Rapport de Stage

Préparé par

Habib Achraf

RT4

Entreprise d'accueil

applicare.ai

Encadrante

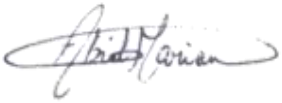
Dr Mariem Abid

Sujet

Comment peut-on assister les médecins dans la prédiction des complications chez les personnes à haut risque?

Période du stage

20/06/2024 - 20/08/2024

Responsable à l'entreprise:	*Avis de la commission des stages*
	

Année académique

2024-2025

Remerciements

Sans les efforts inestimables de nombreuses personnes, mon travail n'aurait pas été possible. Ces mots reflètent toute ma gratitude pour leur soutien indéfectible et leur contribution précieuse.

Tout d'abord, je tiens à remercier **Dr Mariem Abid**, la **PDG**, de m'avoir accordé l'opportunité d'effectuer mon stage d'été au sein de sa startup Applicare.ai et de m'avoir supervisée tout au long de ce projet. Son leadership, ses conseils avisés, son expertise dans le domaine de l'IA ainsi que son savoir faire ont été une source constante d'inspiration pour toute l'équipe, contribuant grandement à la réussite du projet **"Digital biomarkers for predicting the risk of medical complications in ICU patients"**.

Je tiens également à exprimer ma sincère reconnaissance à mon coéquipier **Achraf Habib** dont la collaboration et le dévouement ont été essentiels à la réalisation de notre partie du projet. Son engagement, ses compétences techniques, et son esprit d'équipe ont grandement contribué à notre succès commun.

Je remercie, également, tous les autres membres de l'équipe pour leur disponibilité et les précieuses explications qu'ils m'ont apportées, répondant avec patience et clarté à chacune de mes questions.

La réussite de notre travail s'est concrétisée par l'acceptation de la soumission de nos chapitres dans le livre **"AI and Data Science in Healthcare and Medicine : Applications and Future Opportunities"** de Mme. Olfa Boubaker, Professeur en ingénierie des systèmes de contrôle et en robotique à l'Institut National des Sciences Appliquées et de Technologie. Ce livre sera publié par Elsevier. Nous aurons, également, l'opportunité de présenter notre projet lors des vendredis conférences au laboratoire CDSS lab Dr Philipe Jouvet au CHU Sainte-Justine à Montréal.

Pour résumer, je suis reconnaissante à cette opportunité hors pair et ce fut un honneur pour moi de collaborer avec toutes les personnes impliquées dans cette entrprise. Toutes nos contributions ont joué un rôle crucial dans le succès de ce projet.

Table des matières

Table des figures

Introduction générale

Dans le cadre d'une convention de double diplomation entre l'Institut national des sciences appliquées et de technologie de Tunis (INSAT) et l'ENSEM, la réalisation d'un stage d'été est requise afin de mettre en pratique mes connaissances acquises et de s'intégrer dans le monde professionnel.

Passionnée par le domaine de l'intelligence artificielle, j'ai choisi d'effectuer mon stage au sein d'Applicare.ai, une startup en plein essor spécialisée dans le développement des modèles prédictifs du risque de complications pour les personnes atteintes de maladies chroniques.

Le Machine Learning, une branche de l'intelligence artificielle, a pour objectif de développer des machines capables d'apprendre de manière autonome. Il joue un rôle crucial dans notre monde axé sur les données, où les grandes quantités de données permettent aux data scientists de travailler avec des ensembles de données volumineux.

C'est particulièrement vrai dans le secteur de la santé, qui génère une quantité immense de données numériques, en utilisant les informations provenant de toutes les sources de soins de santé. Dans ce secteur, notamment dans les environnements hospitaliers, les professionnels sont souvent confrontés à un flot constant d'alarmes provenant des moniteurs de chevet. Ce volume élevé peut entraîner l'ignorance des véritables urgences, mettant ainsi en danger la sécurité des patients.

C'est dans ce contexte, que s'intègre le projet de Applicare.ai qui consiste à identifier, à partir de données collectées par des moniteurs de signes vitaux, des biomarqueurs numériques caractérisant le risque de subir une complication chez les patients à haut-risque afin de construire des modèles prédictifs précis et en temps réel.

Toutefois, pour que ces modèles soient réellement utiles, il est crucial que les médecins disposent d'interfaces utilisateur intuitives et accessibles. Ces interfaces leur permettront non seulement d'exploiter efficacement les modèles, mais aussi de suivre et de gérer les patients de manière optimale ce qui soulève la problématique suivante : **Comment peut-on assister les médecins pour garantir un suivi plus efficace de leurs patients et une détection précoce des complications chez les personnes à haut risque ?** C'est dans ce cadre, que se présente mon travail à votre bienveillante appréciation, consistant à développer une application web, concevoir et implémenter une pipeline MLOps pour gérer le cycle de vie des modèles de prédiction.

Mon rapport se compose de trois parties. La première partie est dédiée à la présentation de l'entreprise d'accueil. La seconde partie aborde l'introduction de la problématique, les concepts fondamentaux, ainsi que la méthodologie choisie pour traiter mon sujet. Enfin, la troisième partie se concentre sur les relations humaines au sein de l'entreprise.

Chapitre 1

Présentation de l'entreprise

1.1 Introduction

Dans ce chapitre, je vais présenter la startup hôte Applicare.ai, en décrivant son organisation opérationnelle ainsi que les produits et services liés à son secteur d'activité.

1.2 Identification

Applicare.ai est une start-up qui aide à prévenir les complications chez les patients à risque grâce à des modèles prédictifs basés sur la surveillance des signes vitaux.

- **Dénomination :** Applicare.ai [1]

FIGURE 1.1 – Logo de la startup

- **Forme juridique :** Entreprise scientifique
- **Date de création :** Septembre 2022 à Montréal
- **Fondatrice :** Mariem Abid (PhD experte en traitement de signal et intelligence artificielle)
- **Missions :**
  - Développement de modèles prédictifs du risque de complications pour les personnes atteintes de maladies chroniques.
  - Prédiction de maladies cardiovasculaires chez les personnes atteintes de diabète de type 2.
  - Identification des biomarqueurs digitaux associés aux complications en analysant les données des appareils médicaux de surveillance continue.

— Utilisation de techniques d'intelligence artificielle pour l'analyse des données biomédicales.

- **Clients cibles :** Medecins, professionnels de la santé, hôpitaux.
- **Partenaires :**

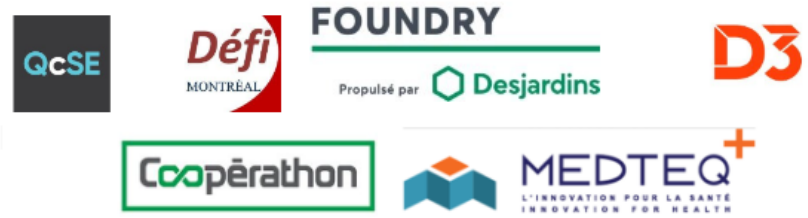


FIGURE 1.2 – Partenaires de la startup

1.3 Organisation opérationnelle

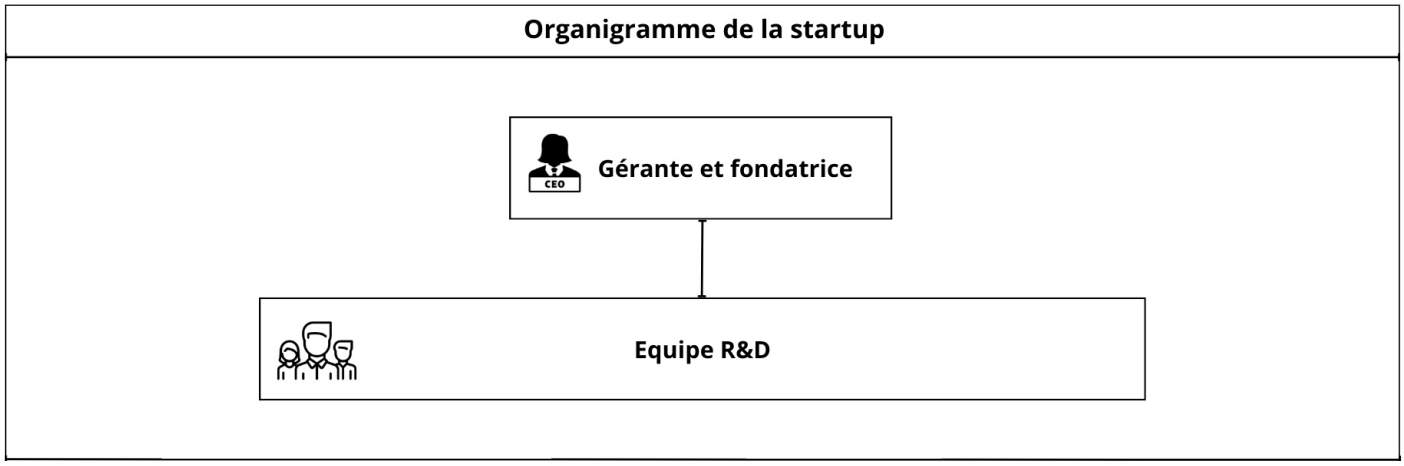


FIGURE 1.3 – Organigramme de la startup (en phase de développement)

1.4 Secteur d'activité

Cette entreprise opère dans les domaines de l'Informatique de la Santé, un domaine interdisciplinaire qui combine l'expertise médicale, l'informatique, les mathématiques et les statistiques pour analyser et étudier les données des patients. Grâce à l'intersection des soins de santé et de la technologie de l'information, en particulier l'apprentissage automatique, les data scientists et les experts médicaux ont réalisé un saut quantique dans le domaine de la médecine personnalisée. De nombreuses applications et cas d'utilisation ont émergé, parmi lesquels :

- **Prédiction et Diagnostic des Maladies :**
  Les modèles d'apprentissage automatique peuvent analyser les données des patients pour prédire la probabilité de maladies ou d'affections et aider à un diagnostic précoce, améliorant ainsi les résultats des traitements.
- **Personnalisation des Traitements :**
  En analysant les profils des patients, l'apprentissage automatique peut aider à adapter les traitements aux besoins individuels, augmentant ainsi l'efficacité et réduisant les effets secondaires.
- **Dispositifs de Santé Connectés :**
  L'apprentissage automatique analyse les données des dispositifs portables en temps réel, fournissant des informations et des alertes liées aux indicateurs de santé.

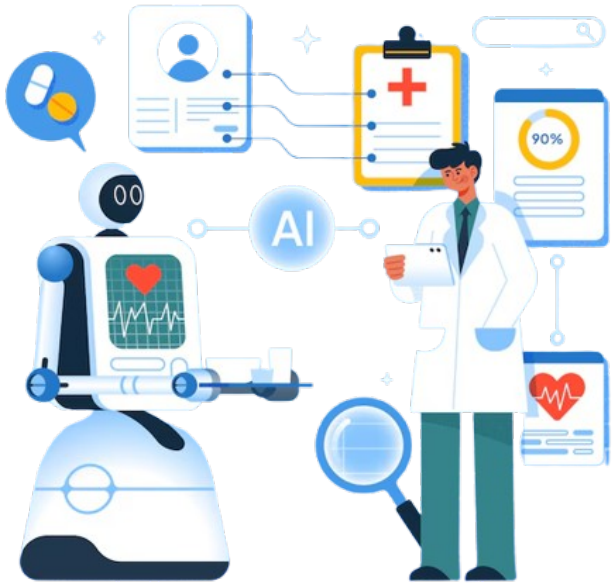


FIGURE 1.4 – Intelligence aritficielle appliquée dans le domaine médical

Chapitre 2

Déroulement du stage

2.1 Introduction

Dans ce chapitre, je vais tout d'abord exposer ma mission ainsi que le contexte de travail. Je détaillerai ensuite le travail réalisé, les défis rencontrés, les solutions que j'ai apportées pour les surmonter, et les compétences acquises au cours de cette expérience.

2.2 Description du stage

2.2.1 Problématique

Pour garantir que les modèles prédictifs développés par Applicare.ai soient pleinement fonctionnels dans un environnement clinique, il est crucial de mettre en place une pipeline MLOps efficace. Comment concevoir alors cette pipeline qui non seulement gère le cycle de vie des modèles prédictifs de manière fluide, mais aussi assure leur intégration réussie dans une application web conviviale pour les médecins ?

Les défis à relever dans ce projet comprennent la nécessité d'optimiser les performances des modèles afin de garantir des prédictions précises et rapides. Il est également essentiel de gérer efficacement les différentes versions des modèles et leurs déploiements pour maintenir une amélioration continue.
Par ailleurs, il est important de garantir que le pipeline MLOps soit suffisamment flexible pour s'adapter aux évolutions technologiques et aux changements de besoins des utilisateurs.

L'objectif est alors de concevoir la pipline MLOps qui gère le cycle de vie des modèles prédictifs et de développer une application web intuitive, permettant aux médecins de suivre et de gérer les patients de manière efficace.

2.2.2 Ma mission : cahier des charges

Ma mission dans ce projet consiste à concevoir le pipeline MLOps qui gère le cycle de vie des modèles prédictifs des complications chez les personnes à haut risque et à développer une application web pour faciliter aux médecins le suivi de leurs patients. Le cahier des charges englobe plusieurs objectifs clés :

- **Etude des modèles prédictifs développés :** Le premier objectif consiste à étudier la base de données utilisée et les modèles prédictifs développés par les autres membres de l'équipe.

- **Conception du pipeline MLOps :** Concevoir un pipeline et une architecture MLOps qui gèrent efficacement toutes les étapes du cycle de vie de l'application, depuis l'analyse des données jusqu'au déploiement des workflows et l'entraînement des modèles.

- **Développement d'ue application web :** Je vais développer une application conviviale à l'aide de Streamlit. Cette application permettra aux médecins de faire le suivi de leurs patients.

- **Déploiement sur Azure :** Mon rôle s'étend également au déploiement de l'application sur l'infrastructure Azure. Cela englobe la configuration d'un serveur optimisé, la création d'une interface utilisateur intuitive et accessible via le web, tout en garantissant une sécurité renforcée et une disponibilité continue pour les utilisateurs.
  La solution de déploiement est illustrée dans la Figure 2.1, dont les détails seront expliqués dans les sections suivantes.

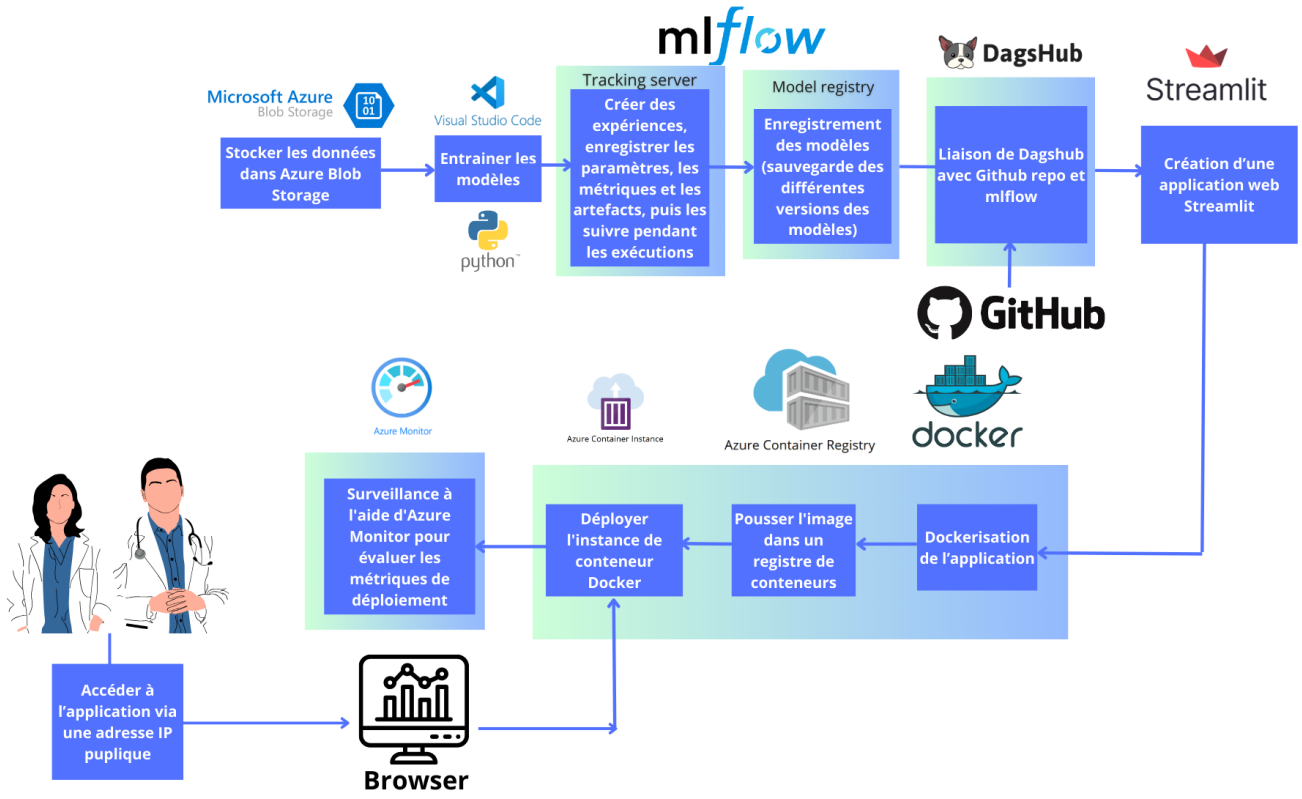


FIGURE 2.1 – Solution de déploiement

2.2.3 Les conditions de travail

Mon travail était réalisé à distance, ce qui m'a permis de gérer mon emploi du temps de manière très flexible. Nous organisions chaque semaine des réunions via Google Meet afin de discuter de l'avancement du projet, résoudre les difficultés rencontrées et établir les prochaines étapes.

En collaboration étroite avec un membre de l'équipe, nous utilisions GitHub pour la gestion du code, où chaque membre contribuait en poussant des versions mises à jour sur GitHub. Une réunion de suivi était prévue chaque jour pour répartir les nouvelles tâches et adapter nos priorités en fonction des progrès réalisés.

2.2.4 Mes relations au sein de l'entreprise

J'ai établi des liens étroits avec l'équipe de la startup. J'avais, particulièrement, une relation directe avec mon encadrante, qui est également la CEO de la startup. Elle supervisait de près mes progrès, m'offrant des conseils et un encadrement personnalisé pour me guider afin d'atteindre les objectifs du projet. Ma collaboration avec mon coéquipier, avec qui je travaillais sur la même partie du projet, était tout aussi étroite, permettant un échange constant et une coordination efficace.

Parallèlement, je communiquais avec les autres membres de l'équipe pour solliciter des éclaircissements techniques ou pour discuter des chapitres du livre, qui sera publié par Elsevier, pour m'assurer que notre contribution satisfaisait les exigences et la vision globale de l'équipe.

2.3 Concepts techniques de base

2.3.1 Les étapes du MLOps

MLOps (Machine Learning Operations) est une approche qui combine les principes de DevOps (Development Operations) avec les besoins spécifiques du Machine Learning (ML). Il vise à rendre l'intégration, le déploiement et la gestion des modèles de machine learning plus simples tout au long de leur cycle de vie, en garantissant que les modèles soient stables, reproductibles et robustes, tout en réduisant les conflits entre les équipes de développement et d'opérations.

Dans le cadre de la prédiction des complications chez les patients à haut risque, un pipeline MLOps bien conçu permet non seulement d'optimiser la performance des modèles, mais aussi de garantir que ces modèles puissent être déployés et utilisés efficacement par les médecins, via une interface web conviviale.

La figure 2.2 ci-dessous présente le cycle de vie de l'ingénierie Machine Learning (ML), intégrant les phases de développement (DEV) et d'exploitation (OPS). Elle met en évidence les étapes essentielles telles que la création, le test, la configuration, et le suivi continu, illustrant ainsi le concept d'MLOps.

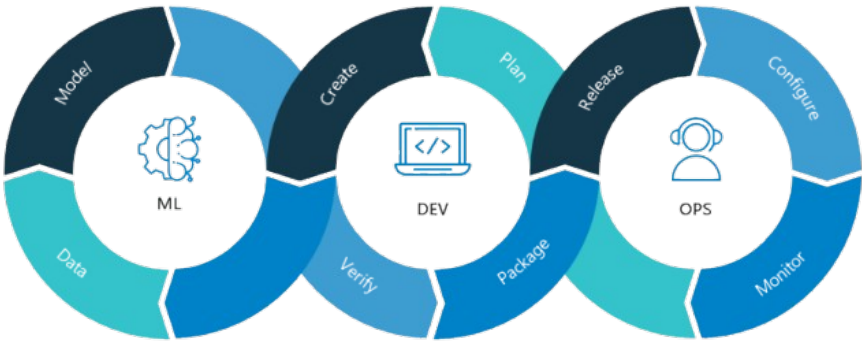


FIGURE 2.2 – MLOps

2.4 Travail réalisé

2.4.1 Recherche bibliographique et étude des modèles existants

- **Etude de la base de données utilisée : MIMIC-III**

  MIMIC-III (Medical Information Mart for Intensive Care 3) est une base de données de soins intensifs accessible au public, régulièrement mise à jour par le laboratoire de Physiologie Computationnelle du Massachusetts Institute of Technology (MIT). Cette base de données regroupe des données cliniques de patients admis dans une unité de soins intensifs entre 2001 et 2012 [12].

  MIMIC-III constitue une avancée majeure dans le domaine de la médecine de soins intensifs et de la science des données. Elle continue d'évoluer et s'enrichit avec le temps, incluant désormais des données de formes d'ondes, des notes cliniques, des images radiologiques et des données issues des services des urgences.

  Le choix de travailler avec MIMIC-III repose sur plusieurs raisons qui en font une option attrayante :

  — Cette base de données reflète des scénarios cliniques réels, permettant de développer des modèles applicables dans des contextes de soins de santé concrets.

  — Elle est à grande échelle, contenant des données cliniques détaillées et anonymisées issues de 53 423 hospitalisations distinctes de patients adultes (âgés de 15 ans et plus), ce qui permet des analyses robustes et statistiquement significatives [12].

— MIMIC-III est une base de données riche et diversifiée, incluant des informations démographiques, des mesures de signes vitaux, des résultats de tests de laboratoire, des procédures, des médicaments, et des notes cliniques. Cela offre la possibilité d'explorer divers aspects des conditions de santé.

La Figure 2.3 présente un schéma détaillé du processus de gestion des données au sein de la base de données de soins intensifs MIMIC-III, incluant les sources de données hospitalières, leur archivage, et les étapes de dé-identification avant leur intégration dans la base de données

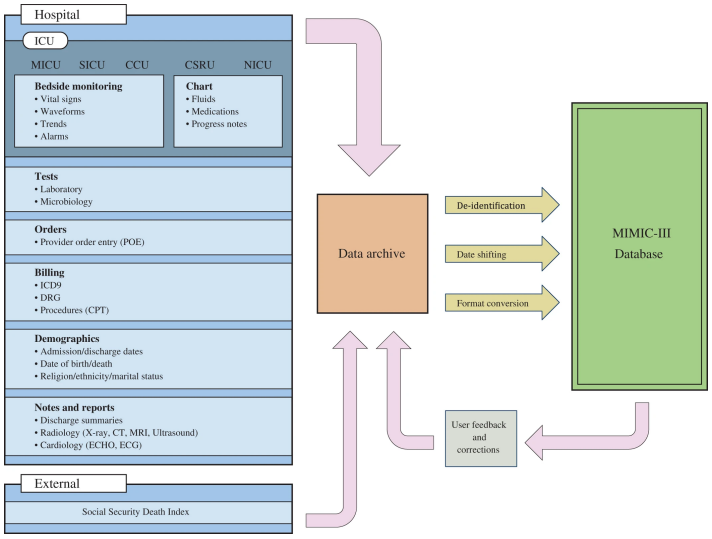


FIGURE 2.3 – Présentation de la base de données de soins intensifs MIMIC-III [12]

- **Etude des modèles développés par l'équipe**
  L'équipe de la startup a préalablement développé trois modèles différents pour la prédiction des complications chez les personnes à haut risque, en se basant sur les données de MIMIC-III. Ce travail a inclus le prétraitement des données ainsi que l'ingénierie des caractéristiques.
  Les architectures et performances de ces modèles sont présentées ci-dessous.
  — **Modèle LSTM :** LSTM (Long Short-Term Memory) est une architecture de réseau de neurones récurrents (RNN) largement utilisée en apprentissage profond. Elle excelle dans la capture des dépendances à long terme, ce qui la rend idéale pour les tâches de prédiction de séquences. **Architecture :**

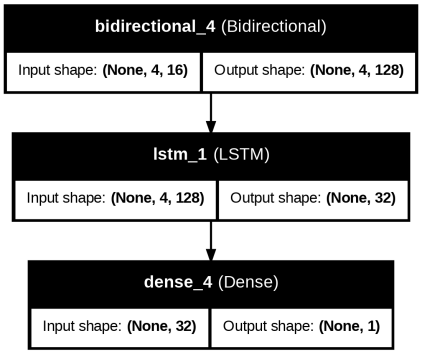


FIGURE 2.4 – Architecture de LSTM

— **Modèle GRU :** Le GRU (Gated Recurrent Unit) est une variante des réseaux de neurones récurrents (RNN) développée dans le but de résoudre le problème des gradients qui disparaissent ou explosent dans les RNN classiques. À l'instar du LSTM, le GRU a la capacité de détecter les liens à long terme dans les séquences de données. **Architecture :**

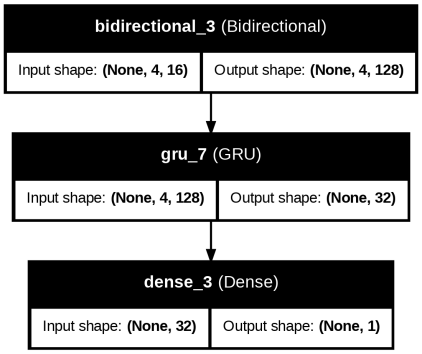


FIGURE 2.5 – Architecture de GRU

— **Custom model :** C'est un modèle personnalisé qui combine les caractéristiques statiques (comme le sexe, l'âge, la taille et le poids) avec des caractéristiques dynamiques (telles que les signes vitaux et les conditions cliniques qui changent au fil du temps). L'idée est de traiter ces deux types de données séparément avant de les fusionner pour prédire la probabilité d'une alerte d'AVC deux heures après l'observation. **Architecture :**

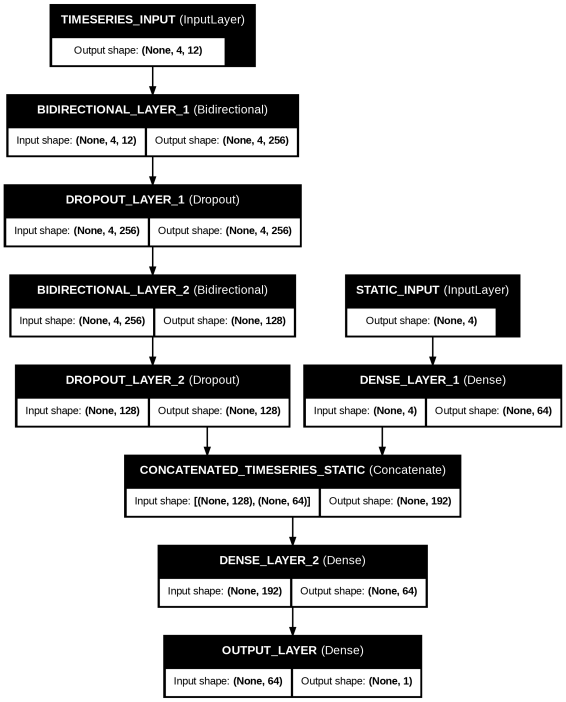


FIGURE 2.6 – Architecture du modèle personnalisé

2.4.2 Outlis et Technologies utilisés

FIGURE 2.7 – Logo de Python

- **Python :** Python est un langage de programmation interprété, interactif, orienté objet et de haut niveau. Selon son site officiel, Python "combine une syntaxe claire avec une grande lisibilité, ce qui permet aux développeurs d'écrire un code plus lisible et maintenable" (python.org). Dans ce projet, Python a été utilisé pour la mise en œuvre des modèles d'apprentissage profond, ainsi que pour le développement de l'application web avec Streamlit, permettant aux professionnels de la santé d'interagir avec les modèles déployés via une interface utilisateur intuitive [2].

- **MLflow :** MLflow est une plateforme open-source pour gérer le cycle de vie du machine learning. Selon le site officiel, MLflow "gère l'enregistrement des expériences, le packaging des modèles pour la reproductibilité, et le déploiement des modèles dans des environnements de production" (**mlflow.org**). Dans ce projet, MLflow a été utilisé pour versionner les modèles, suivre les paramètres d'entraînement et les métriques, garantissant ainsi la reproductibilité des expériences et la gestion du cycle de vie des modèles. [3]

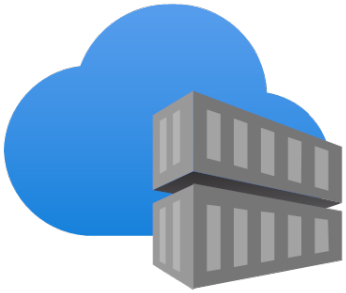
FIGURE 2.8 – Logo de mlflow

FIGURE 2.9 – Logo de Github

- **GitHub :** GitHub est une plateforme de développement et d'hébergement de code basée sur Git, qui permet aux développeurs de collaborer et de contrôler les versions de leur code. Selon GitHub, "des millions de développeurs et d'entreprises construisent, expédient et maintiennent leur logiciel sur GitHub" (**github.com**). Dans ce projet, GitHub a été utilisé pour héberger le code source et pour mettre en place une pipeline d'intégration et de déploiement continus (CI/CD) avec **GitHub Actions**, facilitant ainsi l'automatisation du processus de déploiement [4].

- **DagsHub :** DagsHub est une plateforme de collaboration pour les projets de machine learning et de science des données. Selon leur site officiel, "DagsHub centralise le contrôle de version pour les données, les modèles, et le code, en s'appuyant sur des outils open-source comme Git, DVC, et MLflow" (**dagshub.com**). Dans ce projet, DagsHub a été utilisé pour centraliser et gérer le versioning du code, des données et des modèles, facilitant ainsi la collaboration et le suivi des expériences [5].

FIGURE 2.10 – Logo de Dagshub



- **Azure Container Registry :** Azure Container Registry permet de créer, stocker et gérer des images de conteneurs et des artefacts dans un registre privé, adapté à tous les types de déploiements de conteneurs [6].

FIGURE 2.11 – Logo de Azure Container Registry

- **Azure Container Instance :** Azure Container Instances (ACI) est un service qui permet aux développeurs de déployer des conteneurs sur le cloud public de Microsoft Azure sans avoir à provisionner ni gérer l'infrastructure sous-jacente.les types de déploiements de conteneurs [7].

FIGURE 2.12 – Logo de Azure Container Instances

FIGURE 2.13 – Logo de Azure Monitor

- **Azure Monitor :** Azure Monitor est une solution de surveillance complète qui permet de collecter, analyser et réagir aux données de surveillance provenant des environnements cloud et sur site. Avec Azure Monitor, on peut optimiser la disponibilité et les performances des applications et services. Il aide à comprendre comment les applications fonctionnent et permet de réagir, manuellement ou de manière automatisée, aux événements du système [8].

- **Github Actions :** GitHub Actions simplifie l'automatisation des workflows logiciels, en intégrant une solution CI/CD de premier ordre. On peut construire, tester et déployer le code directement depuis GitHub ainsi que gérer les revues de code, les branches, et le triage des issues selon les besoins. [9]

FIGURE 2.14 – Logo de Github Actions

- **Streamlit :** Streamlit est un framework open-source de Python, conçu pour permettre aux data scientists et aux ingénieurs en IA/ML de créer des applications de données dynamiques avec seulement quelques lignes de code. Il permet de développer et déployer des applications de données puissantes en quelques minutes. [10]

FIGURE 2.15 – Logo de Streamlit

- **Docker :** Docker est une technologie de conteneurisation qui regroupe une application et toutes ses dépendances dans un conteneur léger, permettant une portabilité et une reproductibilité accrues.

FIGURE 2.16 – Logo de Docker

2.4.3 Configuration de MLflow

Dans le cadre de ce projet, MLflow a été choisi comme outil de gestion du cycle de vie des modèles prédictifs. MLflow offre une plateforme unifiée pour le suivi des expériences, la gestion des versions des modèles, et leur déploiement, facilitant ainsi l'intégration dans une pipeline MLOps complète. Sa configuration inclut plusieurs étapes clés :

- **Suivi des expériences :** MLflow enregistre les métriques, paramètres, et artefacts à chaque exécution, permettant ainsi de suivre et de comparer différentes versions de modèles.
- **Enregistrement des modèles :** Les modèles sont sauvegardés et versionnés dans un registre ce qui facilite leur gestion et leur déploiement.
- **Déploiement :** MLflow permet de déployer les modèles sur des conteneurs Docker, garantissant une transition fluide vers la production.

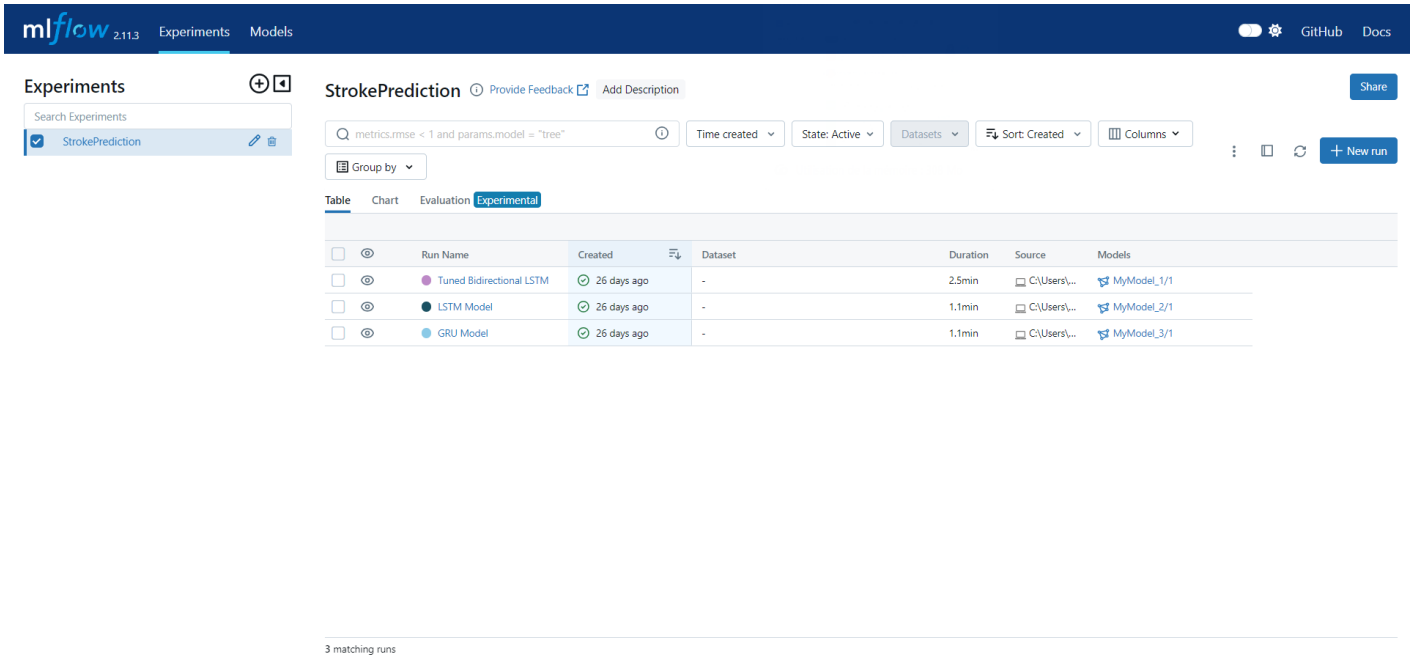


FIGURE 2.17 – Interface de MLflow

2.4.4 Configuration de DagsHub

DagsHub a servi de plateforme de gestion collaborative des données et des modèles. Elle repose sur plusieurs éléments fondamentaux :

- Il a été configuré pour utiliser Git et DVC (Controle de la version des données), ce qui facilite la version du code et des données. Cela garantit une trace exhaustive des modifications effectuées sur le code source et les ensembles de données utilisés pour l'entraînement des modèles.
- Il offre la possibilité de suivre les expériences en intégrant MLflow, ce qui permet de centraliser le suivi des paramètres, des métriques et des artefacts. Dans la configuration, il est prévu de se connecter à MLflow afin de pouvoir suivre en temps réel la plateforme DagsHub.
- Il a été mis en place afin de favoriser une collaboration efficace. Tous les membres de l'équipe ont la possibilité de suivre, de commenter et d'étudier les modifications apportées aux données et aux modèles, tout en conservant un historique précis des contributions.

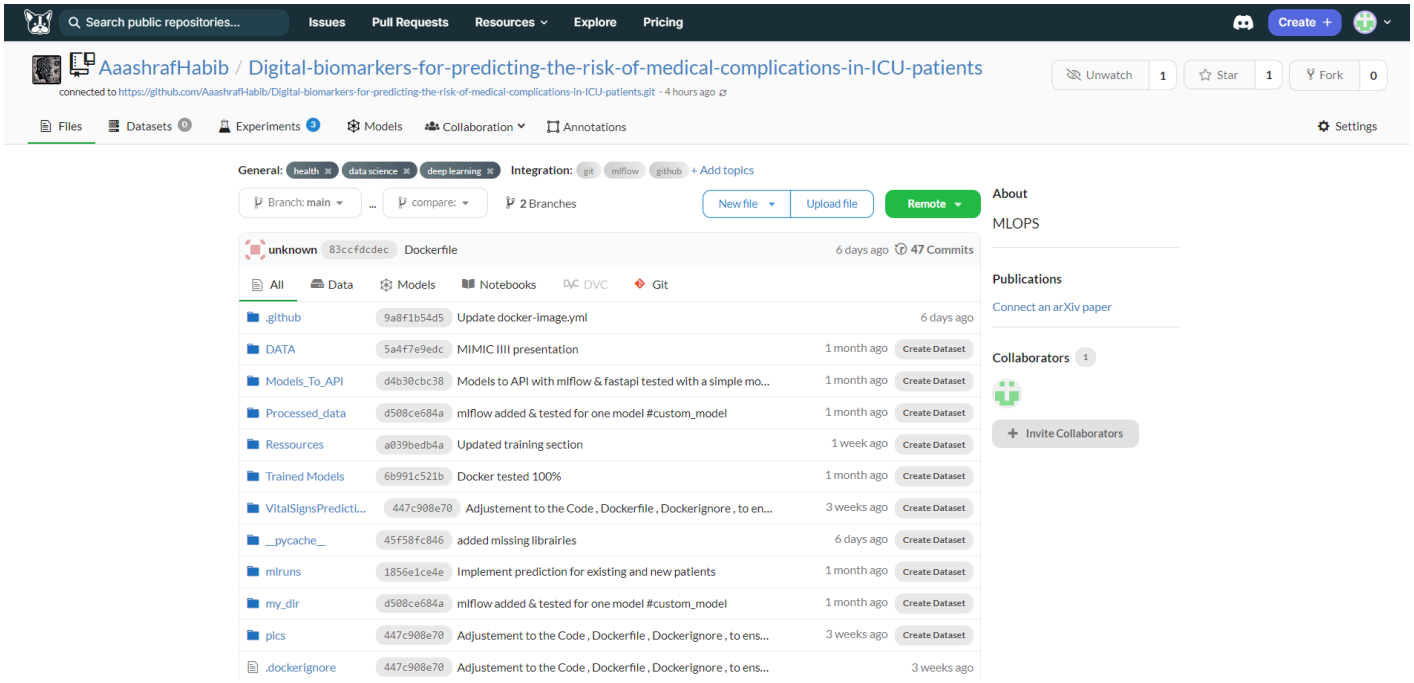


FIGURE 2.18 – Interface de DagsHub

2.4.5 Développement de l'application web avec Streamlit

L'application commence par une interface utilisateur interactive où les utilisateurs peuvent interagir avec ses différents éléments.
La première section de l'application est consacrée à la présentation de la base de données.

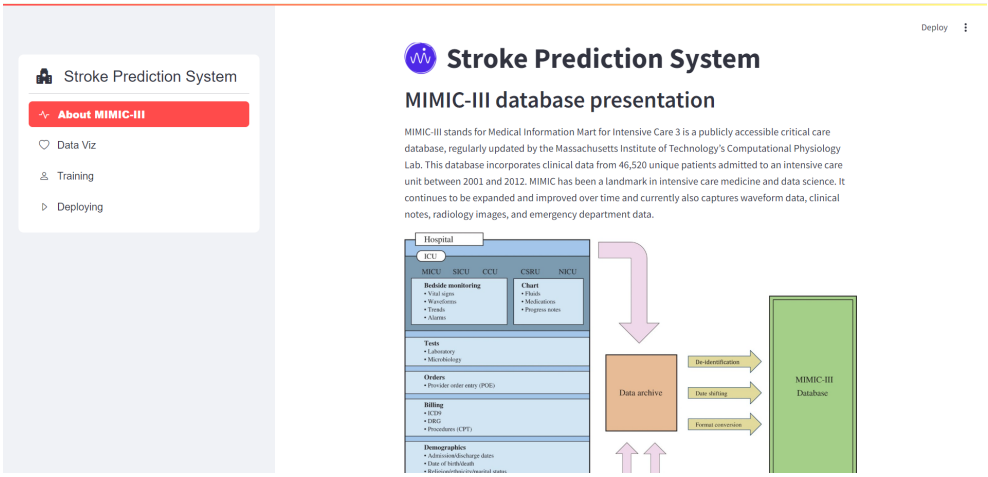


FIGURE 2.19 – Interface de l'application

Ensuite, l'utilisateur a la possibilité de visualiser le dataset, d'afficher des graphiques, ainsi que de suivre en temps réel les données dynamiques des signes vitaux comme le montrent les figures ci-dessous.

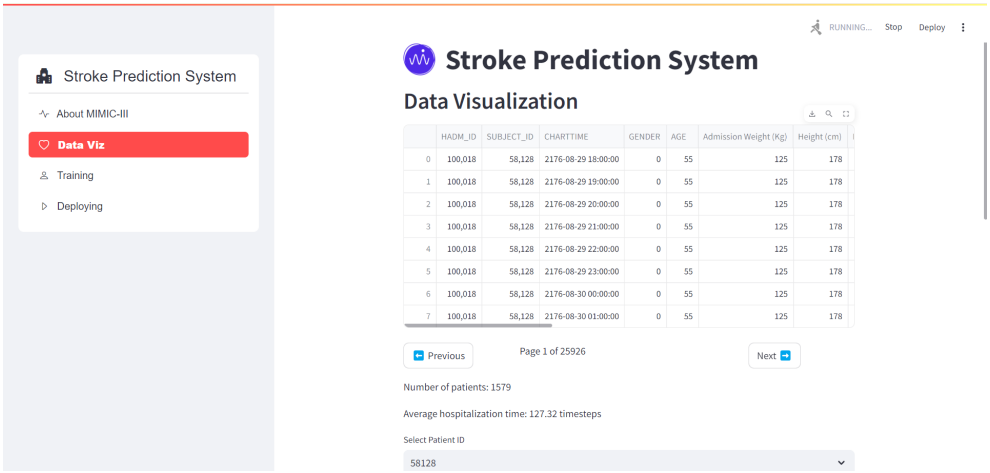


FIGURE 2.20 – Visualisation du dataset

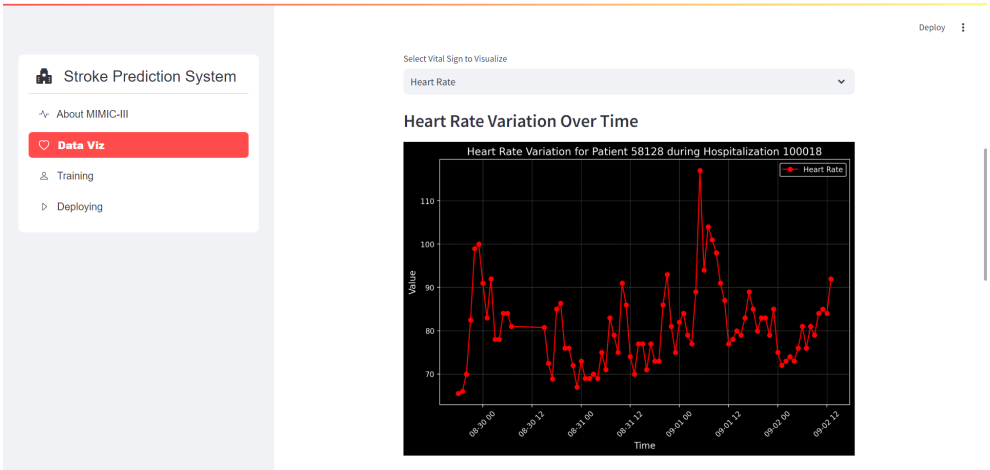


FIGURE 2.21 – Visualisation des signes vitaux en temps réel

FIGURE 2.22 – Visualisation de certains graphiques

L'utilisateur a également la possibilité de consulter les architectures des modèles existants qui sont enregistrés dans MLflow, de visualiser une comparaison de leurs performances, ou d'entraîner un nouveau modèle personnalisé, en sélectionnant les hyperparamètres, et ce modèle sera automatiquement enregistré dans MLflow pour qu'il puisse le consulter et/ou l'utiliser.
Ceci est illustré par les figures ci-dessous.

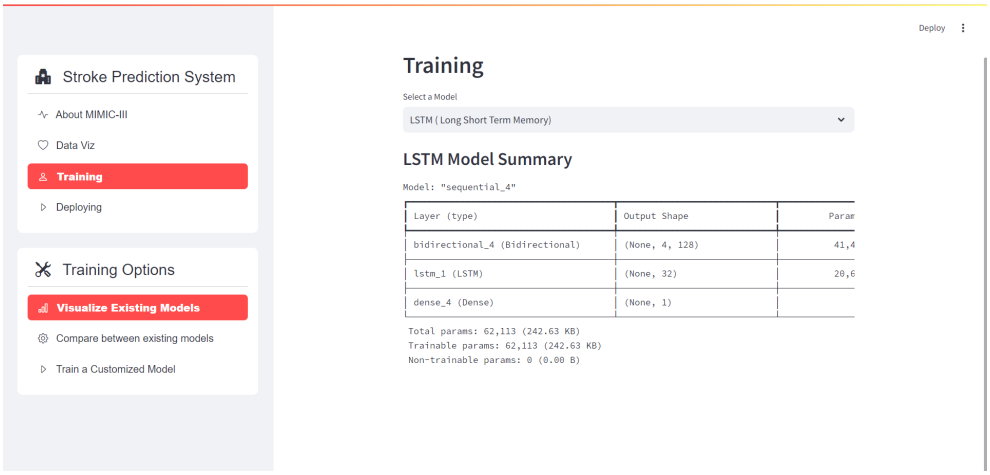


FIGURE 2.23 – Visualisation des modèles existants

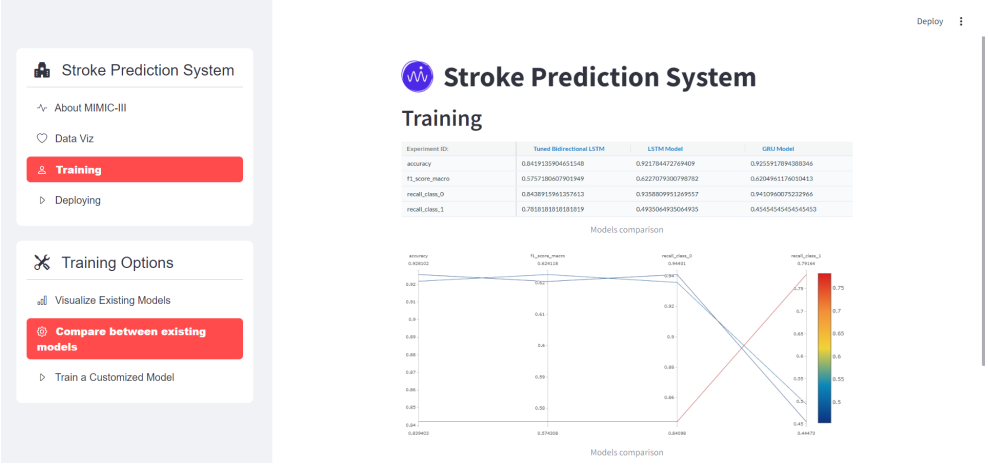


FIGURE 2.24 – Comparaison des performances

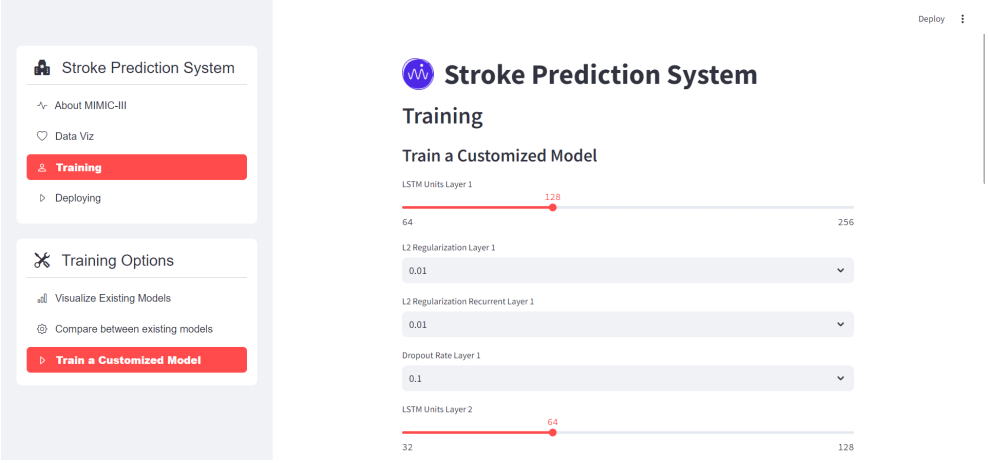


FIGURE 2.25 – Entrainer un nouveau modèle

L'utilisateur peut ensuite sélectionner l'un des modèles enregistrés dans MLflow pour prédire le risque d'AVC, que ce soit pour un patient existant ou pour un nouveau patient.

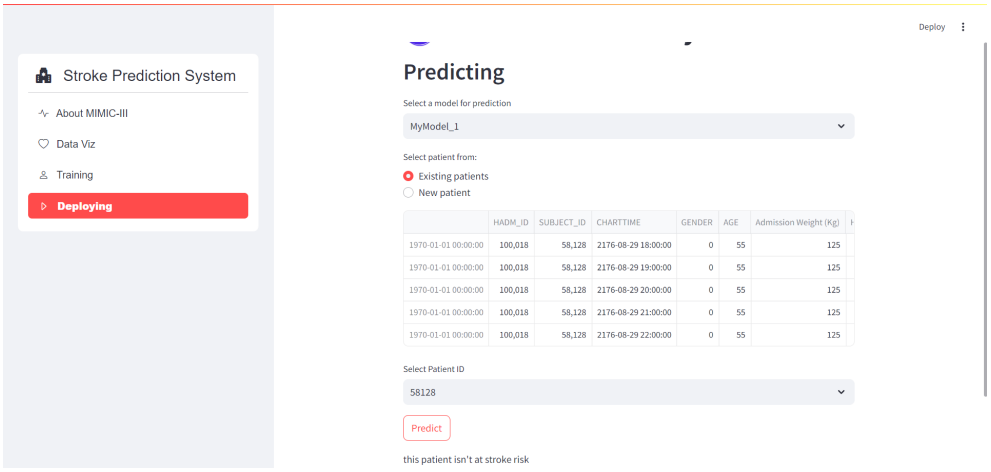


FIGURE 2.26 – Prédiction

2.4.6 Dockerisation de l'application

Dans le cadre de ce projet, on a utilisé la dockerisation pour encapsuler l'application dans un environnement isolé et reproductible ce qui a facilité son déploiement sur Azure.
L'application a été dockerisée en utilisant un Dockerfile simplifié, qui a été développé dans le but de créer un environnement d'exécution léger et efficace, en se basant sur l'image officielle de Python. Cela permet de prévenir des soucis de compatibilité et de dépendances entre diverses machines.

- **Dockerfile :** Nous avons utilisé une image légère de **Python 3.9-slim** afin de réduire la taille du conteneur et d'améliorer les performances. Toutes les dépendances nécessaires sont installées via **pip**, ce qui assure que toutes les bibliothèques, y compris Streamlit, soient correctement configurées dans l'environnement du conteneur. Ensuite, les fichiers requis pour la mise en œuvre de l'application, comme **Streamlit.py** et **Preprocessing.py**, sont copiés dans le conteneur, garantissant ainsi que seules les parties essentielles de l'application sont incluses, ce qui permet de réduire la taille de l'image.

```
Dockerfile > ...
1  # Use the official lightweight Python image.
2  FROM python:3.9-slim
3
4  # Set environment variables
5  ENV PYTHONUNBUFFERED=1 \
6      PYTHONDONTWRITEBYTECODE=1
7
8  # Set the working directory
9  WORKDIR /app
10 # Copy the requirements file into the container
11 COPY Requirements.txt .
12 # Upgrade pip and install dependencies
13 RUN pip install --upgrade pip && \
14     pip install --no-cache-dir -r Requirements.txt
15 # Copy only the necessary files
16 COPY Streamlit.py .Z
17 COPY Preprocessing.py .
18 COPY Ressources ./Ressources
19 # Expose the port the app runs on
20 EXPOSE 8501
21
22 # Command to run the app
23 CMD ["streamlit", "run", "Streamlit.py"]
```

FIGURE 2.27 – Dockerfile

- **Volumes :** Afin de réduire encore davantage la taille de l'image Docker, nous avons déployé un volume pour stocker le dataset utilisé pour l'entraînement des modèles. Grâce à cette méthode, les données sont séparées du conteneur et les ressources sont optimisées en dehors de l'image Docker, ce qui évite de recréer l'image à chaque modification du dataset.
- **Exposition des Ports :** Le port 8501 est exposé afin de faciliter l'accès à l'application Streamlit à travers un navigateur. L'application fonctionne en isolation, mais elle est toujours accessible depuis l'hôte via le port exposé.
- **Avantages de la Dockerisation :** L'utilisation de Docker nous a permis de garantir la portabilité de l'application ce qui a facilité son déploiement sur différents environnements, y compris les ceux de production comme Azure. Grâce à cette méthodologie, nous avons pu automatiser le déploiement et assurer la transition fluide entre les phases de développement et de production.

2.4.7 Déploiement sur Azure

Grâce à l'emploi d'un service cloud comme Azure, il est possible de déployer des applications de manière scalable et flexible, tout en bénéficiant de la gestion des ressources en temps réel. Cela permet d'éviter la gestion complexe des infrastructures matérielles tout en garantissant une disponibilité élevée et une gestion optimale des coûts. Nous avons opté pour Microsoft Azure dans le cadre de ce projet en raison de son intégration native avec Docker, de ses services de conteneurs et de son vaste éventail d'outils de supervision et de gestion.

- **Groupe de Ressources** Nous avons tout d'abord créé un groupe de ressources sur Azure, qui a servi de conteneur logique pour tous les services associés au projet, ce qui a facilité l'organisation et la gestion des ressources comme le montre la figure ci-dessous.

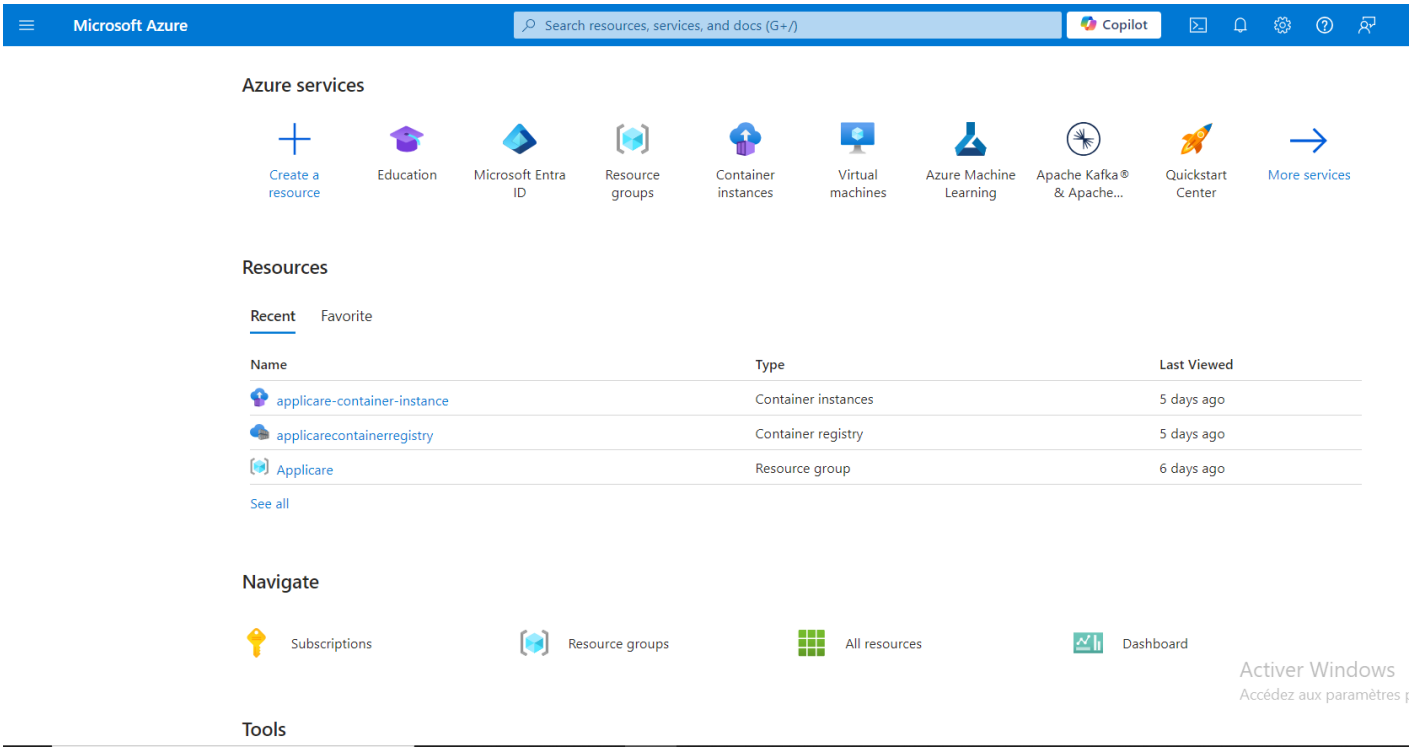


FIGURE 2.28 – Groupe de ressources

- **Azure Container Registry (ACR)** Nous avons configuré un Azure Container Registry dans le groupe de ressources créé. ACR permet de stocker et de gérer centralement les images Docker. Il facilite le déploiement d'images sur d'autres services Azure sans avoir à gérer des infrastructures externes de stockage de conteneurs, tout en assurant la sécurité et l'intégration continue.

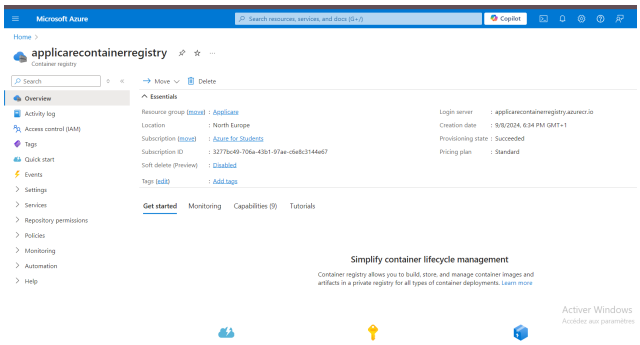


FIGURE 2.29 – Azure Container Registry

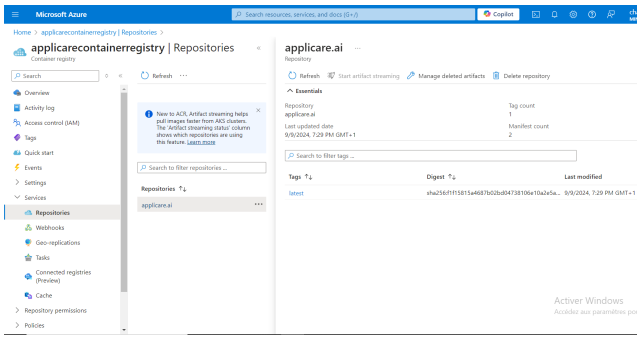


FIGURE 2.30 – Azure Container Registry Repo

- **Azure Container Instance (ACI)** Nous avons ensuite utilisé Azure Container Instance (ACI) afin d'exécuter les images Docker. Pour assurer une performance optimale et un déploiement proche des utilisateurs réels, nous avons assigné des ressources spécifiques, comme des unités de processeur, de mémoire et la région géographique, lors de la création de l'instance. Le déploiement avec Azure ACI est simplifié, ce qui facilite la montée en charge rapide sans avoir besoin de gérer des machines virtuelles.

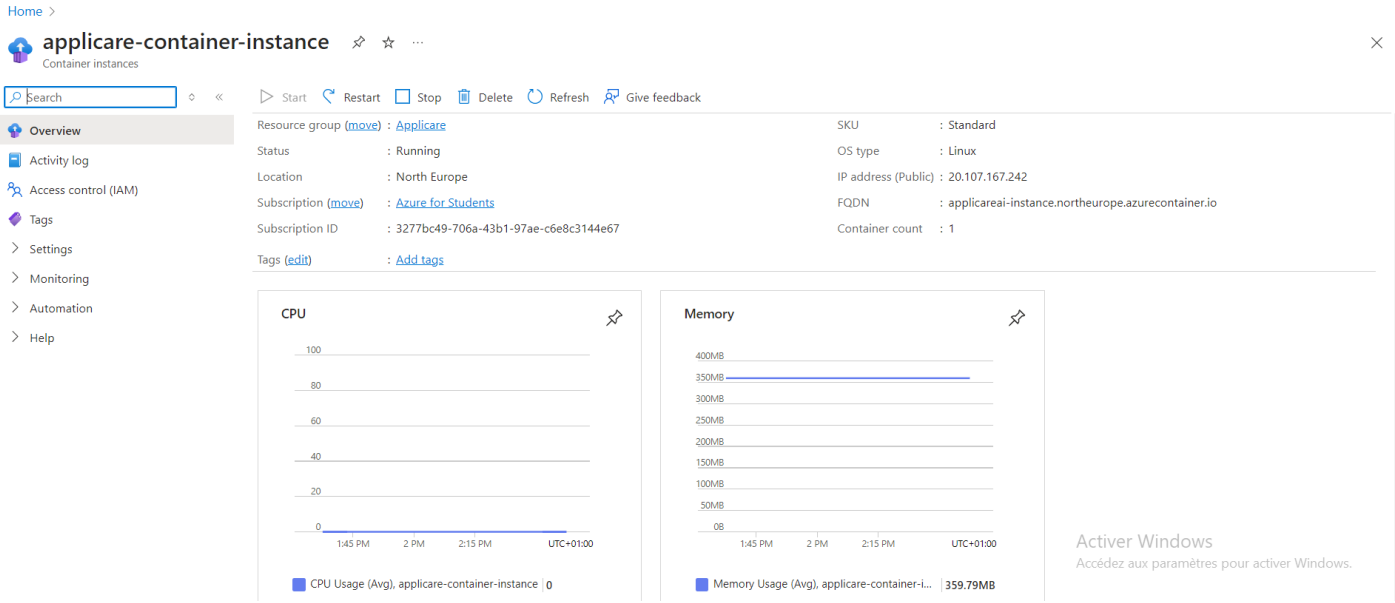


FIGURE 2.31 – Azure Container Instance

- **Azure Container Registry (ACR)** Nous avons configuré un Azure Container Registry dans le groupe de ressources créé. ACR permet de stocker et de gérer centralement les images Docker. Il facilite le déploiement d'images sur d'autres services Azure sans avoir à gérer des infrastructures externes de stockage de conteneurs, tout en assurant la sécurité et l'intégration continue.
- **Azure Container Instance (ACI)** Nous avons ensuite utilisé Azure Container Instance (ACI) afin d'exécuter les images Docker. Pour assurer une performance optimale et un déploiement

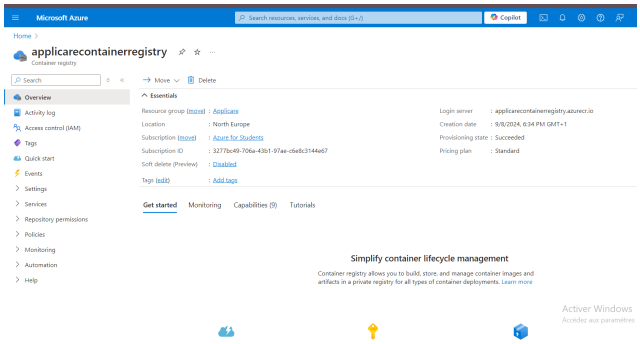


FIGURE 2.32 – Azure Container Registry

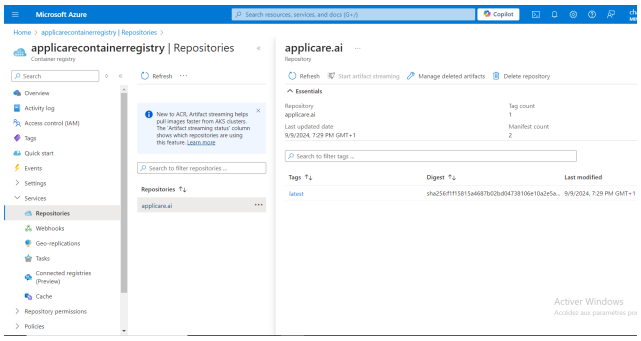


FIGURE 2.33 – Azure Container Registry Repo

proche des utilisateurs réels, nous avons assigné des ressources spécifiques, comme des unités de processeur, de mémoire et la région géographique, lors de la création de l'instance. Le déploiement avec Azure ACI est simplifié, ce qui facilite la montée en charge rapide sans avoir besoin de gérer des machines virtuelles.

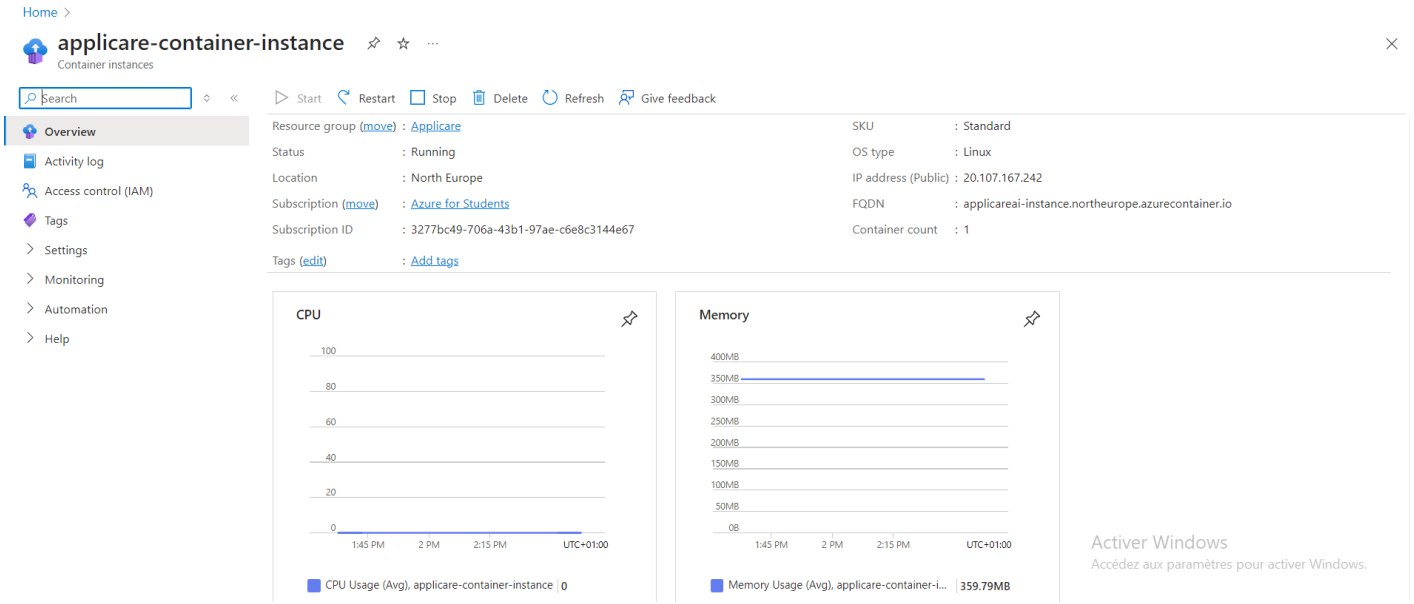


FIGURE 2.34 – Azure Container Instance

Pourquoi ACI est le plus facile et rentable :

- **Complexité :** ACI nécessite une configuration et une gestion minimales, ce qui le rend idéal pour les déploiements de petite ou moyenne taille sans avoir besoin de couches d'orchestration comme Kubernetes.
- **Coût :** Le service de paiement ACI est basé sur l'utilisation, ce qui signifie qu'on ne paye que les ressources CPU et mémoire réellement utilisées, ce qui le rend particulièrement avantageux pour les charges de travail de petite envergure.
- **Temps de mise en place :** Il est rapide et efficace de mettre en place sur ACI, en particulier pour le développement, les tests et les applications de production légères.

2.4.8 Mise en place du CI/CD :

Dans cette section, nous détaillons la mise en place du pipeline CI/CD avec **GitHub Actions** pour automatiser le processus de build, de push et de déploiement d'une image Docker sur **Azure Container Instance**.

- **Déclenchement du pipeline :** Le pipeline est déclenché à chaque **push** sur la branche `main`. Ceci permet de s'assurer que chaque mise à jour sur la branche principale initie un processus d'intégration continue.

- **Runners :** Le pipeline est exécuté sur un environnement virtuel appelé `ubuntu-latest`, fourni par GitHub Actions. Cet environnement contient déjà les outils nécessaires comme Docker et Azure CLI, facilitant ainsi les tâches CI/CD.

- **Étapes du pipeline :**

  — **Vérification du code :**
  La première étape consiste à utiliser `actions/checkout@v4`, une action préconfigurée pour cloner le dépôt GitHub dans l'environnement du runner, cela permet de s'assurer que le code source est disponible pour les étapes suivantes.
  — **Configuration de Docker Buildx :**
  Ensuite, nous configurons Docker Buildx, un outil qui permet de gérer plusieurs architectures et plateformes pour la création d'images Docker grâce à l'action `docker/setup-buildx-action@v2`.
  — **Connexion à Docker Hub :**
  À l'aide de `docker/login-action@v2`, le pipeline se connecte à Docker Hub en utilisant les informations d'authentification sécurisées fournies par les secrets GitHub (`DOCKER_USERNAME` et `DOCKER_PASSWORD`).
  — **Build et tag de l'image Docker :**
  L'image Docker est construite en utilisant la commande `docker build` et est ensuite taggée avec la dernière version pour qu'elle soit poussée sur Docker Hub.
  — **Connexion à Azure Azure command-line interface (Azure CLI) :**
  La connexion à Azure est effectuée en ligne de commande en utilisant `az login` avec les secrets pour l'authentification , cela permet d'interagir avec les services Azure à travers le pipeline.

— **Login à Azure Container Registry :**
Ensuite, le pipeline se connecte au **Azure Container Registry (ACR)** pour permettre le stockage de l'image Docker, nous utilisons la commande `az acr login` pour cette étape.

— **Poussée de l'image Docker vers ACR :**
Après avoir tagué l'image Docker avec l'URL du registre Azure, l'image est poussée sur ACR à l'aide de la commande `docker push`.

— **Déploiement sur Azure Container Instance :**
Enfin, l'image Docker est déployée sur un **Azure Container Instance (ACI)**, qui est une solution légère et économique pour déployer des conteneurs, Nous utilisons la commande `az container create` pour créer une instance avec les ressources nécessaires (CPU, mémoire, etc.) et exposer le port utilisé par l'application Streamlit 8501.

- **Pipline CI/CD Workflow :**

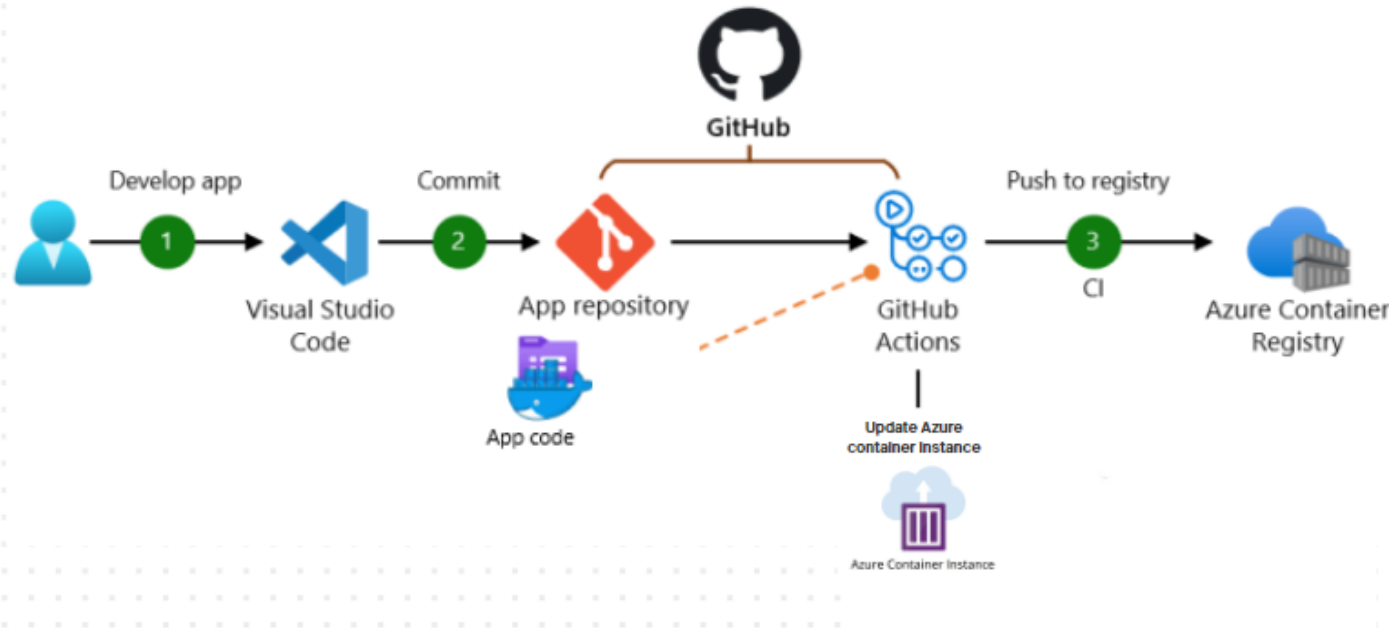


FIGURE 2.35 – Pipline Workflow

2.4.9 Les compétences acquises

- L'adaptation au télétravail est une responsabilité qui m'a permis de m'améliorer en gestion du temps et en synchronisation à distance au sein de l'équipe, malgré la commodité qui l'accompagne.
- Le travail collaboratif sur GitHub m'a permis de maîtriser la gestion des branches, les pull requests, ainsi que la résolution des conflits de fusion (merge conflicts), renforçant la communication et la coordination au sein de l'équipe.
- J'ai appris à utiliser des pipelines d'intégration continue (CI) et de déploiement continu (CD) pour automatiser les tests et garantir un déploiement sécurisé des projets.

- L'exploration des outils MLOps, notamment MLFlow, m'a permis de suivre et gérer le cycle de vie des modèles de machine learning, améliorant la traçabilité et la reproductibilité des expérimentations.
- L'utilisation des services Azure, tels qu'Azure Container Registry (ACR) et Azure Container Instances (ACI), m'a permis de stocker, gérer et déployer des conteneurs Docker dans le cloud, facilitant le déploiement d'applications sans gestion d'infrastructure complexe.
- J'ai également renforcé mes compétences dans la conteneurisation en suivant les bonnes pratiques Docker pour optimiser les images et les environnements de déploiement.

2.4.10 Difficultés rencontrées et solutions trouvées

- **Limitation en terme de sécurité :** La sécurité de la pipeline CI/CD du projet est limitée, en particulier en ce qui concerne les pratiques DevSecOps. Malgré la protection de certaines informations confidentielles par GitHub Actions grâce aux variables d'environnement, leur gestion demeure vulnérable et pourrait entraîner une exposition accidentelle des secrets, mettant ainsi en péril la sécurité du projet.
  →Solution Afin d'optimiser cette caractéristique, il est possible d'utiliser des services spécialisés tels qu'Azure Key Vault ou HashiCorp Vault, qui assurent une sécurité de stockage et un accès contrôlé aux secrets.
- **Difficulté au niveau de collaboration en ligne :** En raison de la nature du stage (à distance), la communication avec les collègues et les superviseurs a été un défi.
  →Solution : Utilisation des outils de collaboration en ligne, tels que Zoom, pour faciliter la communication et les réunions régulières.
- **Utilisation excessive des ressources informatiques pendant l'entrainement des modèles.**
  →Solution : Utilisation des services cloud et des plates-formes qui offrent des ressources GPU ou TPU.

Conclusion générale

Au terme de ce stage au sein de Applicare.ai, il est manifeste que l'intégration des techniques d'intelligence artificielle dans le domaine de la santé ouvre de nouvelles perspectives pour l'amélioration des soins aux patients. Mon expérience au sein de cette startup innovante a été extrêmement enrichissante et m'a permis de contribuer activement à un projet de grande envergure, visant à prédire le risque de complications médicales chez les patients à haut risque.

Le développement de l'application web et la mise en place des pipelines MLOps ont non seulement été des défis techniques stimulants, mais ont également offert une opportunité précieuse de travailler dans un environnement collaboratif et dynamique. La clarté des objectifs, le soutien constant de l'équipe et les interactions positives avec mes collègues ont été des facteurs clés dans la réussite de ce projet. Les relations humaines, marquées par la coopération et le soutien mutuel, ont grandement facilité la réalisation des tâches et ont contribué à créer une atmosphère de travail harmonieuse et productive.

L'acceptation de nos travaux dans le livre "AI and Data Science in Healthcare and Medicine : Applications and Future Opportunities" ainsi que la possibilité de présenter notre projet lors des conférences au CDSS Lab témoignent de la pertinence et de la qualité du travail accompli. Ces réalisations confirment non seulement le potentiel des modèles prédictifs pour transformer les pratiques médicales, mais également l'impact significatif que l'IA peut avoir sur le suivi et la gestion des patients.

En somme, ce stage a été une expérience formatrice qui a consolidé ma passion pour l'intelligence artificielle appliquée à la santé et a renforcé mes compétences techniques et professionnelles. Je suis profondément reconnaissante envers toutes les personnes qui ont contribué à ce projet, non seulement pour leur expertise et leur soutien technique, mais aussi pour leur engagement et leur esprit d'équipe. Cette expérience mémorable a été enrichie par les relations humaines positives et le soutien constant de mes collègues, préparant ainsi le terrain pour une carrière future dans le domaine de l'IA en santé, avec une vision claire de son potentiel transformateur.

Bibliographie

[1] Applicare.ai. URL : https://www.applicare.ai/.

[10] Streamlit. URL : https://docs.streamlit.io/.

[12] Présentation de mimic-iii. URL : https://www.nature.com/articles/sdata201635.

[2] Python. URL : https://www.python.org/.

[3] Mlflow. URL : https://mlflow.org/.

[4] Github. URL : https://github.com/.

[5] Dagshub. URL : https://dagshub.com/.

[6] Azure container registry. URL : https://azure.microsoft.com/fr-fr/products/container-registry.

[7] Azure container instances. URL : https://azure.microsoft.com/fr-fr/products/container-instances.

[8] Azure monitor. URL : https://learn.microsoft.com/en-us/azure/azure-monitor/.

[9] Github actions. URL : https://github.com/features/actions.