Rapport d'activité de fin d'alternance

Réalisé par

Ulrich KEMKA TAKENGNY

Développement et optimisation des plateformes de simulation pour les scénarios de sécurité et défense : *IntelLab*

Diplôme préparé : **Manager de Solutions Digitales et Data**

Sous la supervision de : **Jonas RENAULT**, *Chef de projet informatique*

Année Académique 2023-2024

Table des matières

Table des matières ii

Liste des Figures iii

Introduction 1

1 Présentation de l'entreprise 2
- 1.1 Présentation du centre de recherche . . . 2
  - 1.1.1 Inria : métiers et chiffres clés. . . . 2
  - 1.1.2 Le service de l'apprenti : Mission Défense et Sécurité . . . 2
- 1.2 Principales activités . . . 3

2 Présentation des projets - Analyse - Résultats obtenus 4
- 2.1 Projet IntelLab : Application Web . . . 4
  - 2.1.1 Contexte du projet . . . 4
  - 2.1.2 Moyens mis à la disposition des équipes de développement . . . 4
  - 2.1.3 Outils de gestion de projet . . . 5
  - 2.1.4 Réalisation du projet . . . 6
  - 2.1.5 Stratégie de tests . . . 8
- 2.2 Projet IntelLab : Serveur de messagerie . . . 10
  - 2.2.1 Contexte du projet . . . 10
  - 2.2.2 Outils de gestion de projet . . . 10
  - 2.2.3 Réalisation du projet . . . 11
- 2.3 Projet Détection et reconnaissance de véhicules militaires sur des images et vidéos (DetReco) . . . 14
  - 2.3.1 Context du projet DetReco . . . 14
  - 2.3.2 Méthodologie . . . 14

3 Analyse des défis et des enseignements 21
- 3.1 Difficultés rencontrées . . . 21
- 3.2 Solutions apportées . . . 21

3.3 Leçons apprises 22

Conclusion **23**

Liste des Figures

2.1 Planification des issues 5
2.2 Architecture Application Web IntelLab 8
2.3 Rapport test frontend avec VITEST 8
2.4 pipeline des tests automatiques 9
2.5 Liste des conteneurs de Mailu 13
2.6 Fiftyone dataset 16
2.7 Exemples transformation d'images de véhicules militaires 19

Introduction

Dans le cadre de ma formation en alternance, j'ai eu l'opportunité de participer à plusieurs projets au sein du centre de recherche Inria, plus précisément dans le département Mission Défense et Sécurité. Mon rôle en tant que Développeur Web m'a permis de contribuer à des initiatives qui combinent des approches avancées en technologie web et en intelligence artificielle pour répondre à des problématiques concrètes liées à la sécurité et à la défense.

Au cours de cette expérience professionnelle, j'ai été impliqué dans le développement de l'application web du projet IntelLab. Cette plateforme permet de simuler divers scénarios de sécurité, dont l'exploitation du renseignement d'intérêt militaire et la détection des signaux faibles précurseurs d'actions d'ingérence.

En parallèle, j'ai participé à la mise en place d'une autre plateforme du projet IntelLab spécifique au scénario de type sécurité économique, qui se focalise sur le déploiement d'un serveur de messagerie. Cette plateforme est utilisé pour des exercices simulant des scénarios de sécurité économique, recréant des environnements d'entreprises telles que des startups ou des équipes de recherche.

Enfin, j'ai contribué au projet de détection et reconnaissance (DetReco), proche du temps réel, de véhicules militaires sur des images et vidéos. Ce projet vise à optimiser les algorithmes de Deep Learning pour la détection et la reconnaissance en temps réel de véhicules militaires dans des images et vidéos.

Ces trois projets, dont l'objectif principal est de faire le lien entre la recherche académique et les besoins opérationnels des acteurs de la défense, constituent la pierre angulaire de ce rapport. Ils illustrent comment des solutions technologiques avancées peuvent être développées, testées et optimisées pour répondre à des défis spécifiques dans le domaine de la sécurité et de la défense.

Chapitre 1

Présentation de l'entreprise

1.1 Présentation du centre de recherche

1.1.1 Inria : métiers et chiffres clés.

L'Inria est l'institut national de recherche en sciences et technologies du numérique, crée en 1967, il dispose de 11 centres et plus de 20 antennes et emploie 2600 personnes. La recherche de rang mondial, l'innovation technologique et le risque entrepreneurial constituent son ADN. Au sein de 215 équipes-projets en général communes avec des partenaires académiques, plus de 3 900 chercheurs et ingénieurs y explorent des voies nouvelles. 900 personnels d'appui à la recherche et à l'innovation contribuent à faire émerger et grandir des projets scientifiques ou entrepreneuriaux qui impactent le monde.

L'institut fait appel à de nombreux talents dans plus d'une quarantaine de métiers différents, travaille avec de nombreuses entreprises et a accompagné la création de plus de 180 start-ups. Inria soutient la diversité des voies de l'innovation : de l'édition open source de logiciels à la création de startups technologiques (Deeptech).

1.1.2 Le service de l'apprenti : Mission Défense et Sécurité

Le renforcement des partenariats avec la sphère Sécurité et Défense de l'État est une priorité stratégique de l'Inria. C'est de ce contexte qu'est né le département. Créé en mars 2020 et dirigé par Frédérique Segond, la Mission Défense et Sécurité a pour objectif le soutient des politiques gouvernementales qui visent la souveraineté et l'autonomie stratégique numérique de l'Etat français, voire européen. Elle fédère tous les projets sécurité défense d'Inria bientôt de toute la France. L'équipe en pleine croissance est actuellement composée de seize personnes ayant chacun un rôle bien définit avec le soutien des intervenants externes à la mission.

1.2 Principales activités

Sous la supervision du responsable informatique, j'ai développé divers outils pédagogiques et technologiques visant à faciliter la prise en main des applications et à enrichir la compréhension des scénarios d'exercice de simulation. Mon travail a contribué à l'élaboration et à l'optimisation de nouvelles technologies au sein de l'équipe.

J'ai principalement participé au développement et à l'évolution de la plateforme IntelLab, un projet clé permettant de simuler des scénarios de sécurité variés. Nous avons travaillé en mode projet avec la méthode Agile, ce qui nous a permis d'assurer une gestion efficace et un suivi optimal des projets.
Mes contributions se sont réparties sur trois projets distincts :

- **Développement de l'application web IntelLab** : J'ai pris en charge le développement des interfaces web Frontend et Backend de la plateforme, intégrant les dernières technologies pour garantir une performance et une expérience utilisateur optimales.
- **Mise en place d'un serveur de messagerie** : En parallèle, j'ai contribué au déploiement d'une plateforme dédiée à la simulation de scénarios de sécurité économique, centrée sur la gestion d'un serveur de messagerie. Cette plateforme est utilisée dans des exercices de simulation reproduisant des environnements d'entreprises tels que des startups ou des équipes de recherche.
- **Détection et reconnaissance, proche du temps réel, de véhicules militaires sur des images et vidéos** : J'ai participé à un projet visant à développer et affiner des algorithmes de Deep Learning pour la détection et la reconnaissance en temps quasi réel de véhicules militaires dans des images et vidéos.

Dans le cadre de ces projets, mes responsabilités incluent :

- **Développement web Frontend et Backend** : Conception et implémentation des fonctionnalités sur les deux volets de l'application.
- **Développement d'algorithmes génératifs** : Création et intégration d'algorithmes destinés à enrichir les jeux de données pour améliorer les performances des modèles.
- **Rédaction des tests** : Rédaction et exécution de tests pour garantir la fiabilité et la robustesse des solutions développées.
- **Mise en production** : Déploiement des solutions dans l'environnement de production, assurant leur bon fonctionnement et leur maintenance.
- **Maintenance des applications et de leurs infrastructures** : Assurer la stabilité et la disponibilité continue des applications en identifiant et résolvant les problèmes techniques rapidement.

Chapitre 2

Présentation des projets - Analyse - Résultats obtenus

Ce chapitre est celui dans lequel nous allons présenter les projets sur lesquels nous avons travaillé, notre méthodologie de travail, leur analyse et les résultats obtenus durant notre alternance au sein d'inria.

2.1 Projet IntelLab : Application Web

2.1.1 Contexte du projet

IntelLab est un environnement de **simulation, formation, et d'expérimentation**. Il vise d'une part à faire appréhender aux académiques et aux entreprises les problèmes concrets rencontrés par les opérationnels afin d'y proposer des solutions communes, et d'autre part, de permettre d'expérimenter les solutions sur la base de procédures de tests opérationnels. Les plateformes du projet IntelLab permettent de jouer deux types de scénarios :

- Des scénarios simulant l'exploitation du renseignement d'intérêt militaire ;
- Des scénarios de type sécurité économique (détection de signaux faibles précurseurs d'actions d'ingérence).

2.1.2 Moyens mis à la disposition des équipes de développement

Dans le cadre de ce projet, l'équipe de développement est composée uniquement du responsable informatique et de moi-même. Nous avons adopté une approche *agile et collaborative* pour gérer le projet. Cette méthode nous a permis de maximiser notre efficacité en encourageant une communication régulière et une adaptation rapide aux besoins et évolutions du projet.

2.1.3 Outils de gestion de projet

Les projets sont suivis lors de réunions hebdomadaires et de séminaires semestriels, où chaque membre de l'équipe évalue les progrès, identifie les problèmes et propose des corrections pour atteindre les objectifs fixés. En complément, des échanges bilatéraux hebdomadaires entre responsables et membres de l'équipe permettent de prendre des décisions plus rapidement et efficacement, en se concentrant sur des discussions plus ciblées que lors des réunions d'équipe générales.

2.1.3.1 Définition des objectifs et des exigences du projet

Il était important de recueillir les exigences et attentes de l'application. Grâce à cela nous avons planifié notre travail en fonction des priorités et urgences. Cette planification a été faites sur GitLab sous forme d'issues différenciés par des labels (backend, frontend, bug, ...). Ces labels nous permettent de catégoriser et d'organiser les issues.

2.1.3.2 Planification des tâches

Pour le projet IntelLab, notre méthode de travail est fortement axée vers la méthodologie agile car les priorités du développement varient régulièrement en fonction du calendrier des scénarios joués ou à jouer sur les applications. Le responsable informatique Jonas Renault étant chargé de la planification, a choisi d'utiliser la plateforme de gestion de code source GitLab.

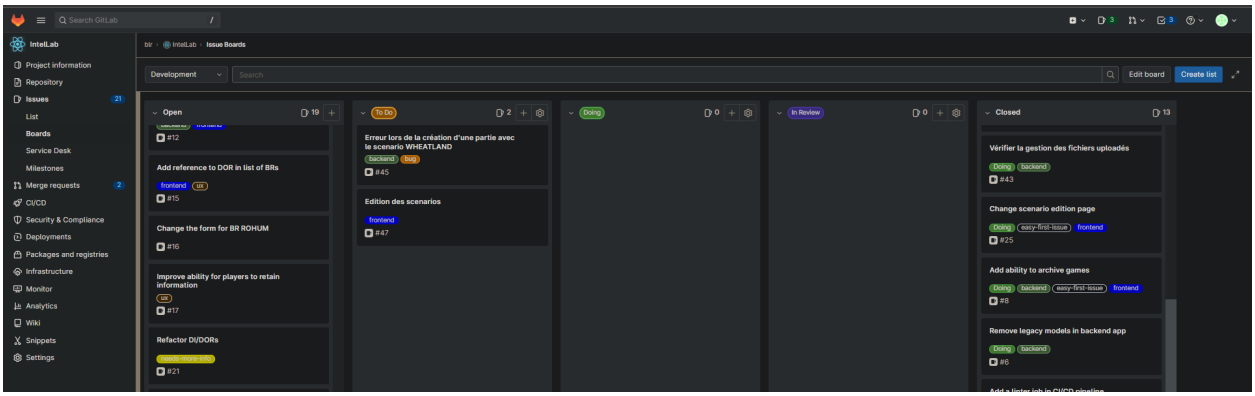


Figure 2.1: Planning issues - GitLab

GitLab est une plateforme de gestion de code source basée sur Git qui permet aux équipes de développeurs de collaborer sur des projets de logiciels, de suivre les modifications du code source et de gérer des versions de code. Elle offre des fonctionnalités pour la collaboration en temps réel, l'intégration continue et la livraison continue, la gestion de projet.

2.1.4 Réalisation du projet

IntelLab anciennement appelé **BLR (Battle Lab Rens)** a été initialement développé par un ancien ingénieur. Le BLR avait pour objectif la simulation exclusive des scénarios des services de renseignements militaires. Dans l'optique de rendre l'application plus générique afin qu'elle s'adapte à d'autres types de scénarios autres que le renseignement militaire, le responsable informatique et moi avions procédé à la refonte complète de l'application en commençant par le changement de nom.

2.1.4.1 Analyse du code existant et développement de la plateforme

Avant de commencer le développement de l'application, nous avons effectué une analyse approfondie du code et de l'infrastructure du projet. Ce qui nous a permis de mettre en place une logique de travail sur la restructuration de tout le projet.

Nous avons fait une refonte complète de l'application pour répondre à plusieurs objectifs : simplifier le code, améliorer la performance, et intégrer de nouvelles fonctionnalités demandées par les utilisateurs.

Les travaux réalisés sont:

- la suppression des parties de code redondantes ou obsolètes.
- la restructuration des API REST en utilisant des méthodes appropriées comme `GET`, `PUT`, `DELETE` et `POST` au lieu de les regrouper sous une seule méthode `POST`.
- l'isolation de chaque composant pour qu'il soit responsable d'une tâche précise, ce qui a réduit la complexité et amélioré la maintenabilité du code.
- l'utilisation des hooks, des composants en fonction au lien des classes, pour tirer parti des fonctionnalités récentes de ReactJS.
- la refonte de la navigation pour la rendre plus intuitive en introduisant des éléments interactifs plus clairs et des raccourcis qui facilitent la navigation, réduisant ainsi le temps d'adaptation des nouveaux utilisateurs.

Cette refonte a permis non seulement de rendre le code plus propre et plus facile à maintenir, mais aussi d'améliorer significativement la performance de l'application et l'expérience utilisateur.

2.1.4.2 Infrastructure des serveurs de développement et de production

Il n'existait pas de serveur de développement sur lequel nous pouvions effectuer des tests de déploiement. Nous avons mis en place un serveur de développement qui sera par la suite la réplique parfaite du serveur de production. Nous avons choisi d'utiliser Docker comme environnement d'exécution, installé sur un système d'exploitation Ubuntu Server. La conteneurisation de nos serveurs nous offre une grande flexibilité dans la gestion des composants de notre application. Ce serveur sert à vérifier le bon fonctionnement de l'application avant sa mise en production.

Les containers du backend, du frontend, du broker MQTT et de la base de données sont automatiquement installés dans l'environnement Docker des serveurs de développement et de production. Cela est possible grâce au déploiement continu depuis GitLab que nous avons configuré, ce qui nous fait gagner énormément de temps pendant le déploiement.

2.1.4.3 Description de l'infrastructure logicielle

Cet environnement est développé en utilisant les technologies suivantes :

- **Frontend** : ReactJS, gère l'interface utilisateur ;
- **Backend** :
  - *NodeJS* et *ExpressJS* pour la partie serveur web ;
  - *PostgreSQL* est le système de gestion de base de données ;
  - *Sequelize* est utilisé pour les requêtes entre le serveur et la base de données ;
  - *MQTT* : Il fournit une méthode de communication asynchrone de messages entre deux ou plusieurs appareils connectés à un réseau.
- **Infrastructure** :
  - *Docker* : Utilisé pour le déploiement de l'application sur les serveurs de test et de production ;
  - *GitLab* : notre projet y est répertorié pour le travail collaboratif, les tests unitaires, les tests d'intégration, déploiement et intégration automatique.
- **Tests** :
  - *Vitest* : Outils de gestion des tests côté frontend ;
  - *Jest* : Outils de gestion des tests côté backend.

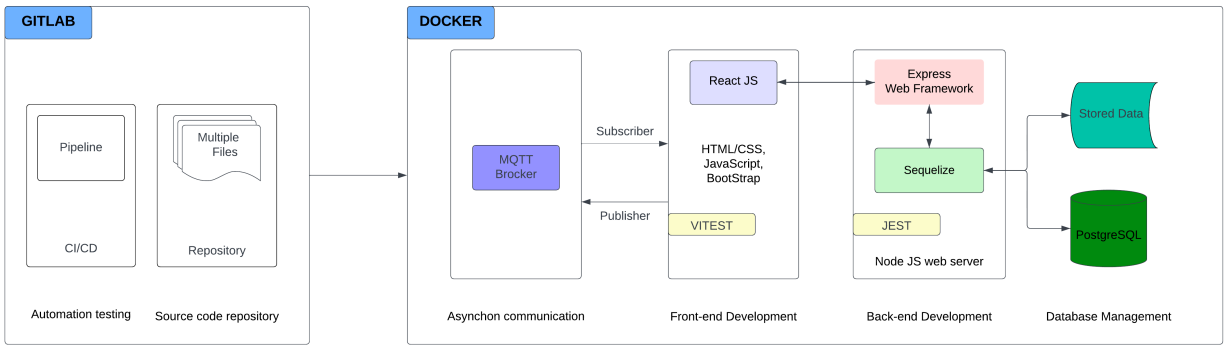


Figure 2.2: Architecture Application Web IntelLab

2.1.5 Stratégie de tests

En tant qu'environnement de simulation, formation et expérimentation, la fiabilité et la qualité d'IntelLab sont d'une importance capitale pour atteindre les objectifs visés par cette plateforme. IntelLab est une application de recherche, donc nous n'avons pas les mêmes attentes ni besoins en termes de stratégie de tests.

2.1.5.1 Objectifs et critères de tests

Les tests sur IntelLab visent à assurer que l'application répond aux exigences fonctionnelles, en permettant des simulations réalistes et fiables des scénarios de renseignement. Ils visent également à garantir la stabilité et la sécurité des données, à détecter et corriger les anomalies pendant les exercices, et à améliorer l'ergonomie et la convivialité de l'application pour une meilleure expérience utilisateur.

2.1.5.2 Mise en place des outils de tests

- **Outils de gestion des tests** : Nous utilisons l'outil de gestion des tests **vitest** coté frontend et **jest** coté backend pour suivre nos cas de test, les résultats des tests et les anomalies détectées.

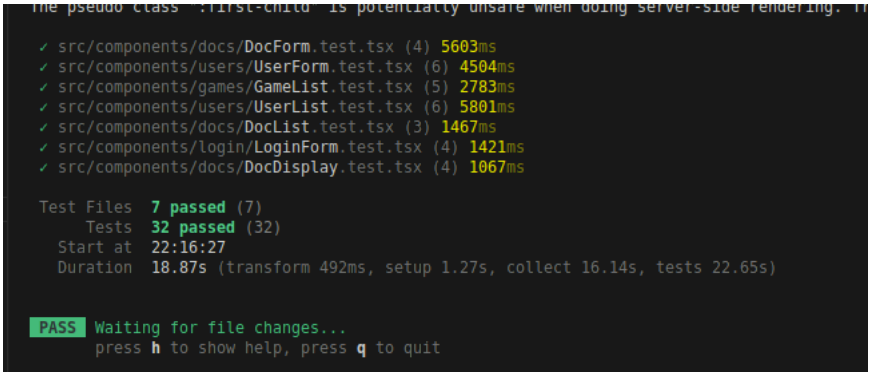


Figure 2.3: Rapport tests avec VITEST

- **Outils d'automatisation des test** : Nous utilisons les pipelines de GitLab pour automatiser l'exécution des tests.

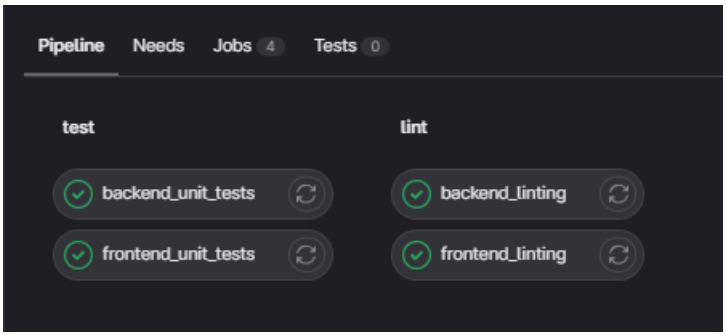


Figure 2.4: pipeline des tests automatiques

2.1.5.3 Rédaction des procédures de tests

- **Scénarios de test** : Chaque scénario est accompagné de données de test spécifiques et de critères de succès clairement définis.
- **Procédures de test** : Nous documentons les procédures de test pour chaque type de test afin de garantir une exécution cohérente.
- **Critères d'acceptation** : Nous établissons des critères d'acceptation clairs pour chaque scénario de test, définissant ce qui est considéré comme un test réussi.

Menu Name	Description	Test data	Expected Output	Actual Output
Login	should render a sign in button	-	Se connecter	Se connecter
Login	should display required helper text	Username: empty Password: Empty	Le nom d'utilisateur est requis. Un mot de passe est requis.	Le nom d'utilisateur est requis. Un mot de passe est requis.
Login	should login user	Username: john Password: john-john	Connexion réussie	Connexion réussie
Login	should handle server errors	Username: john Password: john-pass	Incorrect email or password	Incorrect email or password

Table 2.1: Scénarios de test

2.2 Projet IntelLab : Serveur de messagerie

Ce projet a beaucoup de similitudes que celui de l'application Web. Nous allons ressortir quelques parties avec leur différences :

2.2.1 Contexte du projet

Le contexte initial de ce projet est pratiquement le même que celui de l'application web, la différence réside au niveau du type de scénario. Ce projet permet la simulation des scénarios de types sécurité économique. Ce type de scénario a exigé le développement d'un autre d'environnement car diffère du monde du renseignement militaire. Cette plateforme est née du fait qu'on doit recréer un environnement d'entreprise telles que des startups ou des équipes de recherche. Les objectifs sont la formation et la sensibilisation à la détection de signaux faibles précurseurs d'actions d'ingérence.

Pour accomplir cela, nous avons mis en place un serveur d'envoi de mails qui simule l'envoi de messages électroniques entre des personnages fictifs au sein de cet environnement. Des scripts ont été créés pour initialiser les boîtes mail, gérer les sauvegardes, et générer ainsi qu'envoyer des emails. Ce système permet de recréer de manière réaliste les échanges typiques au sein d'une entreprise, afin de mieux préparer les utilisateurs à identifier des menaces potentielles dans un contexte réel.

2.2.2 Outils de gestion de projet

Comme pour les autres projets, nous sommes une équipe de deux personnes, le responsable informatique et moi. Au vu de cela, nous avons toujours adopté la méthodologie agile car cela nous permet de gérer efficacement nos ressources limitées, de nous adapter rapidement aux changements de priorités, et de livrer des fonctionnalités clés du serveur de messagerie de manière itérative et continue, en garantissant une meilleure réactivité aux besoins des utilisateurs finaux.

2.2.2.1 Planification des tâches

Comme pour l'autre projet, nous avons centralisé la gestion de ce projet sur la plateforme GitLab. La description des taches, la gestion du code, collaboration en temps réel, l'intégration continue et le déploiement automatique, toutes ces fonctionnalités sont gérées sur la plateforme GitLab.

2.2.3 Réalisation du projet

2.2.3.1 Etude de l'existant

Dans le cadre du projet de serveur de messagerie, nous avons étudié l'infrastructure existante. Cette infrastructure présente un serveur de messagerie utilisant *HmailServer*, avec *Mozilla Thunderbird* et *Outlook* pour les utilisateurs finaux.

L'ancienne infrastructure reposait sur l'installation d'HmailServer sur un PC dédié en local, non utilisé par les joueurs, et nécessitait la configuration de répéteurs WiFi pour permettre la connectivité des utilisateurs au réseau de messagerie. Les adresses IP et les noms de domaine devaient être manuellement configurés pour garantir la connexion des PC clients au PC contenant le serveur de messagerie. La gestion des comptes de messagerie était effectuée via HmailServer, où chaque domaine et adresse étaient créés manuellement avant d'être ajoutés dans Mozilla Thunderbird et ou Outlook.

Cette approche offrait une solution locale robuste, mais nécessitait une configuration minutieuse et fastidieuse, notamment lors de changements d'adresses IP ou de la configuration des serveurs SMTP dans Thunderbird. Nous avons également souligné des limitations, telles que la nécessité d'ouvrir certains ports spécifiques pour que le serveur puisse fonctionner correctement sur certains PC, ce qui impliquait l'intervention de la DSI.

L'étude de cette infrastructure existante a permis d'identifier les points forts et les limites de l'ancien système, ce qui nous guidera dans l'optimisation et la modernisation du nouveau projet de serveur de messagerie.

2.2.3.2 Nouvelle infrastructure du serveur de messagerie

Le projet de serveur de messagerie utilise une infrastructure basée sur Docker pour les environnements de développement et de production. Docker permet de conteneuriser les différents services nécessaires au fonctionnement du serveur de messagerie, offrant ainsi une grande flexibilité et une gestion simplifiée des déploiements. Cette approche permet également de maintenir des environnements cohérents entre le développement et la production, assurant une transition fluide et sans heurts.

Les services critiques, tels que le serveur SMTP, IMAP, l'interface d'administration, ainsi que les services auxiliaires comme le résolveur DNS, sont déployés dans des conteneurs Docker distincts. Cette architecture modulaire permet une maintenance facilitée et un déploiement rapide des mises à jour.

2.2.3.3 Description de l'infrastructure

Mailu est un serveur de messagerie simple mais complet, facile à configurer et à entretenir.

Cet environnement est développé en utilisant un ensemble d'images docker utilisées pour configurer et gérer le serveur de messagerie.

Les services principaux incluent :

- **Serveur de messagerie standard,** IMAP et IMAP+, SMTP et soumission avec profils de configuration automatique pour les clients.
- **Fonctionnalités de messagerie avancées,** alias, alias de domaine, routage personnalisé, recherche en texte intégral des pièces jointes des e-mails
- **Accès Web (Roundcube),** fournit les services IMAP pour l'accès aux boîtes aux lettres.
- **Fonctionnalités utilisateur,** alias, réponse automatique, transfert automatique, comptes récupérés, managesieve.
- **Fonctionnalités d'administration,** un service de filtrage de spam qui analyse et marque les emails suspects.
- **Sécurité,** TLS appliqué, DANE, MTA-STS, Letsencrypt!, DKIM sortant, scanner antivirus, Snuffleupagus , blocage des pièces jointes malveillantes.
- **Antispam,** auto-apprentissage, liste grise, DMARC et SPF, anti-usurpation d'identité.
- **Transparence,** aucun tracker inclus dans tous les composants.

La mise en place de l'infrastructure a nécessité l'utilisation de :

- **Docker :** pour déployer et orchestrer l'ensemble des services de messagerie sur les serveurs de développement et de production.
- **Poetry :** pour gérer les dépendances Python et les scripts d'administration du serveur. Voir la figure 2.5

2.2.3.4 Scripts d'administration

L'objectif principal de ce projet était d'automatiser et de simplifier la mise en place, le déploiement, et l'administration des boîtes mails utilisées dans le cadre des scénarios de simulation.

Pour cela, j'ai développé des **scripts en Python et Bash** qui permettent de :

- Configurer les comptes utilisateurs sur le serveur de messagerie.

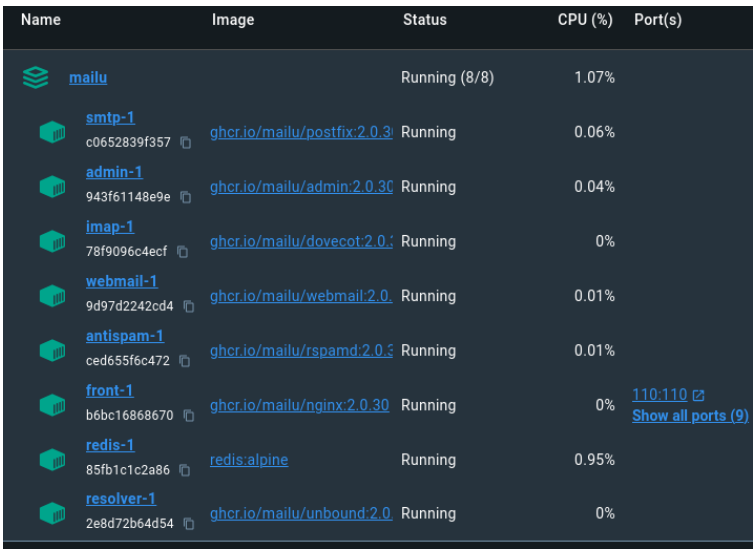


Figure 2.5: Containers Mailu

- Initialiser les boîtes aux lettres des utilisateurs avec les mails nécessaires pour le déroulement des scénarios.
- Sauvegarder les mails des utilisateurs à la fin de chaque session de jeu.
- Réinitialiser et reconfigurer le serveur de messagerie pour chaque nouvelle session.
- Restaurer les mails d'une session antérieure pour des sessions ultérieures, selon les besoins des scénarios.

La rédaction de ces scripts est ma principale contribution au projet. Ils ont été spécifiquement conçus pour répondre aux exigences du projet, en assurant une administration fluide et efficace des boîtes mails au sein de l'environnement de simulation.

2.2.3.5 Avantages de la nouvelle infrastructure

La nouvelle infrastructure présente plusieurs avantages par rapport à l'ancienne :

- **Flexibilité et Scalabilité :** L'utilisation de Docker pour conteneuriser les différents services permet une grande flexibilité dans la gestion des composants du système. Cette approche facilite le déploiement, la mise à jour, et la maintenance des services, tout en permettant de faire évoluer l'infrastructure selon les besoins sans l'interruption des opérations en cours.
- **Automatisation et Gestion Simplifiée** : L'intégration des scripts d'administration, que j'ai développés en **Python et Bash**, a considérablement simplifié la gestion des comptes utilisateurs, l'initialisation des boîtes mails, la sauvegarde, la réinitialisation

et la restauration des données sur le serveur de messagerie. Ces scripts automatisent des tâches importante, telles que la création des comptes et la gestion des boîtes mails, qui auparavant auraient nécessité une intervention manuelle complexe et fastidieuse. Grâce à ces scripts, l'administration du serveur de messagerie est plus rapide et plus fiable, car toutes opérations sont standardisés et automatisés. Cela réduit les erreurs humaines et améliore l'efficacité opérationnelle du l'infrastructure.

2.3 Projet Détection et reconnaissance de véhicules militaires sur des images et vidéos (DetReco)

2.3.1 Context du projet DetReco

Cette mission s'inscrit dans le cadre d'une collaboration entre l'équipe STARS du centre de Sophia-Antipolis, la Direction Générale de l'Armement (DGA) et le département Défense et Sécurité de l'Inria.

L'objectif de ce projet consiste à mener une étude de l'état de l'art des algorithmes pouvant s'appliquer à la détection et à la reconnaissance, proche du temps réel, de véhicules militaires sur des images et vidéos; de proposer une implémentation de cet état de l'art fusionnant éventuellement plusieurs approches; ainsi que d'en faire une évaluation. L'évaluation consistera non seulement en une quantification globale des performances mais aussi une mesure plus fine des erreurs de traitement (faux positifs, silence).

2.3.2 Méthodologie

Dans le cadre de ce projet, nous avons implémenté un algorithme qui s'appuie sur le modèle **YOLOv8** pour la détection et l'algorithme **DeepOCsort** pour le tracking de véhicules. La méthodologie du projet DetReco repose sur plusieurs étapes clés :

1. **Préparation des données :** Extraction et annotation d'images provenant de bases de données existantes. telles qu'ImageNet, OpenImages et Roboflow, complétées par des images collectées via des outils de scraping. Ces images constituent le jeu de données de base pour l'entraînement de notre modèle de détection et de reconnaissance de véhicules militaires.

2. **Entraînement du modèle** : Utilisation des jeux de données pour entraîner le modèle YOLOv8, qui est un modèle de détection d'objets en temps réel. L'objectif est d'affiner le modèle pour qu'il soit capable de détecter et de reconnaître efficacement les véhicules militaires dans des séquences d'images et de vidéos. L'entraînement permet d'améliorer

la précision du modèle dans des environnements variés, avec des conditions de visibilité et des camouflages différents.

3. **Suivi des objets** : Le modèle entraîné est ensuite appliqué pour détecter les véhicules militaires dans des vidéo.

4. **Génération d'images** : Utilisation de techniques de génération d'images basées sur des modèles génératifs, pour créer des images supplémentaires de véhicules militaires dans diverses conditions (par exemple, sous différents angles, éclairages ou environnements). Cette étape est indispensable pour enrichir le corpus d'entraînement car il manque des images pertinentes pour certains scénarios spécifiques. En augmentant les images d'entraînement sous diverses conditions, les performances du modèle sur des situations réelles pourront s'accroître.

2.3.2.1 Définition des classes

Afin de développer un modèle capable de faire la différence entre différents types de véhicules militaires, nous allons définir des classes larges en utilisant la catégorie *Véhicules militaires par type* de Wikipédia.

Nous utiliserons 4 classes :

- **Véhicule de combat blindé (AFV)**
- **Transport de troupes blindé (APC)**
- **Véhicule du génie militaire (MEV)**
- **Véhicule blindé léger (LAV)**

2.3.2.2 Préparation des données

Prepare-01 : Nous commençons par créer un ensemble de données d'entraînement à partir d'images disponibles dans des ensembles de données de détection (dataset) d'objets open source. (ImageNet, OpenImages, Roboflow).

- **ImageNet :** Le premier dataset que nous avons utilisé est ImageNet21k. De ce dataset, nous avons pu avoir 378 images annotées de tanks.
- **OpenImages :** Dans ce dataset, nous avons réussi à charger 1246 images annotées de tanks.
- **Roboflow :** Nous disposons désormais de 1624 images, mais cela reste insuffisant pour un entraînement optimal du modèle. Pour obtenir encore plus d'images d'entraînement, nous avons charger un autre dataset annotées de véhicules militaires.

Pour cette première étape, nous avons réussi à collecter **2666 images** de tanks dont **89 images** non annotées.

Prepare-02 : Nous utilisons également des outils de scraping pour collecter davantage d'images de véhicules militaires à partir d'images Google.
Nous avons collectés 669 images de plus, ce qui nous fait un total de **3335 images**.

Ce nombre d'images nous permet de définir quatre grandes classes de véhicules militaires que notre modèle peut ensuite discriminer : **Armoured Fighting Vehicle (AFV), Armoured Personnel Carrier (APC), Military Engineering Vehicle (MEV) and Light Armoured Vehicle (LAV)**.

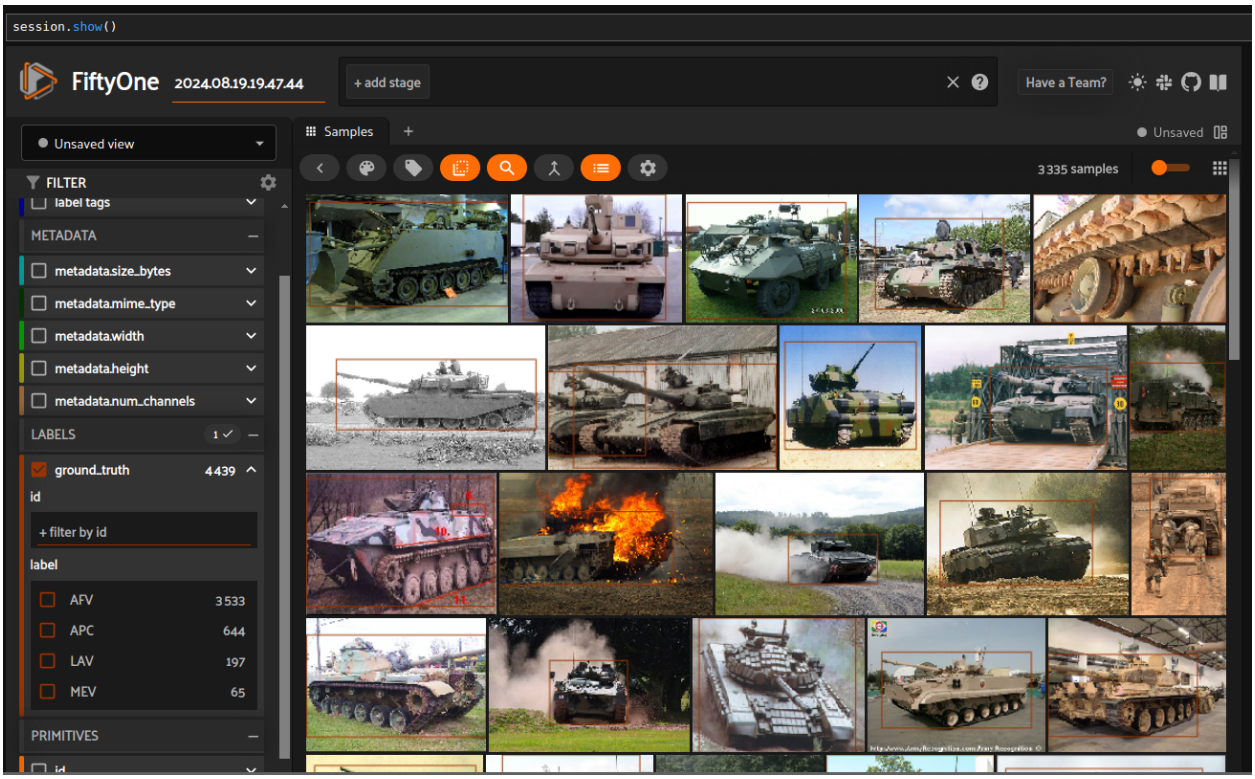


Figure 2.6: FiftyOne - Dataset (3335 images)

2.3.2.3 Entrainement du modèle

Pour la détection de véhicules militaires, en utilisant les datasets d'images annotées exportés, nous allons utiliser le model de base **YoloV8**.

De base, nous utilisons le modèle **yolov8m.pt** (medium), qui offre un bon compromis entre taille et vitesse, mais d'autres modèles que nous utiliserons pour l'expérience, sont disponibles auprès *d'ultralytics*.

Model	**Size (pixels)**	**mAP^{val} 50-95**	**Speed CPU ONNX (ms)**	**Speed A100 TensorRT (ms)**	**Params (M)**	**FLOPs (B)**
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Table 2.2: Comparaison des modèles YOLOv8

A ce niveau de l'expérience, la collecte des données du dataset d'entraînement c'est faite en deux grande étapes (Prepare-01 et Prepare-02). Nous allons présenter les résultats d'entraînement avec et sans images non annotées avec des modèles de base de différences tailles.

La précision (**mAP : Mean Average Precision**) de notre modèle sera relevé à la fin de chaque entraînement afin de suivre son évolution.

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i \tag{2.1}$$

- **N** : Représente le nombre total de classes dans le dataset.
- **AP**$_i$: Représente l'Average Precision (Précision Moyenne) pour la i-ème classe.
- **mAP** : Représente la moyenne des précisions moyennes sur toutes les classes.

Prepare-01: ImageNet, OpenImages, Roboflow (2666 images)

A cette étape, le modèle détecte uniquement une classe 'tank', donc la mAP sera une moyenne de toutes les images.

J'ai essayé plusieurs scénarios sur l'algorithme en changeant le paramètres *Epoch* (époques: passe complète à travers toutes les instances de dataset d'entraînement.) et la taille du modèles.

Modèles	Type d'images	Epoch	mAP
yolov8m	Avec images négatives	60	0.661
		80	0.655
		100	0.648
	Sans images négatives	60	0.714
		70	0.692
		80	0.700
		100	0.700
yolov8l	Avec images négative	60	0.651
		80	0.649
		100	0.638
yolov8x	Avec images négative	60	0.652
		80	0.654
		100	0.631

Table 2.3: Résultats des entraînements à l'étape **Prepare-01(2666 images)**.

Au vu de ces résultats, nous nous sommes rendu compte que la taille du modèle n'influence pas les résultats. Nous allons donc uniquement travailler avec la taille medium du modèle YoloV8. Nous allons également utiliser des datasets contenant des images annotées.

Prepare-02: ImageNet, OpenImages, Roboflow and Google (3335 images)

A cette étape, nous avons entraîné le modèle a détecter les 04 classes (AFV, APC, MEV, LAV). La moyenne mAP tient donc compte de ces 04 classes, mais nous allons aussi fournir la moyenne de chacune des classes.

`yolov8m :` - Epoch = 60 $\Rightarrow$ mAP = 0.6609

Classe	Précision	Rappel	F1-score	Support
AFV	0.76	0.85	0.80	343
APC	0.69	0.72	0.70	64
LAV	0.55	0.72	0.62	25
MEV	0.62	0.71	0.67	7
micro avg	0.74	0.82	0.78	439
macro avg	0.65	0.75	0.70	439
weighted avg	0.74	0.82	0.78	439

Table 2.4: Résultats pour `yolov8m` à Epoch 60

2.3.2.4 Suivi des objets

Après avoir entraîné notre model avec notre jeu de donnée, nous utilisons le meilleur model (**best.pt**) obtenu pendant l'entraînement et un tracker (**bytetrack.yaml**) pour tracker les

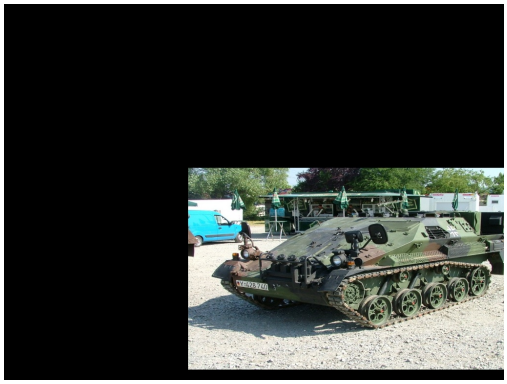
objets dans les vidéos.

2.3.2.5 Génération d'images :

Nous avons utilisé des méthodes d'augmentation des données afin d'améliorer la capacité du modèle à reconnaître des véhicules même lorsqu'ils sont éloignés ou partiellement masqués. Les méthodes utilisées incluent :

- Une transformation **scale** pour dézoomer l'image et donner l'impression que le véhicule est vu de loin. (Figure 2.7b)
- Une transformation **XYMasking** pour masquer des bandes horizontales ou verticales, simulant un véhicule caché par un obstacle. (Figure 2.7c)
- Une transformation de type **Météo** pour donner l'impression qu'il y a de la neige, de la pluie ou du brouillard sur l'image. (Figure 2.7d)

(a) Image originale



(b) Effet zoom

(c) XYMasking

(d) Météo (pluie)

Figure 2.7: Exemples transformation d'images de véhicules militaires

Les étapes spécifiques suivies sont les suivantes :

1. Dans un premier temps, nous avons récupéré une image de notre dataset avec ses labels (bounding boxes).

2. Ensuite, nous avons créé une fonction qui applique les trois transformations à toutes les images du dataset.

3. Enfin, nous avons entraîné le modèle YOLO avec ce dataset augmenté.

Après l'augmentation des images, nous avons pu collecter en plus du dataset deja existant, **8403 images**.

Prepare-03: Dataset ImageNet, OpenImages, Russian Military Annotated (Roboflow), Google et données augmentées : 8403 images

Avant de commencer les analyses, il est important de noter que les performances du modèle dépendent fortement des paramètres d'entraînement, tels que le nombre d'époques et le type de données utilisées. Le tableau ci-dessous montre les résultats obtenus pour différents modèles et paramètres d'Epoch, en utilisant un ensemble de données combinant plusieurs sources.

Modèles	**Epoch**	**mAP**	**Précision par classe**				**micro avg**
			AFV	APC	LAV	MEV	
`yolov8m`	60	0.343	0.71	0.59	0.51	0.22	0.68
	70	0.364	0.71	0.57	0.51	0.26	0.67
	80	0.405	0.73	0.54	0.51	0.21	0.67
	90	0.374	0.72	0.63	0.49	0.26	0.69
	100	0.327	0.67	0.49	0.60	0.27	0.64
`yolov8l`	80	0.360	0.69	0.53	0.45	0.24	0.65
	100	0.331	0.69	0.63	0.52	0.21	0.67

Table 2.5: Résultats des entrainements à l'étapes **Prepare-03 (8403 images)**.

Ces résultats montrent que le modèle `yolov8m` atteint ses meilleures performances avec un nombre d'époques de 80, où le mAP est maximal. L'augmentation du nombre d'époques au-delà de 80 n'améliore pas nécessairement les performances, ce qui souligne l'importance d'un bon équilibre entre surapprentissage et sous-apprentissage.

Chapitre 3

Analyse des défis et des enseignements

3.1 Difficultés rencontrées

Pendant le développement des différents projets, plusieurs défis techniques et organisationnels ont été rencontrés.

Le projet **IntelLab** a nécessité une refonte complète de l'application initiale pour s'adapter à de nouveaux scénarios, ce qui a révélé des limitations dans le code existant, ainsi que des problèmes de compatibilité lors de l'intégration de nouvelles fonctionnalités.

La mise en place d'une infrastructure de serveur de messagerie a été compliquée par la nécessité de configurer manuellement les adresses IP et les noms de domaine, ce qui était fastidieux et sujet à erreurs.

Pour le projet DetReco, la collecte de données d'entraînement suffisantes et représentatives a posé un défi, en particulier pour certaines classes de véhicules militaires sous-représentées.

3.2 Solutions apportées

Pour résoudre ces difficultés, plusieurs stratégies ont été mises en place.

La refonte de l'application IntelLab a été réalisée en adoptant une approche modulaire qui a facilité l'intégration de nouvelles fonctionnalités et amélioré la maintenabilité du code.

Concernant le serveur de messagerie, l'utilisation de **Mailu et Docker** a permis de containeriser les différents services, rendant le déploiement plus flexible et automatisé, et réduisant ainsi la complexité liée à la configuration manuelle.

Dans le cadre du projet DetReco, des techniques d'augmentation des données ont été utilisées pour compenser le manque d'images annotées, permettant ainsi d'améliorer la

précision du modèle de détection des véhicules militaires.

3.3 Leçons apprises

Ces projets ont permis de tirer plusieurs enseignements précieux.

Tout d'abord, l'importance de l'agilité dans la gestion de projets complexes a été confirmée, particulièrement dans un contexte où les priorités et les besoins évoluent rapidement.

Ensuite, la nécessité de disposer d'une infrastructure flexible et scalable s'est avérée cruciale pour garantir la réussite du projet, comme l'a montré l'utilisation de Docker pour la gestion de nos infrastructures.

Enfin, l'expérience avec le projet DetReco a mis en lumière l'importance de la qualité et de la diversité des données d'entraînement pour le développement de modèles d'apprentissage automatique performants, soulignant ainsi la nécessité de bien planifier cette étape dès le début d'un projet.

Conclusion

Les projets auxquels j'ai contribué apportent une nouvelle dimension technologique à la stratégie de sécurité et de défense, en renforçant la capacité à anticiper et à répondre aux menaces actuelles. En combinant les technologies web avancées et l'intelligence artificielle, ces projets permettent de mieux exploiter les données disponibles et d'améliorer la précision et la rapidité des réponses face à des scénarios critiques.

Ces initiatives s'inscrivent parfaitement dans la vision stratégique du département Mission Défense et Sécurité, qui vise à intégrer les avancées de la recherche académique dans des applications concrètes pour les acteurs de la défense. En développant des solutions comme IntelLab et en optimisant les algorithmes de Deep Learning pour la détection et la reconnaissance en temps réel, ces projets contribuent à la résilience et à la réactivité des systèmes de sécurité.

En somme, ces travaux permettent de renforcer la défense et la sécurité, en limitant les vulnérabilités et en optimisant l'efficacité des opérations de surveillance et d'intervention. Ils ouvrent la voie à de futures améliorations qui seront essentielles pour faire face aux défis de demain dans un contexte de menaces toujours plus sophistiquées.