المدرسة الوطنية العليا للذكاء الاصطناعي وعلوم المعطيات - تارودانت
ÉCOLE NATIONALE SUPÉRIEURE DE L'INTELLIGENCE ARTIFICIELLE
ET SCIENCES DES DONNÉES - TAROUDANT

Université IBN ZOHR

Ecole Nationale Supérieure de l'Intelligence Artificielle et Sciences des Données de Taroudant

Filière : Sciences des données, Big Data & IA

Rapport de Mini Projet de Java

Sous le thème :

Assistance pour personnes muettes

Réalisé par :

Sohaila Ait-hamou

Mariam MOUH

ASSIA BENDAOU

Ghita Benlahcen

Encadré par :

Prof Loubna KARBIL

Année universitaire : 2024-2025

المدرسة الوطنية العليا للذكاء الاصطناعي وعلوم المعطيات - تارودانت
ÉCOLE NATIONALE SUPÉRIEURE DE L'INTELLIGENCE ARTIFICIELLE
ET SCIENCES DES DONNÉES - TAROUDANT

Université IBN ZOHR

Ecole Nationale Supérieure de l'Intelligence Artificielle et Sciences des Données de Taroudant

Filière : Sciences des données, Big Data & IA

Rapport de Mini Projet de Java

Sous le thème :

Assistance pour personnes muettes

Réalisé par :

Sohaila Ait-hamou

Mariam MOUH

ASSIA BENDAOU

Ghita Benlahcen

Encadré par :

Prof Loubna KARBIL

Année universitaire : 2024-2025

Remerciements

Nous tenons à exprimer notre profonde gratitude envers notre encadrant, Prof Loubna KARBIL, qui nous a accompagnés tout au long de ce projet avec une grande patience, un professionnalisme exemplaire et un soutien indéfectible. Ses précieux conseils, son expertise et sa bienveillance ont été d'une aide inestimable pour la réussite de notre travail. Nous avons appris énormément grâce à lui et nous sommes fiers de ce que nous avons accompli sous sa direction. Nous tenons à le remercier du fond du cœur pour son engagement et son dévouement à notre égard.

Nous souhaitons remercier Prof Loubna KARBIL pour le temps et l'énergie qu'elle a consacré à la lecture et à l'évaluation de notre travail. Vos commentaires et suggestions constructives nous ont aidés à améliorer la qualité de notre travail.

Nous sommes conscients que l'évaluation de notre travail était exigeante et nous sommes reconnaissants pour la rigueur et l'équité avec laquelle le jury a mené leur évaluation.

Nous tenons à exprimer notre profonde reconnaissance envers notre encadrant pour Votre soutien indéfectible tout au long de notre étude de tous le module.

Table des matières

Table des figures

Introduction générale

L’être humain est un ensemble de capteurs, sensibles à ce qui l’entoure, percevant le monde à travers les sens, les émotions et les interactions. Pourtant, certaines personnes, comme les muets, vivent dans un silence imposé, un monde où la parole ne leur appartient pas. Ils ressentent, comprennent, aiment, souffrent… mais ne peuvent pas exprimer tout cela avec des mots. Ce manque de voix n’est pas qu’un simple handicap physique, c’est aussi une douleur intérieure, une barrière entre leurs émotions et le reste du monde. Mais pas après l’existence de notre application, une intermédiaire entre l’utilisateurs et son entourage, tout en sensibilisant le grand public à la langue des signes.

De plus, notre plateforme ne se limite pas seulement à aider les personnes concernées. Elle est constituée également des quiz interactifs permettent aux utilisateurs (muets ou non) d’apprendre ou de réviser la langue des signes ainsi que savoir leurs niveaux actuels. C'est une opportunité de sensibiliser et éduquer les personnes à la langue des signes tous en offrant un outil pratique et éducatif, accessible à tous, pour construire un monde plus inclusif.

Le premier chapitre est une exposition approfondie de notre projet qui se concentre sur la création d'une application concernant les muets. Nous procéderons ensuite à la définition claire de nos objectifs, tout en scrutant les problématiques existantes et les critiques afin de formuler des solutions adaptées et réalisables.

Le deuxième chapitre est la partie d’analyse et conception tout en explorerons la modélisation et la conception de notre projet à l’aide des diagrammes UML. Nous commencerons par le diagramme de cas d’utilisation, puis nous examinerons le diagramme de séquence et le diagramme de classes. Cette approche nous a permis de créer une application bien organisée et adaptée aux besoins de nos utilisateurs.

Le troisième chapitre dresse une liste exhaustive des outils et technologies utilisés tout au long du développement de l'application, fournissant un aperçu précieux du cadre technologique qui sous-tend notre application.

Enfin, dans le quatrième chapitre, nous présentons l'application finale sous toutes ses facettes, démontrant comment elle répond aux besoins des utilisateurs et facilite transition des paroles de l'utilisateur vers le récepteur voulut parfaitement.

Chapitre 1 : Spécifications

1. Introduction:

Dans ce chapitre, nous entamerons par une exposition approfondie de notre projet qui se concentre sur la création d'une plateforme concernant les muets. Nous procéderons ensuite à la définition claire de nos objectifs, tout en scrutant les problématiques existantes et les critiques afin de formuler des solutions adaptées et réalisables.

2. Contexte général du projet :

1.2 Présentation du projet, définition du problème et objectifs cibles:

Ce projet a pour ambition de concevoir une application mobile intelligente capable de traduire en temps réel les gestes de la langue des signes en paroles audibles ou en texte affiché à l'écran. En utilisant la caméra du smartphone combinée à des technologies d'intelligence et reconnaissance gestuelle, l'application permettrait aux personnes muettes de communiquer plus facilement avec ceux qui ne comprennent pas la langue des signes. Le problème de départ est simple mais profond : dans la société actuelle, la majorité des individus ne maîtrisent pas la langue des signes, ce qui crée une barrière de communication importante pour les personnes muettes. Cette difficulté d'expression peut engendrer une grande frustration, un sentiment de solitude, voire une souffrance psychologique, car ces personnes ressentent les choses intensément, mais ne peuvent pas toujours les exprimer comme elles le souhaitent.

Face à cette réalité, le projet vise plusieurs objectifs. D'abord, il s'agit d'offrir un outil pratique et autonome pour permettre aux personnes muettes d'interagir plus librement avec leur entourage, que ce soit dans des situations de la vie quotidienne, professionnelle ou sociale. Ensuite, l'application intègre une dimension éducative sous forme de quiz interactifs, destinés aux utilisateurs entendant, pour leur permettre d'apprendre progressivement la langue des signes de manière ludique et accessible. À long terme, cette approche vise à favoriser l'inclusion, la sensibilisation et le rapprochement entre les communautés entendant et non entendant, dans une démarche citoyenne et humaine.

3. Cahier de charge :

3.1. Problématique :

La communication est un besoin fondamental de l'être humain, essentielle à l'expression des idées, des émotions et à la construction du lien social. Pourtant, les personnes muettes, privées de la parole, font face à une marginalisation silencieuse, marquée par des difficultés à se faire comprendre et à participer pleinement à la vie sociale, scolaire ou professionnelle. Bien que la langue des signes constitue un moyen de communication efficace, elle reste peu connue et rarement maîtrisée par la majorité de la population, ce qui aggrave l'isolement des personnes concernées. Par ailleurs, les outils technologiques existants manquent souvent d'accessibilité, de simplicité ou d'adaptabilité aux besoins spécifiques de ces utilisateurs.

La question centrale réside alors dans la capacité d'une application mobile intelligente et Inclusive à innover afin de réduire cette fracture communicationnelle, en traduisant les gestes en langage parlé ou écrit, tout en sensibilisant le grand public à la langue des signes par le biais d'un apprentissage interactif. Une telle solution permettrait non seulement d'améliorer l'autonomie des personnes muettes, mais aussi de favoriser une meilleure compréhension mutuelle et une inclusion réelle au sein de la société.

3.2. Les acteurs :

Acteurs principaux de notre plateforme : personne muets, personne intéressé et récepteurs :

- **Les personnes muettes :**

Les principaux utilisateurs de l'application en exprimer leurs besoins, pensées et émotions à travers les gestes et tous est transmis et sortie vers le récepteur.

- **Les utilisateurs entendant :**

En globe public général, famille, enseignants, amis. Interlocuteurs des personnes muettes. Public cible de la partie éducative (quiz et apprentissage de la langue des signes), tout en interagissant avec l'application pour comprendre ce que la personne muette souhaite dire et peuvent apprendre à leur rythme la langue des signes grâce au module de quiz interactifs.

3.3. Besoins fonctionnels :

- **Traduction de gestes en texte en temps réel** : Un utilisateur peut effectuer des gestes devant une caméra, et la plateforme les traduit instantanément en texte, facilitant ainsi l'interaction avec le langage des signes.
- **Apprentissage interactif par quiz** : Un utilisateur peut apprendre les signes à travers un quiz éducatif, tester ses connaissances et suivre sa progression grâce à un score final automatiquement calculé.

3.4. Les besoins non fonctionnels :

- **Performance** :

Temps de réponse rapide pour la traduction des gestes en texte en temps réel, assurant une interaction fluide avec la plateforme.
Traitement optimisé des flux vidéo pour garantir une reconnaissance gestuelle efficace et continue.
Utilisation efficace des ressources système pour offrir une expérience fluide, y compris pendant l'exécution des quiz.

- **Interface Utilisateur**

:

Conception intuitive et attrayante de l'interface utilisateur facilitant l'accès aux modules de traduction et d'apprentissage.

Cohérence visuelle entre les différentes fonctionnalités pour une expérience homogène. Hiérarchisation claire des contenus pédagogiques et des résultats pour améliorer la navigation.

Feedback visuel instantané sur les gestes reconnus et les réponses du quiz pour renforcer l'interaction.

- **Réactivité :**

Temps de chargement minimal pour les modules de traduction et d'apprentissage. Notifications en temps réel sur les résultats, conseils d'apprentissage ou erreurs de reconnaissance. Support des technologies modernes pour une navigation fluide et sans rechargement de page.

3.5. Nom et Logo de site :

Figure 1:Logo de notre application

Ce logo, inspiré du signe ASL pour « interpréter », symbolise la rencontre et la compréhension entre les personnes sourdes et entendant. Les mains qui se rejoignent illustrent l'idée de connexion, de dialogue et de respect mutuel. Elles représentent notre volonté de créer un pont entre deux mondes, en facilitant la communication et en valorisant la richesse de la langue des signes.

À travers ce symbole, nous affirmons notre engagement pour une société plus inclusive, où chacun peut s'exprimer et être compris, quelle que soit sa manière de communiquer. Ce logo incarne ainsi notre mission : briser les barrières de la communication et favoriser l'inclusion de tous.

5. Conclusion :

Dans cette section, nous avons offert une perspective globale sur le projet en exposant le cahier des charges, énumérant tous les défis et objectifs, et enfin en décrivant l'outil que nous avons adopté pour la structuration et la répartition des tâches.

Chapitre 2 : Analyse et conception

1. Introduction:

Dans ce chapitre, nous explorerons la modélisation et la conception de notre projet à l'aide des diagrammes UML. Nous commencerons par le diagramme de cas d'utilisation, puis nous examinerons le diagramme de séquence et le diagramme de classes. Cette approche nous a permis de créer une application bien organisée et adaptée aux besoins de nos utilisateurs.

2. Diagramme de cas d'utilisation:

Le rôle des diagrammes de cas d'utilisation est de collecter, d'analyser, d'organiser les exigences et de lister les principales fonctionnalités d'un système. C'est donc la première étape ML pour la conception du système. Le diagramme de cas d'utilisation comporte trois éléments principaux.

- **Acteur :** Représente un utilisateur externe ou une entité qui interagit avec le système.
- **Cas d'utilisation :** Représente une action ou une fonctionnalité spécifique que le système offre à ses utilisateurs, décrivant une interaction entre un acteur et le système.
- **Relation :** Montre les liens entre les acteurs et les cas d'utilisation, ou entre différents cas d'utilisation

Diagramme 1 : diagramme ce cas d'utilisation primaire.

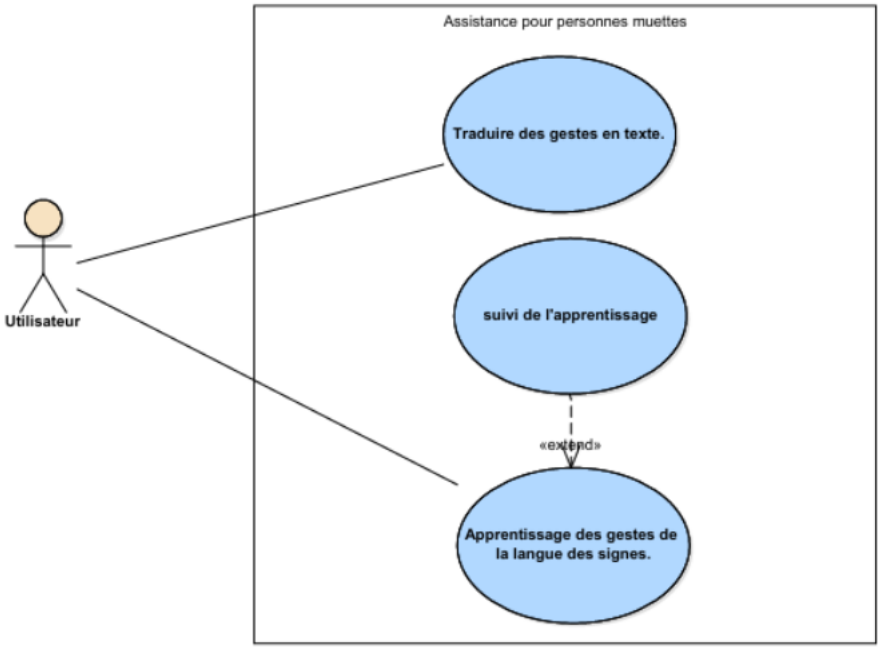

Figure 2:Diagramme de cas d'utilisation primaire.

3. Diagramme de séquence:

Les diagrammes de séquence constituent la représentation graphique des interactions entre les acteurs et le système, organisées chronologiquement dans le cadre de l'UML (Unified Modeling Language). Au sein de cette section, nous avons élaboré trois diagrammes de séquence distincts. Le premier concerne le processus de visionnage et de téléchargement de films, tandis que le second illustre la gestion des acteurs, et le dernier concerne les interactions pour la vente de livres d'occasion Diagramme.

Diagramme 2 : Diagramme de séquence du cas d'utilisation : Traduire des gestes en texte

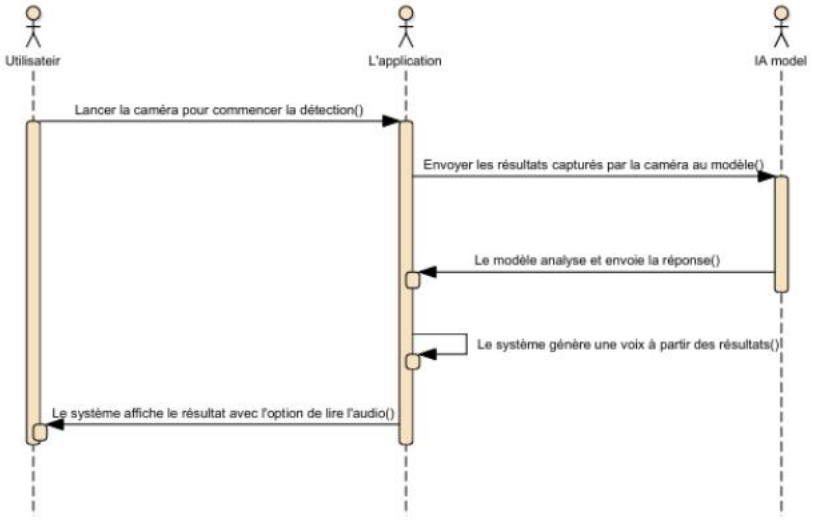

Figure 3 : Diagramme de séquence.

4. Conclusion:

Ce chapitre de conception, axé sur l'utilisation des diagrammes UML, a permis de définir clairement les interactions entre les utilisateurs et le système, ainsi que sa structure interne. L'intégration du Modèle Conceptuel de Données (MCD) et du Modèle Logique de Données (MLD) a assuré une gestion efficace des données, jetant ainsi les bases d'un système robuste et fonctionnel.

Chapitre 3 : Environnement de travail et outils

1. Introduction :

Dans le cadre de notre projet de traduction du langage des signes en texte, nous avons développé une application interactive favorisant la communication et l'inclusion. Elle intègre une traduction en temps réel ainsi qu'un module d'apprentissage par quiz. Le développement s'est appuyé sur Java, Eclipse, SceneBuilder et CSS, chacun contribuant à la conception, l'interface et la logique de l'application.

2. Outils de travail:

1. Visual studio code:

Eclipse est un environnement de développement intégré (IDE) open source principalement utilisé pour le développement en Java, mais il prend également en charge de nombreux autres langages grâce à des plugins. Disponible sur Windows, Linux et macOS, Eclipse propose des fonctionnalités avancées telles que l'auto-complétion du code, le débogage, le refactoring, la navigation dans le code, et la gestion de projets complexes. Il intègre également des outils pour le développement Web, mobile et embarqué. L'IDE est hautement personnalisable, et les développeurs peuvent enrichir ses fonctionnalités en installant une large gamme de plugins disponibles via le Eclipse Marketplace.

Figure 4:Eclipse

2. SceneBuilder:

SceneBuilder est un outil graphique développé par Gluon qui permet de concevoir des interfaces utilisateur JavaFX en glissant-déposant des composants (boutons, menus, champs de texte, etc.). Utilisé en complément d'Eclipse, il permet de créer rapidement des fichiers FXML, qui décrivent la structure de l'interface de manière déclarative. SceneBuilder facilite la séparation entre le design visuel et la logique métier du programme. Il propose également une prévisualisation en temps réel, la personnalisation des styles via CSS, et une intégration fluide avec les projets JavaFX développés dans Eclipse ou d'autres IDEs.

Figure 5 : SceneBuilder

3. GitHub

GitHub est une plateforme de développement collaboratif basée sur Git, utilisée pour héberger et gérer du code source. Elle permet aux développeurs de suivre les modifications, collaborer en temps réel, et gérer différentes versions de leurs projets. GitHub s'intègre facilement avec des IDE comme Eclipse ou VS Code, facilitant les opérations de commit, push, pull et merge. Il propose également des fonctionnalités avancées telles que les pull requests, les issues, la revue de code, la documentation en ligne (README, wikis), et l'intégration continue. Grâce à sa large communauté et à son écosystème d'outils, GitHub est devenu un incontournable du développement logiciel moderne.

Figure 6 : GitHub

4. Enterprise Architect :

Enterprise Architect est un outil complet de modélisation et de conception de systèmes, développé par Sparx Systems. Il est largement utilisé dans l'ingénierie logicielle, l'analyse métier, l'architecture d'entreprise et les projets complexes. Il prend en charge une multitude de langages de modélisation comme UML, SysML, BPMN, et Archimate, permettant de créer des diagrammes détaillés pour visualiser, concevoir et documenter des systèmes. Enterprise Architect propose également des fonctionnalités de génération de code, d'ingénierie inverse, de simulation, de traçabilité, et de gestion des exigences. Il s'intègre avec des environnements de développement comme Eclipse, facilitant la synchronisation entre modèles et code source.

Figure 7 : Enterprise Architect

5. Jupyter lab :

JupyterLab est un environnement de développement interactif puissant pour le travail avec du code, des données, et des documents scientifiques. Il permet d'éditer et d'exécuter des notebooks Jupyter, des scripts Python, des fichiers Markdown, CSV, JSON, et bien plus encore, dans une interface web flexible. JupyterLab offre une disposition en onglets et panneaux modulables, une prise en charge de la visualisation de données en temps réel, et une compatibilité avec de nombreux langages via les kernels (comme Python, R, Julia, etc.). Il est très utilisé en data science, en apprentissage automatique, en enseignement et en recherche. Grâce à son système d'extensions, il peut être personnalisé pour s'adapter à différents flux de travail.

Figure 8 : Jupyter lab.

3. Outils de développement:

1. JavaFX:

JavaFX est une bibliothèque graphique pour le développement d'interfaces utilisateur en Java. Conçu pour créer des applications riches en fonctionnalités avec une interface graphique moderne, il permet de développer des interfaces de bureau interactives et réactives. JavaFX utilise un modèle basé sur des composants tels que des boutons, des champs de texte, ou encore des tableaux, que l'on peut organiser à l'aide de conteneurs comme VBox, HBox ou GridPane. Ces éléments peuvent être décrits en XML via le langage FXML, facilitant ainsi la séparation entre la logique et la présentation. Couplé à **SceneBuilder**, un éditeur visuel FXML, JavaFX offre une méthode intuitive pour construire des interfaces sans écrire manuellement tout le code. De plus, JavaFX prend en charge la **personnalisation via CSS**, permettant aux développeurs de styliser les composants de l'application de manière précise, un peu à la façon des frameworks web comme Bootstrap.

Figure 9 :javafx

2. Java:

Java est un langage de programmation orienté objet, robuste et multiplateforme, largement utilisé pour le développement d'applications desktop, web, mobiles, et embarquées. Grâce à sa machine virtuelle Java (JVM), le code Java peut s'exécuter sur différents systèmes d'exploitation sans modification. Java est connu pour sa syntaxe claire, sa grande bibliothèque standard, et sa communauté active. Il est souvent utilisé avec des environnements comme Eclipse, IntelliJ IDEA, ou NetBeans. Dans le développement d'interfaces graphiques, Java s'associe à JavaFX (souvent

utilisé avec SceneBuilder), et il prend également en charge les frameworks populaires comme Spring, Hibernate ou JUnit pour les tests.

Figure 10 : Java

3. CSS :

CSS (Cascading Style Sheets) est utilisé avec SceneBuilder pour personnaliser l'apparence des interfaces graphiques JavaFX. Il permet de styliser les composants visuels comme les boutons, labels, champs de texte, menus, etc., en modifiant facilement leurs couleurs, polices, bordures, marges, et effets visuels. Grâce à CSS, les développeurs peuvent séparer la logique de l'application (en Java) de son design, rendant l'interface plus cohérente et facile à maintenir. SceneBuilder permet de lier directement des fichiers CSS aux fichiers FXML, et de voir un aperçu visuel du style appliqué en temps réel.

Figure 11 : Css

4. Python :

Python est un langage de programmation très utilisé dans le domaine de l'intelligence artificielle en raison de sa syntaxe simple et de sa grande flexibilité. Il a été employé pour toutes les étapes du projet : depuis le chargement et le traitement des données jusqu'à la création,

l’entraînement et l’évaluation du modèle. Son large éventail de bibliothèques spécialisées en fait un choix idéal pour les projets d’IA.

Figure 12 : python

5. TensorFlow :

TensorFlow est une bibliothèque open source développée par Google, conçue pour construire et entraîner des modèles d’apprentissage automatique, notamment les réseaux de neurones profonds. Elle permet de gérer des calculs complexes de manière optimisée grâce au support des GPU et TPU, rendant l’entraînement plus rapide et plus performant.

Figure 13 : TensorFlow

6. Pandas :

Pandas est une bibliothèque essentielle pour la manipulation de données. Elle fournit des structures puissantes comme les DataFrames, qui facilitent le nettoyage, la transformation et l’analyse des ensembles de données. Dans notre projet, elle a permis de structurer efficacement les données avant leur utilisation dans le modèle.

Figure 14 : Pandas

7. Scikit-learn :

Scikit-learn est une bibliothèque de référence pour le machine learning. Elle regroupe de nombreux algorithmes d'apprentissage supervisé et non supervisé, ainsi que des outils pour la sélection de modèles, la validation croisée et l'évaluation des performances. Elle a été utilisée notamment pour préparer les données et évaluer les résultats.

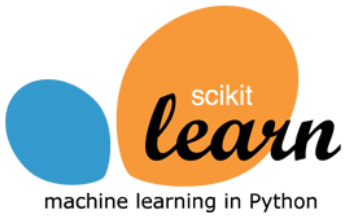

Figure 15 : Scikit-learn

8. Matplotlib :

Matplotlib est une bibliothèque de visualisation permettant de générer des graphiques variés tels que des courbes, des histogrammes ou des matrices de confusion. Elle a été utilisée pour visualiser les performances du modèle et mieux comprendre les résultats obtenus lors de l'entraînement et des tests.

Figure 16 : Matplotlib

9. Conclusion :

Au cours de ce chapitre, nous avons présenté une vue d'ensemble complète des outils utilisés pour développer notre application dédiée à la traduction du langage des signes en texte, tout en intégrant un module d'apprentissage interactif basé sur des quiz. Nous avons ensuite détaillé les environnements et technologies de développement tels que Eclipse, SceneBuilder, Java et CSS, qui ont joué un rôle fondamental dans la conception, la réalisation et l'évolution de notre solution. Chacun de ces outils a contribué à renforcer la qualité, la performance et l'accessibilité de notre application, en facilitant à la fois le développement technique et l'optimisation de l'interface utilisateur.

Chapitre 4 : Réalisation de projet

1. Introduction :

Ce modèle d'intelligence artificielle a été conçu dans le but de favoriser l'inclusion des personnes muettes en permettant une reconnaissance en temps réel de 29 signes de la main, correspondant aux lettres de l'alphabet (A à Z) ainsi que trois gestes additionnels. Les images utilisées pour l'entraînement proviennent de différentes sources en ligne, ainsi que de données collectées manuellement, ce qui a nécessité un travail de nettoyage et de prétraitement rigoureux. Ce processus est essentiel pour assurer une reconnaissance précise et rapide des gestes.

L'objectif de ce modèle est de permettre aux utilisateurs de communiquer plus facilement et de manière fluide. Grâce à son intégration dans une application desktop développée avec JavaFX, le système est conçu pour offrir une interface intuitive et accessible, permettant aux personnes sourdes et muettes de s'exprimer efficacement, en surmontant les barrières de la communication quotidienne.

2. Entraînement du modèle :

- **Préparation du dataset :**

Le dataset est structuré en répertoires, chacun représentant une lettre de l'alphabet (A à Z) ou un geste supplémentaire. Chaque répertoire contient des images correspondantes. Afin d'optimiser l'entraînement du modèle, un prétraitement est effectué pour nettoyer et organiser les données de manière efficace.

Un fichier CSV est généré pour centraliser les informations essentielles : le chemin d'accès à chaque image et son étiquette associée. Ce fichier sert de référence pour l'importation des données lors de l'entraînement du modèle. Il est également utilisé pour appliquer des transformations aléatoires sur les images, telles que la rotation, le zoom et l'ajustement de la luminosité. Ces transformations visent à accroître la diversité des images, réduisant ainsi le risque de surapprentissage et améliorant la robustesse du modèle face à des variations réelles. De plus, des techniques d'augmentation de données sont appliquées pour enrichir le dataset, en générant de nouvelles variations d'images à partir des existantes, ce qui contribue à améliorer la généralisation du modèle.

- **Construction du modèle :**

Dans cette étape, nous construisons le modèle d'intelligence artificielle en utilisant MobileNetV2, un modèle pré-entraîné largement utilisé.

- **Définition MobileNetV2 :**

MobileNetV2 est un réseau de neurones léger, rapide et efficace, spécialement conçu pour fonctionner sur des appareils aux ressources limitées, comme des ordinateurs peu puissants ou des systèmes sans GPU. Il est particulièrement adapté pour des tâches de reconnaissance d'image, notamment sur des dispositifs mobiles ou embarqués.

Nous avons choisi MobileNetV2 car :

- Il est rapide et optimisé pour les faibles ressources,
- Il offre de bonnes performances même sans GPU,
- Il est parfait pour les projets réalisés sur des machines modestes.

- **Structure du modèle :**

Nous importons MobileNetV2 avec des poids pré-entraînés pour tirer parti des connaissances acquises lors de son entraînement initial. Afin de conserver ces connaissances, les 50 premières couches du modèle sont "gelées", ce qui signifie qu'elles ne seront pas modifiées pendant l'entraînement.

Ensuite, nous ajoutons de nouvelles couches pour adapter le modèle à notre tâche spécifique :

- Couche Global Average Pooling : permet de réduire les dimensions des données.
- Couche Dense avec 128 neurones : utilisée pour la prédiction des classes.
- Couche Dropout : aide à prévenir le sur-apprentissage en désactivant aléatoirement Certaines connexions pendant l'entraînement.
- Couche finale de sortie : avec 29 neurones (correspondant aux 29 classes) et une Fonction d'activation softmax pour la classification multiclasse.

- **Construction du modèle :**

Le dataset est d'abord chargé et chaque échantillon est vérifié et affiché pour s'assurer de la qualité des données. Ensuite, les données sont divisées en 80 % pour l'entraînement et 20 % pour la validation. Des générateurs de données sont créés pour chaque phase, permettant de traiter les données efficacement sans surcharger la mémoire. Afin de compenser un éventuel déséquilibre entre les classes, les poids des classes sont calculés. Parallèlement, des callbacks sont configurés pour sauvegarder le meilleur modèle pendant l'entraînement, arrêter l'entraînement de manière anticipée si les performances stagnent et réduire progressivement le taux d'apprentissage pour affiner le modèle. Enfin, l'entraînement est lancé, ce qui permet au modèle d'apprendre à prédire les lettres à partir des images.

- **Test du modèle final :**

Après l'entraînement, nous testons les performances du modèle en chargeant le modèle final et en l'évaluant sur les données de test. Cela nous permet d'obtenir la perte (loss) et la précision (accuracy) du modèle sur les données non vues pendant l'entraînement, fournissant ainsi une évaluation de sa capacité à généraliser aux nouvelles données.

- **Visualisation des résultats :**

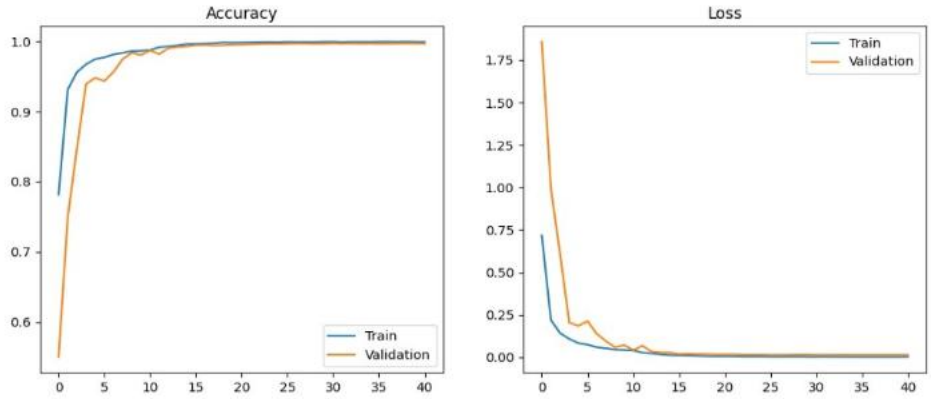

Figure 17 : Évaluation des performances : Accuracy et Loss.

3. Les interfaces de l’application :

Nous avons choisi de développer cette application en utilisant l’alphabet anglais car notre objectif était de créer un outil pouvant aider le plus grand nombre d’utilisateurs à travers le monde. L’anglais est l’une des langues les plus utilisées à l’échelle mondiale, et la langue des signes américaine (ASL) est largement reconnue. Ainsi, le projet, incluant le modèle et l’interface, a été entièrement adapté pour prendre en charge les lettres anglaises et les signes de l’ASL.

➢ La page d’accueil :

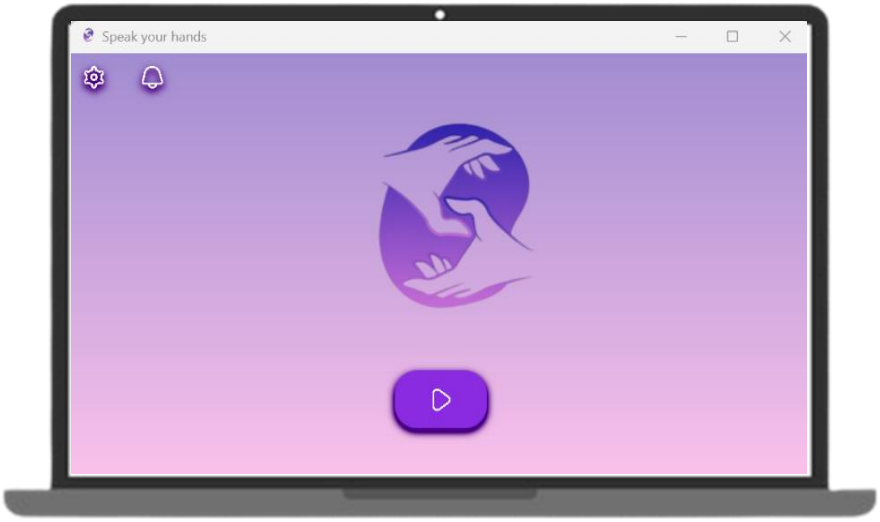

Figure 18 : La page d’accueil

Dans la page d’accueil de « Speak your hands », se distingue par une interface dont il domine le violet son dégradé et aussi celui de rose, elle affiche au centre un logo représentant deux mains en train de signer, illustrant clairement l’objectif de l’application : la communication par la langue des signes. A l’en-tête de la page d’accueil, deux icônes permettent l’accès aux paramètres et aux notifications, renforçant la simplicité d’utilisation. En bas, un bouton central en forme de lecture, invite l’utilisateur à démarrer l’aventure. Cette page d’accueil allie esthétisme et accessibilité pour offrir un accueil chaleureux et fonctionnel.

➢ La page principale :

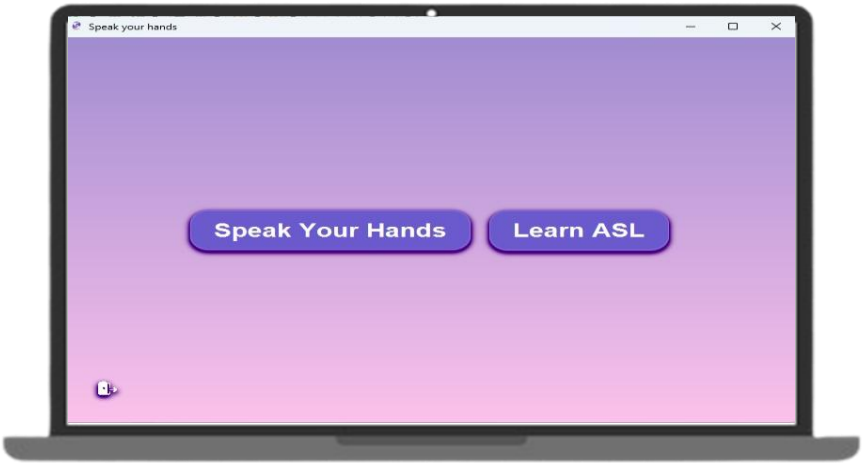

Figure 19 : La page principale.

Dans cette page dite « page principale », c'est le cœur de l'application constituer de deux boutons centraux attirent l'attention :

- **Le bouton « Speak Your Hands »** donne accès à la fonctionnalité principale de l'application : la traduction de la langue des signes en mots. En sélectionnant cette option, l'utilisateur peut utiliser sa caméra pour signer, et voir en temps réel la conversion de ses gestes en texte ou en paroles, facilitant ainsi la communication avec les personnes n'utilisant pas la langue des signes.

- **Le bouton « Learn ASL »** (ASL qui design 'American Sign Language') redirige vers une section pédagogique dédiée à l'apprentissage de la langue des signes à travers des quiz interactifs. Cette fonctionnalité permet aux utilisateurs, notamment les débutants, d'apprendre progressivement les signes de base et de tester leurs connaissances.

Ainsi la porte dans le coin inférieur gauche en bas qui facilite le retour de l'utilisateur à la page précédente.

Tout en appuyant sur le bouton « **Speak Your Hands »,** il nous lance vers la camera

✓ Fonctionnalité de traduction de signes en texte :

Figure 20 : La page de camera.

Figure 21 : fonctionnalité de traduction de signes en texte.

Cette interface correspond à la fonctionnalité de traduction de signes en texte de l'application *Speak Your Hands*. Elle illustre la manière dont l'utilisateur peut signer en face de la caméra pour que l'application reconnaisse et traduise le geste.

Au centre de l'écran, on voit une capture en direct de la main droite de l'utilisateur effectuant un signe, ici correspondant à la lettre « B » de l'alphabet en langue des signes et la lettre « A ». L'instruction en haut de l'écran invite l'utilisateur à effectuer les signes toutes les cinq secondes avec la main droite, afin de faciliter la détection.

Sous l'image, deux boutons affichent les résultats du système et flèche :

- **Prediction: B** —cela indique que l'algorithme a identifié le geste comme étant la lettre « B ».
- **Translation : B** — c'est la traduction confirmée et affichée à l'utilisateur.
- **Flèche :** c'est une flèche de retour à la page principale.

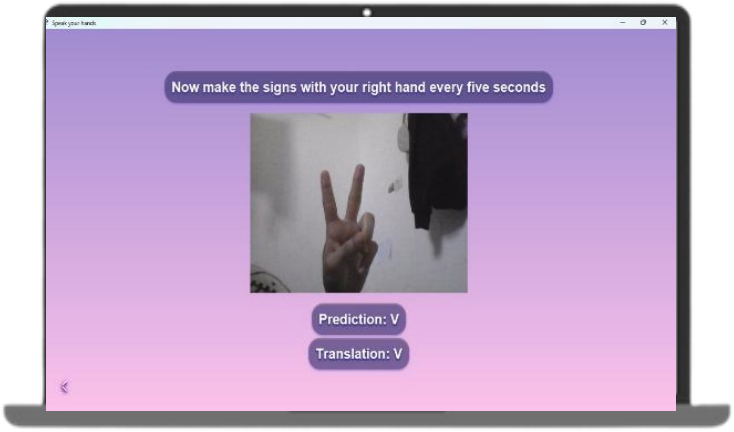

Figure 22:La page de camera (La lettre « V »)

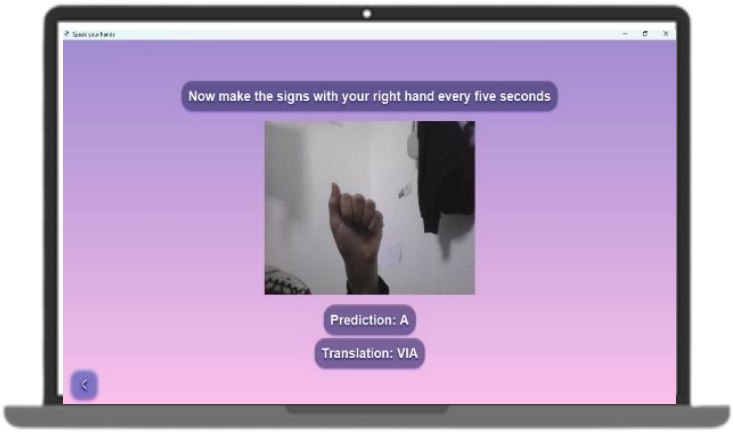

Figure 23:La page de camera (La lettre « A »).

Cette interface se compose de deux parties : une en Java et l'autre en Python. La partie Java utilise JavaFX pour afficher l'image de la caméra et le signe détecté à l'écran. Elle exécute un script Python, qui utilise le modèle d'apprentissage automatique déjà entraîné pour reconnaître les signes de la main à partir de la webcam. Le script Python renvoie trois informations : l'image de la caméra (sous forme de Base64), la lettre ou commande prédite (comme A-Z, espace ou suppression), et une traduction qui se construit au fur et à mesure. Le programme Java lit ces informations et met à jour l'interface utilisateur. Lorsque l'utilisateur arrête ou revient en arrière, le code Java arrête le script Python et nettoie tout. La partie Python capture les images, les prépare, utilise le modèle pour faire une prédiction toutes les 5 secondes, puis renvoie les résultats à Java. Elle gère également des commandes spéciales comme supprimer une lettre ou ajouter un espace.

✓ Fonctionnalité de d'apprentissage :

Tout en appuyant sur le bouton « *Learn ASL*» **,** il nous lance vers le quiz :

Figure 24 : La page d'initiation

On se plonge vers notre quiz pour tester les connaissances en langue des signes. Le texte à l'écran indique : *"If you feel like you need to challenge your knowledge about sign language, try to answer this quiz."* avec un bouton *"Start Learning"* qui nous envoi à démarrer une activité d'apprentissage interactive.

Cette interface a pour objectif de tester les connaissances des utilisateurs sur les signes correspondant aux lettres de l'alphabet

Le quiz fonctionne grâce à une classe abstraite Question, dont héritent deux types : ImageToLetterQuestion (image → lettre) et LetterToImageQuestion (lettre → image). Chaque question contient un énoncé, des choix, et une bonne réponse. Le contrôleur ASLQuizController génère aléatoirement une série de ces questions en s'appuyant sur les images disponibles, puis les affiche dynamiquement en adaptant l'interface selon leur type. À chaque réponse, il met à jour le score, et affiche un récapitulatif à la fin.

- **Question de type image → lettre:**

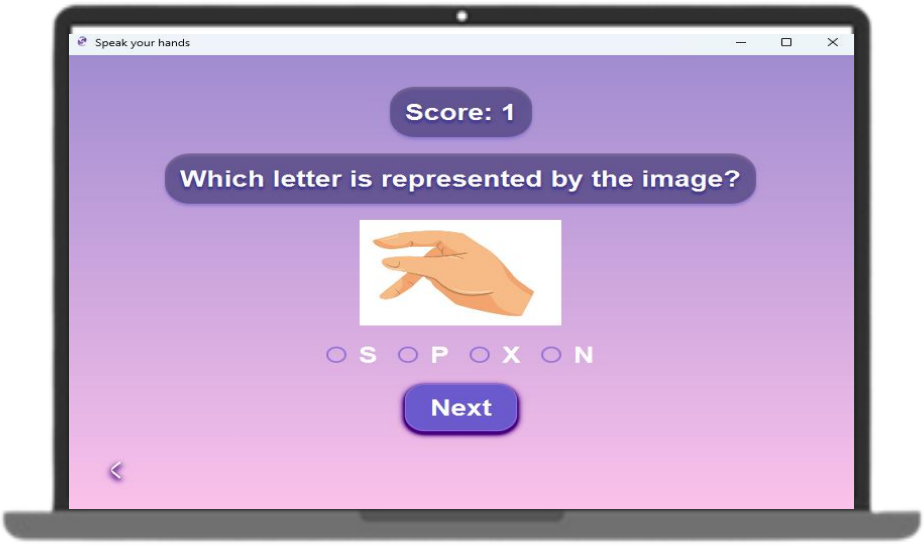

Figure 25 : Question de type image → lettre

ImageToLetterQuestion affiche une image et l'utilisateur doit deviner quelle lettre elle représente, en choisissant parmi quatre propositions de lettres.

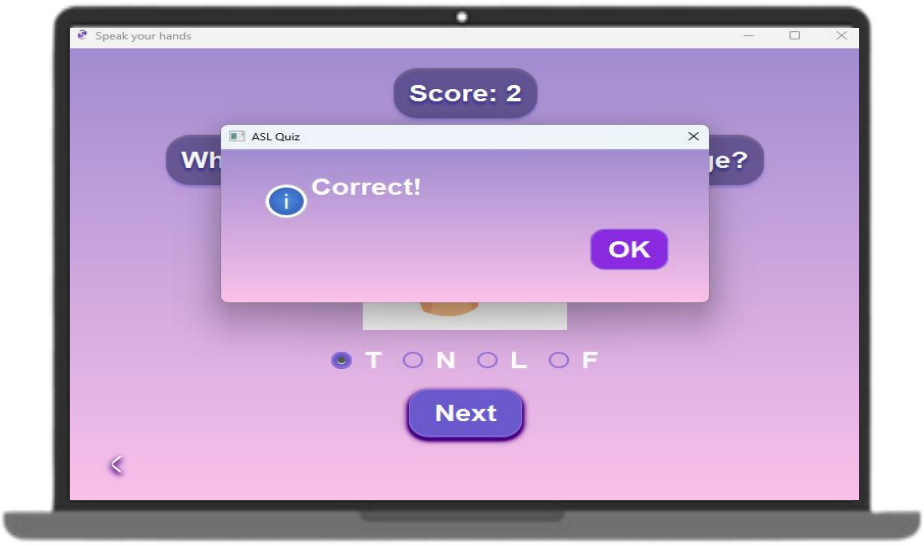

Figure 26 : L'affichage de résultat (si la lettre sélectionnée est Vrai).

Si l'utilisateur choisit la bonne réponse, une fenêtre de dialogue s'affiche pour lui indiquer que sa réponse est correcte.

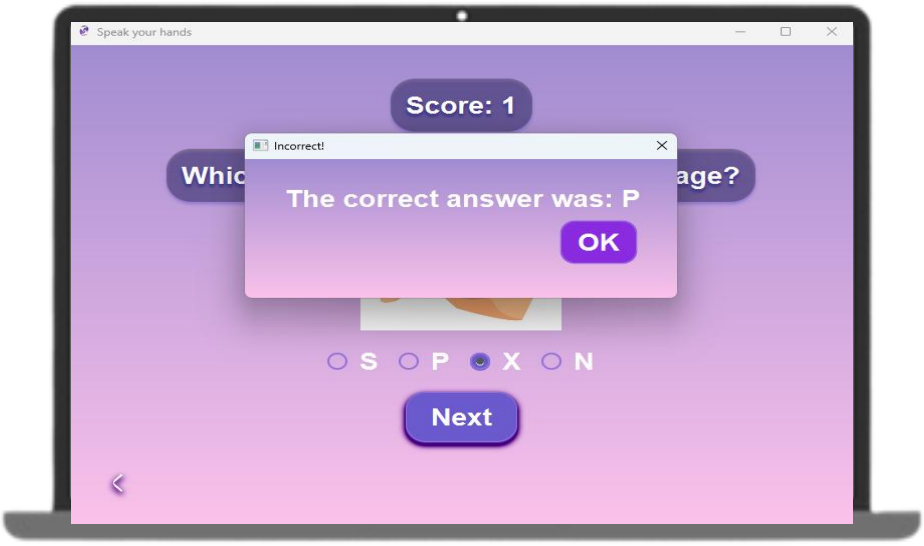

Figure 27 : L'affichage de résultat (si la lettre sélectionnée est fausse).

En revanche, si l'utilisateur sélectionne une mauvaise réponse, une boîte de dialogue apparaît également, mais cette fois pour lui signaler que sa réponse est incorrecte, tout en affichant la bonne réponse.

- **Question de type lettre → image :**

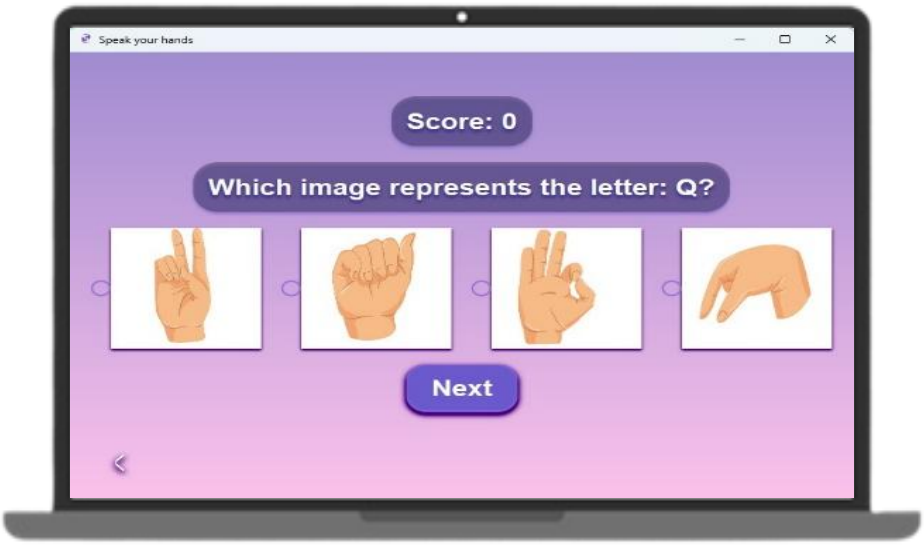

Figure 28 : question de type lettre → image

LetterToImageQuestion présente une lettre sous forme d'une question avec quatre choix d'images. L'utilisateur doit sélectionner la bonne représentation, puis cliquer sur "*Next*" pour savoir si la réponse est correcte ou non.

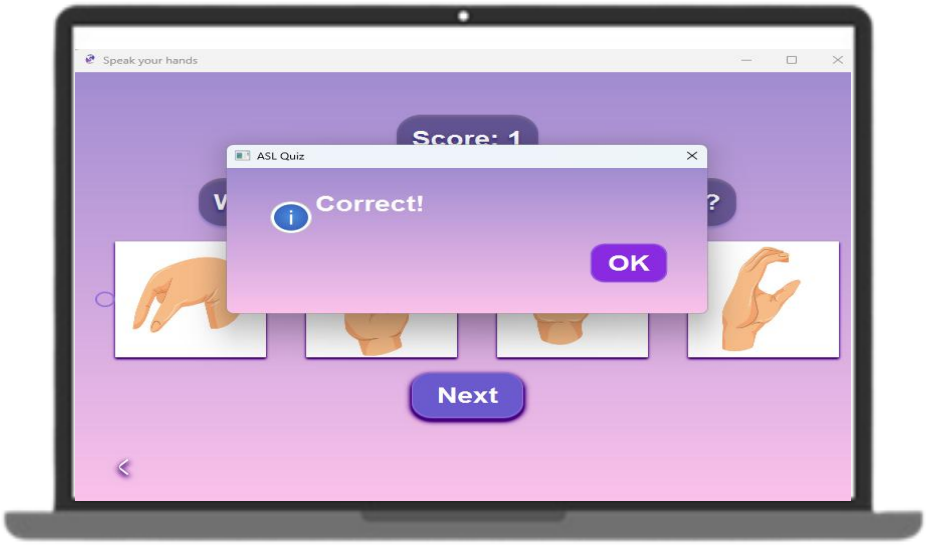

Figure 29 : l'affichage du résultat si l'image sélectionnée est correcte

On peut remarquer que le choix de la photo était correct et désigne exactement la lettre donnée.

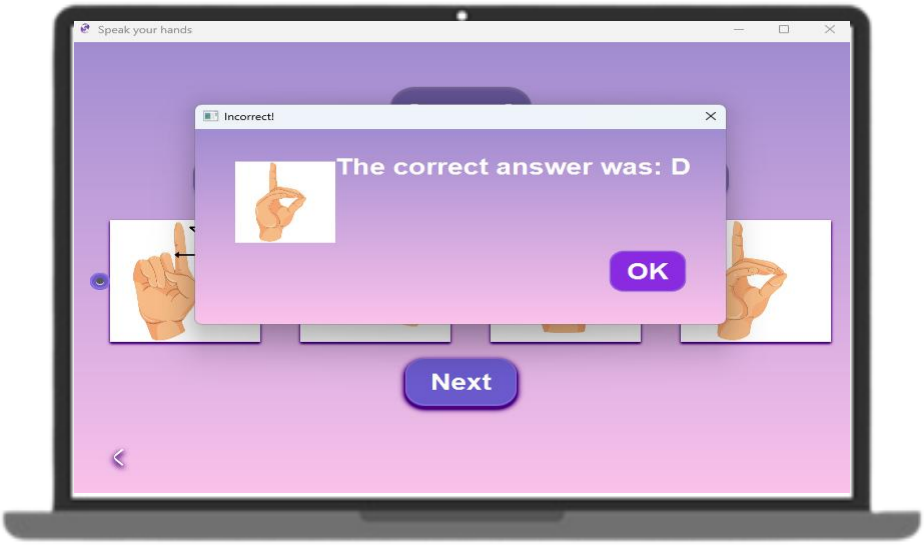

Figure 30 : L'affichage de résultat (si l'image sélectionnée est incorrect).

On peut remarquer que le choix de la photo était incorrect et nous a afficher le message « *The correct answer was* **:»** suivi de la lettre qui désigne exactement la lettre voulue.

Et à la fin du quiz on trouve le message « *Quiz Finished* **»** accompagnéé de score final.

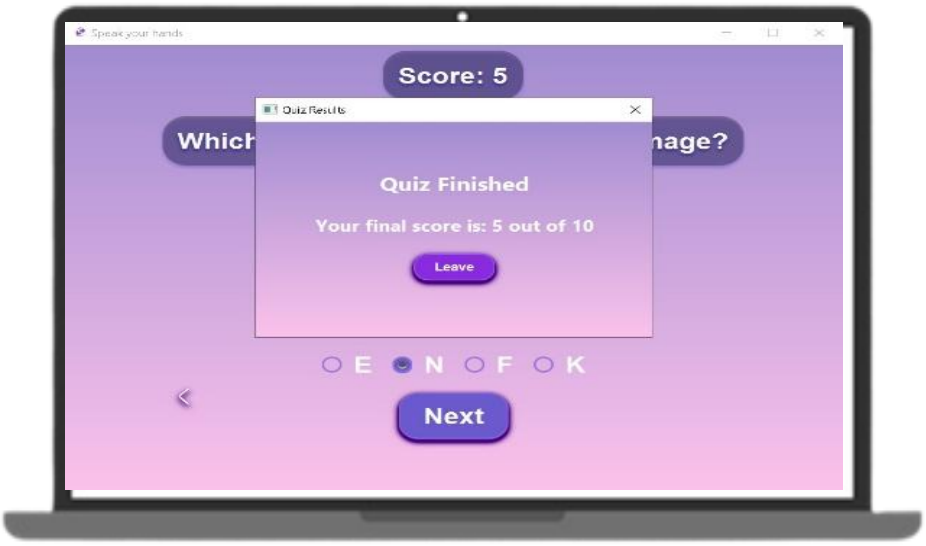

Figure 31 : L'affichage de résultat final de quiz.

4. Conclusion:

Conclusion de chapitre de réalisation cette section fournit une description détaillée de la façon dont notre plateforme fonctionne et interagit avec les utilisateurs pour satisfaire leurs besoins, offrant ainsi une perspective plus complète.

Conclusion Générale

En conclusion, la réalisation de notre projet a été une expérience enrichissante, marquée par une montée en compétences et une collaboration fructueuse entre nous trois. Nous sommes fiers d'avoir développé une application fonctionnelle et accessible qui répond aux besoins des utilisateurs, tout en facilitant la communication des personnes malentendantes avec leur entourage. En créant cette plateforme, nous avons non seulement visé à offrir une solution pratique pour la traduction des gestes en texte en temps réel, mais aussi à sensibiliser le grand public à la langue des signes.

Dans l'avenir, nous prévoyons plusieurs améliorations pour enrichir cette application, notamment l'amélioration du modèle afin de traduire non seulement les lettres, mais aussi les mots, pour une traduction plus complète et fluide. De plus, l'ajout d'une fonctionnalité de traduction des signes en voix serait un progrès significatif pour rendre l'application encore plus inclusive et accessible. Ces évolutions permettront de rendre la plateforme encore plus performante, offrant ainsi une expérience d'apprentissage et de communication plus riche et diversifiée.

Enfin, nous tenons à remercier chaleureusement notre encadrante, Loubna Karbil, pour son soutien et ses conseils tout au long de ce projet. Grâce à son accompagnement, nous avons pu mener à bien cette aventure. Nous sommes convaincus que notre application constituera un outil précieux pour les personnes malentendantes et pour tous ceux souhaitant apprendre la langue des signes, contribuant ainsi à un monde plus inclusif.

Webographie :

Au cours de la réalisation de notre application, nous sommes basées sur :

- https://www.w3schools.com/java/
- https: huggingface.co
- https://github.com/

Cours de java, python, et UML.