothèques de programmation MongoDB aide à empêcher l'injection de données malveillantes dans les requêtes.

3.3 Discussion

Bien que le développement d'une plateforme de gestion d'environnement de test de vulnérabilités Web présente des avantages significatifs en termes de sécurité en ligne, il est important de prendre en compte les défis potentiels liés à la complexité technique, aux ressources nécessaires, à la maintenance et à l'adoption.

3.3.1 Les points forts :

- **Demande du marché :** Avec la montée des cyberattaques et des violations de données, il existe une demande croissante pour des outils et des plateformes permettant de détecter et de corriger les vulnérabilités des applications Web.
- **Opportunité d'innovation :** La conception et le développement d'une plateforme de gestion d'environnement de test de vulnérabilités Web offrent une opportunité d'innovation dans le domaine de la sécurité informatique.
- **Impact potentiel :** En renforçant la sécurité des applications Web, notre plateforme peut potentiellement réduire les risques de cyberattaques, protéger les données sensibles et renforcer la confiance des utilisateurs dans les services en ligne.

3.3.2 Les insuffisances :

- **Ressources nécessaires :** Le développement d'une telle plateforme peut nécessiter des ressources importantes en termes de temps, de main-d'œuvre et de budget. Il est important de bien planifier et de disposer des ressources adéquates pour mener à terme le projet.
- **Maintenance et évolution :** Une fois la plateforme développée, il sera nécessaire de la maintenir, de la mettre à jour et de l'adapter aux évolutions des menaces et des technologies. Cela peut nécessiter un engagement continu et des efforts de maintenance à long terme.
- **Validation et adoption :** Il peut être nécessaire de valider l'efficacité de la plateforme et de promouvoir son adoption par les entreprises et les organisations concernées. Convaincre les utilisateurs potentiels de l'importance et de la valeur ajoutée de la plateforme peut représenter un défi.

Conclusion

Dans ce chapitre, nous avons dévoilé les aboutissements de notre travail. Les résultats issus de la phase de développement sont illustrés à travers des captures d'écran, complétées par une description de chaque étape du processus de réalisation

Conclusion Générale

L'objectif principal de notre projet était de concevoir et de développer une plateforme de gestion d'environnements de test de vulnérabilités web, offrant une expérience d'apprentissage pratique et sécurisée en matière de sécurité web. Notre démarche a commencé par une étude approfondie des besoins des utilisateurs, incluant les administrateurs et les étudiants, afin de garantir que la plateforme réponde efficacement à leurs attentes.

Nous avons ensuite procédé à la conception de l'architecture de la plateforme, en utilisant des outils tels que les diagrammes UML pour modéliser les fonctionnalités et les interactions du système. Le choix de l'architecture MERN pour le développement frontend et backend a été déterminé par sa flexibilité, sa scalabilité et sa compatibilité avec les exigences du projet.

La sécurité des données étant une préoccupation majeure, nous avons accordé une attention particulière à ce domaine en mettant en place des mesures de sécurité robustes pour protéger les informations des utilisateurs contre les menaces potentielles.

Deux tableaux de bord distincts ont été développés pour répondre aux besoins spécifiques des différents types d'utilisateurs : un tableau de bord administrateur, permettant une gestion centralisée et efficace de tous les utilisateurs et des parcours éducatifs, et un tableau de bord client, offrant aux apprenants la possibilité d'accéder aux parcours disponibles et de suivre les cours à leur propre rythme.

La plateforme propose des parcours éducatifs structurés, comprenant divers types de cours : des vidéos, des contenus écrits et des exercices pratiques utilisant des conteneurs Docker pour tester les vulnérabilités des applications web. Chaque cours se termine par un quiz permettant de valider les connaissances acquises, garantissant ainsi une consolidation des compétences.

En conclusion, nous estimons que nos objectifs de départ ont été atteints, offrant une solution innovante et efficace pour l'apprentissage en sécurité web. Pour l'avenir, nous envisageons d'approfondir le traitement des données pour fournir des analyses plus avancées et des fonctionnalités prédictives, contribuant ainsi à améliorer encore l'expérience des utilisateurs et à renforcer la sécurité de la plateforme.

Webographie

[1] aws. What is api. https://aws.amazon.com/fr/what-is/api/, Consulté le 16 mars 2024.

[2] docker. Docker. https://docs.docker.com, Consulté le 4 mars 2024.

[3] larousse. gestion. https://www.larousse.fr/dictionnaires/francais/gestion/, Consulté le 16 mars 2024.

[4] MDN. Sécurité web. https://developer.mozilla.org/fr/docs/Learn/Server-side/First_steps/Website_security, Consulté le 20 février 2024.

[5] mongodb. L'architecture mern. https://www.mongodb.com/mern-stack, Consulté le 3 janvier 2024.

[6] wikipedia. injections de code. https://fr.wikipedia.org/wiki/Injection_de_code_dans_les_applications_web, Consulté le 16 mars 2024.

[7] wikipedia. vulnérabilités web. https://fr.wikipedia.org/wiki/VulnÃl'rabilitÃl'_des_services_web, Consulté le 16 mars 2024.

Table des matières

Dédicace ii

Remerciements iii

Résumé iv

. iv

Abstract v

. v

Table des figures vi

Liste des acronymes vii

Glossaire 1

Introduction 2

1 Revue de littérature 4

Introduction 4

1.1 Généralités sur le sujet 4

1.1.1 Fonctionnement d'une plateforme de gestion 4

1.1.2 Vulnérabilités Web 4

1.1.3 Utilité d'une plateforme de gestion dans le contexte des environnements de test de vulnérabilités Web 5

1.1.4 API 6

1.1.5 Docker et API Docker 7

1.2 Gestion des ressources et sécurité 8

1.2.1 Gestion des utilisateurs, des inscriptions et des progrès des apprenants 8

1.2.2 Sécurité Web 8

1.2.2.1 Normes et bonnes pratiques en matière de sécurité Web 8

1.2.3 Études de cas de violations de sécurité Web 9

1.2.4 Aspects Juridiques et Éthiques de la Sécurité Web 10

1.2.5 Contrôle d'accès et sécurité des environnements de test 10

1.3 Analyse des travaux existants 11

1.3.1 Présentation des plateformes de sécurité 11

1.3.2 Étude comparative 11

1.3.3 Perspectives d'amélioration 12

Conclusion 13

2 Analyse, modélisation, Choix techniques et sécurité **14**
Introduction . . . 14
2.1 Analyse des besoins . . . 14
2.1.1 L'administrateur . . . 14
2.1.2 L'étudiants . . . 15
2.2 Conception . . . 15
2.2.1 Identification des acteurs . . . 15
2.2.2 Le diagramme de cas d'utilisation . . . 15
2.2.3 Le diagramme de classe . . . 16
2.2.4 Le diagramme de séquence . . . 17
2.3 Choix techniques . . . 19
2.3.1 Choix de l'architecture . . . 19
2.3.2 Le langage de modélisation . . . 20
2.3.3 Environnements de développement . . . 21

3 Prototype de l'application et discussion **23**
Introduction . . . 23
3.1 Présentation du prototype de la plateforme . . . 23
3.1.1 Page d'accueil de notre plateforme . . . 24
3.1.2 Authentification . . . 25
3.1.3 Affichage du dashboard admin . . . 26
3.1.3.1 Affichage de la page des statistiques . . . 26
3.1.3.2 Gestion des utilisateurs . . . 26
3.1.3.3 Affichage d'un exemple de parcours. . . . 27
3.1.3.4 Affichage des formulaires de création de cours de type lab. . . . 28
3.1.3.5 Contenu d'un lab. . . . 29
3.1.4 Affichage du dashboard étudiant . . . 29
3.1.4.1 Laboratoire et test de vulnérabilités. . . . 30
3.2 Sécurité . . . 30
3.3 Discussion . . . 31
3.3.1 Les points forts : . . . 31
3.3.2 Les insuffisances : . . . 31
Conclusion . . . 31

Conclusion **32**

Bibliographie **33**

Webographie **33**

Table des matières **34**