# Résumé

En tant que futurs ingénieurs et dans le cadre de notre programme académique, nous avons entrepris un projet innovant visant à créer une application multiplateforme pour la traduction de la Langue des Signes Marocaine (MSL) en texte et en parole, et inversement. Ce projet répond aux défis de communication auxquels est confrontée la communauté malentendante au Maroc et sert également des objectifs éducatifs en facilitant une meilleure interaction entre les utilisateurs de la langue des signes et la population générale. En utilisant Angular pour le développement frontal et des modèles d'apprentissage automatique pour une traduction précise, notre application vise à offrir un outil fiable et convivial pour combler le fossé de communication. Ce rapport détaille le processus de développement, incluant le contexte du projet, la conception fonctionnelle et technique, la mise en œuvre ainsi que l'évaluation de la solution.

# Table of contents

[Introduction 5](#_Toc185932827)

[Chapitre I : Contexte du Projet 6](#_Toc185932828)

[1. Problem Description 6](#_Toc185932829)

[2. Public Cible 6](#_Toc185932830)

[3. La Solution 7](#_Toc185932831)

[4. Méthodologie du travail 8](#_Toc185932832)

[5. Planning 9](#_Toc185932833)

[Chapitre II : Organisation et Développement du projet 10](#_Toc185932834)

[1. Organisation du Projet 10](#_Toc185932835)

[2. Diagramme de Cas d’Utilisation 11](#_Toc185932836)

[3. Technology Selection 12](#_Toc185932837)

[4. Architectural Design 14](#_Toc185932838)

[Chapitre III : Déploiement 18](#_Toc185932839)

[1. Analyse et Conception : 18](#_Toc185932840)

[1.1 Identification des Dépendances 18](#_Toc185932841)

[1.2 Évaluation des Ressources 18](#_Toc185932842)

[2.1 Solution Retenue : PaaS (Platform as a Service) 19](#_Toc185932843)

[2.2 Composants de l'Architecture 19](#_Toc185932844)

[3. Enjeux de la Migration Cloud 20](#_Toc185932845)

[3.1 Défis Techniques 20](#_Toc185932846)

[3.2 Stratégie de Migration 20](#_Toc185932847)

[3.3 Bénéfices Attendus 20](#_Toc185932848)

[Chapitre IV: Evaluation et Résultat 21](#_Toc185932849)

[1. Méthodologies de Test 21](#_Toc185932850)

[2. Résultats 22](#_Toc185932851)

[3. Interfaces Utilisateur 24](#_Toc185932852)

[4. Leçons Apprises 27](#_Toc185932853)

[5. Difficultés Rencontrées 28](#_Toc185932854)

[Conclusion 4](#_Toc185932855)

[Bibliography 5](#_Toc185932856)

# Table de Figures

[Figure 1 - GitHub Repository for Project Management 10](#_Toc185948185)

[Figure 2 - Use Case Diagram for Moroccan Sign Language Translation Application 11](#_Toc185948186)

[Figure 3 - Project Architecture for Moroccan Sign Language Translation Application 14](#_Toc185948187)

[Figure 4 : Deployment Architecture 19](#_Toc185948188)

[Figure 5 - Home Page of EchoSign 24](#_Toc185948189)

[Figure 6 - Sign to Text and Voice Translation Interface 25](#_Toc185948190)

[Figure 7 - Text and Voice to Sign Translation Interface 26](#_Toc185948191)

# Introduction

La communication efficace est un droit humain fondamental, pourtant de nombreux membres de la communauté malentendante rencontrent d'importantes difficultés à cet égard. La Langue des Signes Marocaine (MSL) constitue le principal mode de communication pour de nombreuses personnes sourdes au Maroc, mais elle reste peu comprise par la population générale. Cette barrière de communication entraîne souvent des malentendus et un isolement social pour la communauté malentendante.

En tant que futurs ingénieurs et dans le cadre de notre cursus académique, nous avons entrepris un projet innovant visant à répondre à cette problématique. Notre projet consiste en la création d’une application multiplateforme qui traduit la Langue des Signes Marocaine en texte et en parole, et inversement. Cette application s’appuie sur des technologies modernes de développement web, notamment Angular, et intègre des modèles d’apprentissage automatique pour offrir des traductions précises et en temps réel.

L’objectif principal de ce projet est de réduire le fossé de communication entre les utilisateurs de la MSL et la population générale. Ce faisant, nous cherchons à renforcer l’inclusion sociale, à améliorer l’accès à l’information et à faciliter les interactions au sein de la communauté. En outre, ce projet vise un objectif éducatif en offrant un outil d’apprentissage pour les personnes intéressées par la Langue des Signes Marocaine.

Ce rapport présente une vue d’ensemble complète de notre projet. Il débute par une exploration du contexte du projet, incluant l’identification du problème et du public cible. Ensuite, la conception fonctionnelle et technique de l’application est abordée en détail, en mettant en lumière les méthodologies et les outils utilisés. Le rapport examine ensuite la phase de mise en œuvre, en soulignant le processus de développement et les stratégies de déploiement. Enfin, l’évaluation de la solution et des réflexions sur l’impact du projet ainsi que sur ses perspectives d’avenir sont présentées.

# Chapitre I : Contexte du Projet

Ce chapitre fournit une vue détaillée du contexte dans lequel notre projet a été développé. Il commence par définir le problème spécifique que nous cherchons à résoudre, identifie le public cible et présente notre solution proposée. De plus, il décrit notre méthodologie de travail et notre calendrier de projet.

## Problem Description

Les barrières de communication affectent significativement la vie des individus malentendants utilisant la Langue des Signes Marocaine (MSL). Ces barrières entravent leur capacité à interagir efficacement avec autrui, entraînant un isolement social et un accès limité aux services essentiels et aux informations. Les méthodes traditionnelles de communication, telles que les notes écrites ou les interprètes, sont souvent inadéquates et difficilement accessibles. Il existe donc un besoin urgent d’une solution technologique capable de combler ce fossé et de faciliter une communication fluide entre les utilisateurs de la MSL et la communauté au sens large.

## Public Cible

Le public cible principal de notre projet comprend :

* **Personnes malentendantes** : Utilisateurs de la Langue des Signes Marocaine nécessitant un outil fiable pour communiquer avec des non-utilisateurs de cette langue.
* **Famille et amis** : Personnes souhaitant améliorer leur communication avec leurs proches malentendants.
* **Éducateurs et institutions** : Écoles et organisations impliquées dans l’éducation et le soutien des malentendants, qui peuvent utiliser cet outil pour enseigner et améliorer les compétences en communication.
* **Grand public** : Personnes en interaction avec des malentendants dans divers contextes et ayant besoin d’un moyen pour mieux comprendre et communiquer avec eux.

## La Solution

Notre solution est une application multiplateforme conçue pour traduire la Langue des Signes Marocaine en texte et en parole, et inversement. Cette application exploite des technologies web modernes et des modèles d’apprentissage automatique pour offrir des traductions précises et efficaces. Les composantes clés de notre solution incluent :

* **Interface Intuitive** : Développée avec Angular, l’interface utilisateur est conçue pour être intuitive et réactive, garantissant une expérience fluide pour les utilisateurs de la langue des signes et les non-initiés.
* **Intégration de l’Apprentissage Automatique** : En utilisant des modèles avancés d’apprentissage automatique, notre application peut reconnaître et interpréter avec précision les gestes de la langue des signes en temps réel, en les convertissant en texte et en parole.
* **Traduction Bidirectionnelle** : L'application prend en charge la traduction de la langue des signes vers le texte/la parole et vice versa, permettant des capacités de communication complètes.

Les principaux objectifs de notre solution sont de :

* Fournir un outil de communication accessible et efficace aux utilisateurs de la Langue des Signes Marocaine.
* Faciliter l'apprentissage de la MSL pour les personnes intéressées.
* Renforcer l'inclusion sociale en comblant le fossé de communication entre les utilisateurs de la langue des signes et la communauté élargie.

## Méthodologie du travail

Notre projet a suivi une approche structurée et collaborative, guidée par les principes du cadre SCRUM. Cette méthodologie agile nous a permis de travailler de manière itérative et d'incorporer des retours continus, assurant un progrès constant et des résultats de haute qualité. Les éléments clés de notre méthodologie de travail incluent :

* **Brainstorming et Recherche** : En équipe de trois étudiants, nous avons mené des recherches approfondies et des sessions de brainstorming pour générer des idées innovantes et des solutions. Nous avons évalué différentes approches et sélectionné les plus prometteuses pour les prototyper et les développer davantage.
* **Prototypage et Itérations** : Nous avons créé des prototypes initiaux pour visualiser nos idées et recueillir des retours. En fonction de ces retours, nous avons continuellement affiné et amélioré nos prototypes, garantissant que notre solution finale soit fonctionnelle et conviviale.
* **Division des Tâches et Collaboration** : Pour gérer la complexité du projet, nous l'avons divisé en tâches plus petites et gérables. Chaque membre de l'équipe a pris en charge des tâches spécifiques tout en collaborant étroitement pour assurer l'intégration et la cohérence. Des réunions régulières et une communication fluide ont été essentielles pour coordonner nos efforts et maintenir le progrès.
* **Développement Séquentiel** : Nous avons fixé des délais précis pour chaque tâche, progressant de manière séquentielle d'une tâche à l'autre après leur achèvement. Cette approche structurée nous a permis de respecter les échéances du projet efficacement.
* **Contrôle de Version et Documentation** : En utilisant GitHub pour le contrôle de version, nous avons maintenu un code bien documenté et assuré une collaboration efficace entre les membres de l'équipe. Cela a également facilité le suivi des modifications et l'intégration de nouvelles fonctionnalités.

## Planning

Le planning de notre projet a été soigneusement conçu pour garantir l'achèvement dans les délais de chaque tâche. Nous avons établi des échéances spécifiques pour chaque étape, progressant de manière systématique d'une tâche à l'autre. Cette approche structurée nous a permis de maintenir un rythme constant et d’assurer que tous les composants du projet soient développés et intégrés de manière cohérente. Les phases clés de notre planning incluent :

* **Recherche Initiale et Planification** : Comprendre le problème, définir les exigences du projet et générer des idées potentielles.
* **Phase de Conception** : Création de wireframes, maquettes et diagrammes d’architecture pour guider le processus de développement.
* **Phase de Développement** : Mise en œuvre des composants frontend et backend, intégration des modèles d’apprentissage automatique, et validation des fonctionnalités.
* **Tests et Validation** : Réalisation de tests utilisateurs pour recueillir des retours, affiner l'application en fonction de ces retours et garantir sa fiabilité et sa précision.
* **Déploiement** : Lancement de l’application et mise à disposition des utilisateurs, suivi d’une surveillance et d’améliorations continues.

Ce chapitre a fourni une vue d’ensemble du contexte du projet, détaillant le problème, le public cible, la solution proposée, la méthodologie de travail et le planning du projet. Avec une compréhension claire des bases du projet, le chapitre suivant explorera la conception fonctionnelle, en décrivant les fonctionnalités spécifiques de notre application.

# Chapitre II : Organisation et Développement du projet

Une organisation, une planification et une mise en œuvre efficaces ont été essentielles au succès de notre application de traduction de la Langue des Signes Marocaine (MSL). Ce chapitre décrit l'approche structurée que nous avons adoptée, y compris l'utilisation de diagrammes UML pour définir les cas d'utilisation, la conception architecturale de notre application, les technologies utilisées pour le prototypage et développement, ainsi que les détails de la mise en œuvre

## Organisation du Projet

Afin d'assurer une communication efficace et une progression fluide, nous avons utilisé plusieurs outils et méthodes :

**a. Outils de Collaboration**

**GitHub :** utilisé pour le contrôle de version et la collaboration. Cette plateforme nous a permis de gérer efficacement notre base de code, de suivre les modifications et de collaborer de manière productive. Elle a facilité :

* Le contrôle de version : Suivi de l’historique des modifications et possibilité de revenir à des versions antérieures.
* A screenshot of a computer

  Description automatically generatedLa collaboration : Travail en parallèle des membres de l'équipe avec intégration fluide grâce aux pull requests et aux revues de code.

Figure 1 - GitHub Repository for Project Management

## Diagramme de Cas d’Utilisation

Pour déterminer les fonctionnalités nécessaires à notre application, nous avons créé un diagramme de cas d’utilisation. Ce diagramme nous a permis d’identifier et de définir les interactions clés entre les utilisateurs et le système.

* **Cas d'utilisation principal**

Le cas d’utilisation principal identifié pour notre projet est de faciliter la traduction de la Langue des Signes Marocaine (MSL) en texte et en parole, et inversement. Le diagramme met en évidence les principales interactions qu’un utilisateur peut avoir avec le système.

Figure 2 - Use Case Diagram for Moroccan Sign Language Translation Application

**Fonctionnalités principales incluses dans le diagramme :**

* **Transformation du Texte et de la Voix en Langue des Signes** : Cette fonctionnalité permet aux utilisateurs de saisir du texte ou de la voix, que le système traduit ensuite en Langue des Signes Marocaine. Elle est particulièrement utile pour les non-utilisateurs de la langue des signes qui souhaitent communiquer avec les utilisateurs de la MSL.
* **Transformation de la Langue des Signes en Texte et en Parole** : Les utilisateurs peuvent utiliser cette fonctionnalité pour traduire les gestes de la MSL en texte et en parole. Cela est rendu possible grâce à l’acquisition d’images, au prétraitement, à l’extraction de caractéristiques et à la reconnaissance via des modèles d’apprentissage automatique.
* **Traduction des Résultats en Anglais et en Français** : En complément de la fonctionnalité "Transformation de la Langue des Signes en Texte et en Parole", cette option permet de traduire les gestes reconnus en texte en anglais et en français, augmentant ainsi l’utilisabilité de l’application pour un public plus large.
* **Contact avec les Administrateurs** : Les utilisateurs ont la possibilité de contacter les administrateurs pour demander de l’aide, donner leur avis ou signaler des problèmes rencontrés lors de l’utilisation de l’application.

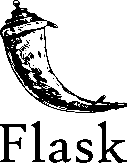
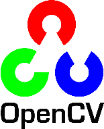
## Technology Selection

After identifying the required functionalities, we researched and selected the appropriate technologies for development. The choice of technologies was crucial in ensuring the application was robust, scalable, and efficient. Below are the details of the technologies we selected for various parts of the project:

1. **Développement Frontend**
   * **Angular :** Choisi pour sa puissance et sa capacité à créer des applications à page unique (SPA), avec une architecture modulaire et des outils robustes pour le routage, la gestion de l'état et la gestion des formulaires.
   * **Architecture basée sur les composants** : Angular permet de créer des composants réutilisables, facilitant la gestion et l'extension de l'application.
   * **Routage** : Angular Router pour une navigation fluide entre les vues.
   * **Gestion de l'état** : Services Angular et bibliothèques pour gérer l'état de l'application efficacement.

* **Tailwind CSS:** Framework CSS utilitaire permettant de créer des designs personnalisés rapidement et d'assurer une conception réactive sur tous les appareils.extend the default styles and create a unique look for our application.

1. **Développement Backend et Machine Learning**

* **Python :** Choisi pour sa simplicité et sa large bibliothèque de support, idéal pour intégrer des modèles de machine learning.
  + **Flask :** Framework léger pour créer des API RESTful, facilitant l'intégration des modèles de machine learning.
  + **Les** **points de terminaison API** : Flask permet de créer des API pour gérer les requêtes du frontend et interagir avec les modèles de machine learning pour le traitement des données.
* **OpenAI API :** Utilisée pour la synthèse vocale, permettant de convertir les résultats des modèles en discours naturel.
* **TensorFlow and Keras:** Utilisés pour développer et entraîner les modèles de reconnaissance gestuelle.
* **OpenCV:** Bibliothèque open-source de vision par ordinateur utilisée pour le traitement d'images et de vidéos, cruciale pour capturer et interpréter les gestes en langage des signes.
  + **Traitement d'images :** OpenCV offre un ensemble d'outils pour prétraiter les images capturées depuis la webcam
  + **Extraction de caractéristiques :** utilisé pour extraire les caractéristiques pertinentes des images prétraitées, qui sont ensuite alimentées dans les modèles de machine learning pour la classification.

## Architectural Design

L'architecture de notre application est un schéma complet qui décrit l'interaction entre les différents composants et services. Ce design garantit un système évolutif, maintenable et efficace. L'architecture est divisée en trois couches principales : le frontend, le backend et le modèle de machine learning.

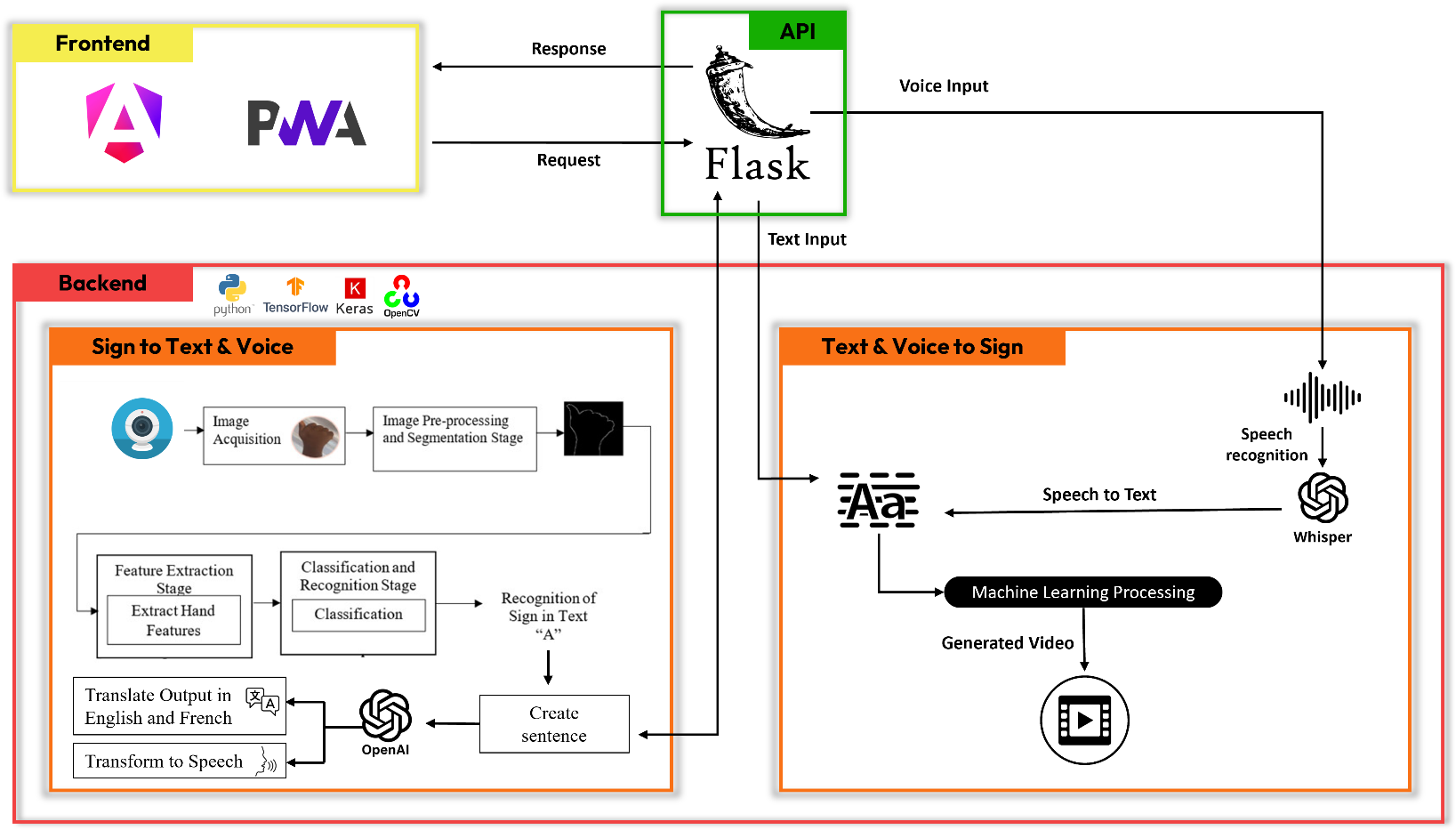


Figure 3 - Project Architecture for Moroccan Sign Language Translation Application

L'architecture du projet est conçue pour intégrer de manière transparente diverses technologies et composants afin de fournir une application de traduction efficace. Elle est structurée en les couches suivantes :

* **Frontend**

Le frontend est responsable de l'interface utilisateur et de l'expérience utilisateur.

* **Angular :** Utilisé pour développer une application monopage (SPA) avec une interface dynamique et réactive. L'architecture basée sur des composants d'Angular permet de créer un code modulaire et maintenable.
* **Progressive Web App (PWA) :** Le frontend est également conçu comme une PWA pour garantir son accessibilité sur divers appareils et plateformes, offrant une expérience native sur le web.
* **Backend**

Le backend est le cœur de l'application, traitant le traitement des données, l'intégration des modèles de machine learning et la communication avec le frontend.

* **Python et Flask :** Le backend est construit avec Python et le framework Flask, permettant de créer des points de terminaison API RESTful de manière efficace.
* **API Layer :** Cette couche facilite la communication entre le frontend et le backend, assurant un flux de données fluide et une interaction sans faille entre les différents composants.
* **Machine Learning Models**

Les modèles de machine learning sont essentiels pour traiter les gestes en langue des signes et les traduire en texte et en parole.

* **TensorFlow et Keras :** Ces bibliothèques sont utilisées pour développer et entraîner les modèles de machine learning pour la reconnaissance des gestes et les tâches de traduction.
* **OpenCV :** Utilisé pour le traitement d'images et de vidéos, essentiel pour capturer et interpréter correctement les gestes en langue des signes.
* **Data Flow**

Le flux de données dans l'application comprend plusieurs étapes pour assurer que les entrées de l'utilisateur sont traitées avec précision et efficacité pour produire les sorties souhaitées :

1. **Interaction utilisateur :** ils interagissent avec l'application via le frontend Angular, en entrant du texte, de la voix ou des gestes en langue des signes.
2. **API Requests :** Le frontend envoie des requêtes au backend via l'API pour traitement. Ces requêtes incluent les entrées de l'utilisateur à traduire.
3. **Traitement Machine Learning :** Le backend utilise les modèles de machine learning pour traiter ces requêtes. Les gestes en langue des signes sont reconnus et traduits en texte, tandis que les entrées textuelles et vocales génèrent des gestes en langue des signes correspondants.
4. **Gestion des réponses :** Les données traitées sont renvoyées au frontend, où elles sont affichées à l'utilisateur sous le format souhaité (texte, voix ou gestes en langue des signes).

* **Détails des Composants**

1. **Sign to Text & Voice :** 
   * **Acquisition d'images :** Capturer les gestes en langue des signes avec une webcam.
   * **Prétraitement et segmentation des images :** Préparer les images capturées pour l'extraction des caractéristiques.
   * **Extraction des caractéristiques :** Extraire les caractéristiques pertinentes des images prétraitées pour la classification.
   * **Classification et reconnaissance :** Utiliser des modèles de machine learning pour classifier et reconnaître les gestes en langue des signes.
   * **Traduction et transformation :** Traduire les signes reconnus en texte et convertir le texte en parole avec l'API OpenAI.
2. **Text & Voice to Sign :** 
   * **Traitement du langage naturel :** Traiter le texte ou l'entrée vocale pour comprendre le contexte et la signification.
   * **Génération de signes :** Générer des gestes en langue des signes correspondants à l'entrée textuelle ou vocale traitée.

* **Intégration d’API**

La couche API facilite la communication entre le frontend et le backend, garantissant un flux de données sans faille entre les composants. Elle gère toutes les requêtes et réponses API, assurant que le frontend reçoit les données traitées en temps voulu.

# Chapitre III : Déploiement

### Analyse et Conception :

Dans cette première partie, nous allons examiner en détail les différents aspects techniques et organisationnels nécessaires à la mise en place de l'application **EchoSign** dans un environnement cloud.

### 1.1 Identification des Dépendances

Pour assurer le bon fonctionnement de l'application **EchoSign** dans un environnement cloud, nous avons identifié plusieurs composants essentiels. L'environnement d'exécution repose sur *Node.js* pour la partie frontend, *Python 3.8+* pour les services backend et les modèles de machine learning, ainsi que des conteneurs Docker pour l'encapsulation des services. En termes de bibliothèques et frameworks, l'application s'appuie sur Angular pour le frontend, Flask pour l'API REST, et utilise TensorFlow et OpenCV pour les modèles de machine learning, complétés par diverses bibliothèques Python dédiées au traitement d'images.

### 1.2 Évaluation des Ressources

**1.2.1 Compute**

Les besoins en puissance de calcul sont clairement définis. Le frontend nécessite au minimum 2 vCPUs et 4GB de RAM, tandis que le backend, en raison du traitement ML, requiert 4 vCPUs et 8GB de RAM. Une capacité de scaling automatique est prévue pour adapter les ressources à la charge.

**1.2.2 Stockage et Réseau**

Les besoins en stockage comprennent 20GB pour le code source et les assets, complétés par un stockage élastique pour les données utilisateur. La haute disponibilité est assurée par des mécanismes de backup et de réplication. Le réseau doit fournir une bande passante dédiée pour le streaming vidéo, une faible latence pour le traitement en temps réel, et intégrer SSL/TLS pour la sécurité des communications.

1. **Architecture Azure Cloud**

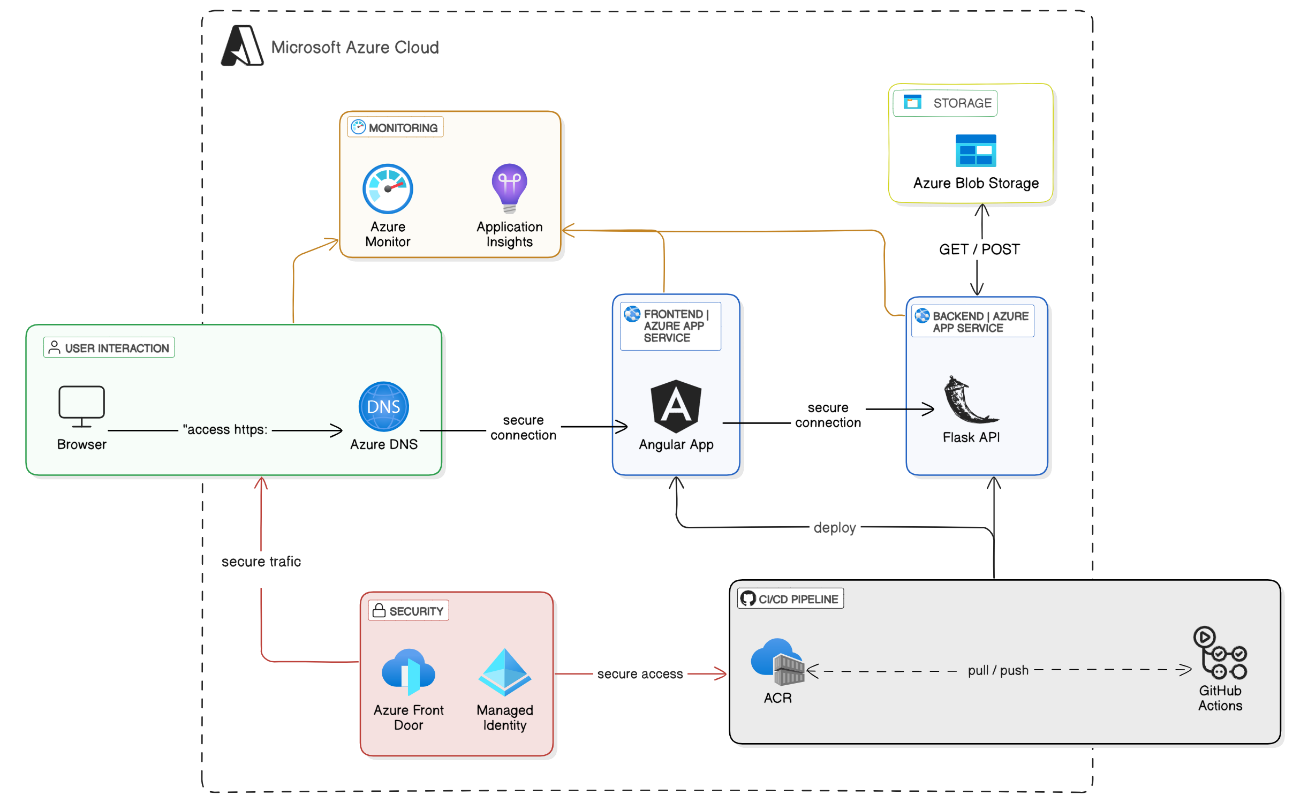
****

Figure 4 : Deployment Architecture

### 2.1 Solution Retenue : PaaS (Platform as a Service)

Nous avons opté pour une solution PaaS sur Microsoft Azure. Ce choix s'explique par plusieurs avantages majeurs :

* + Une gestion simplifiée de l'infrastructure.
  + Un scaling automatique des ressources.
  + Une intégration native des services Azure.
  + Des coûts optimisés par rapport à une solution IaaS.
  + La possibilité de se concentrer sur le développement applicatif.

### 2.2 Composants de l'Architecture

L'architecture se compose de plusieurs éléments clés :

* Le frontend est hébergé sur Azure App Service, permettant un scaling automatique et une intégration SSL/TLS native.
* Le backend, également sur Azure App Service, utilise des conteneurs pour l'API Flask et assure une connexion sécurisée avec le frontend tout en intégrant les modèles ML.
* La sécurité est renforcée par Azure Front Door pour la protection DDoS, l'utilisation de Managed Identity pour l'authentification, et le chiffrement SSL/TLS des données.

### 3. Enjeux de la Migration Cloud

### 3.1 Défis Techniques

La migration vers le cloud présente plusieurs défis techniques, notamment en termes de performance (optimisation du traitement ML en temps réel, gestion de la latence réseau), de sécurité (protection des données sensibles, conformité RGPD), et de disponibilité (configuration haute disponibilité, plan de reprise d'activité).

### 3.2 Stratégie de Migration

La stratégie de migration s'articule autour de trois phases :

* **Phase de préparation :** audit, formation, création des environnements
* **Phase de migration :** déploiement progressif avec tests
* **Phase de stabilisation :** monitoring, optimisation, documentation

### 3.3 Bénéfices Attendus

Les bénéfices attendus sont multiples. Sur le plan technique, nous anticipons un scaling automatique efficace, une haute disponibilité et des performances optimisées. Au niveau organisationnel, la solution permettra de réduire les tâches d'administration, d'accélérer les déploiements et d'améliorer l'agilité. Enfin, les avantages financiers incluent des coûts proportionnels à l'usage, une optimisation des ressources et un ROI amélioré.

# Chapitre IV : Evaluation et Résultat

Dans ce chapitre, nous évaluons la performance et l'impact de notre application multiplateforme, qui traduit la Langue des Signes Marocaine (LSM) en texte et en parole, et vice versa. Nous présentons les méthodologies de test, les résultats, les retours des utilisateurs et l'impact global du projet. Cette évaluation nous aide à comprendre les forces et les faiblesses de notre solution et offre des perspectives pour de futures améliorations.

### Méthodologies de Test

Pour garantir l'exactitude et la fiabilité de notre application, nous avons utilisé diverses méthodologies de test, notamment les tests unitaires, les tests d'intégration et les tests d'acceptation utilisateur.

* 1. **Tests Unitaires**

Les tests unitaires ont été effectués pour valider le bon fonctionnement des composants et modules individuels de l'application. Les domaines clés testés comprenaient :

* **Composants Frontend** : Vérification du bon comportement de chaque composant Angular et de leur interaction correcte avec le backend. Des tests ont été effectués pour vérifier le rendu des éléments UI, la gestion des entrées utilisateur, et la correction des sorties affichées aux utilisateurs.
* **Points de terminaison Backend** : Vérification de la gestion correcte des requêtes et des réponses par les points de terminaison API. Les tests se sont concentrés sur la précision du traitement des données, les temps de réponse, et la gestion des erreurs.
* **Modèles de Machine Learning** : Tests de la performance des modèles dans la reconnaissance et la traduction des gestes en langue des signes. La précision, le rappel et la précision des modèles ont été mesurés pour garantir une reconnaissance et une traduction fiables des gestes.
  1. **Tests d'Intégration**

Les tests d'intégration ont été effectués pour garantir que les différentes parties de l'application fonctionnent ensemble de manière transparente. Cela a inclus des tests sur l'interaction entre le frontend, le backend et les modèles de machine learning.

* **Intégration API** : Vérification que les données circulent correctement entre le frontend et le backend via les appels API. Divers scénarios ont été testés pour garantir que les points de terminaison API reçoivent et traitent correctement les requêtes du frontend et renvoient les réponses appropriées.
* **Intégration des Modèles** : Garantie que les modèles de machine learning reçoivent et traitent correctement les données, fournissant des traductions correctes. Cela a impliqué des tests du processus de bout en bout, depuis la capture des gestes en langue des signes jusqu'à l'affichage des traductions en texte et en parole.

### Résultats

Les résultats de notre processus de test et d'évaluation sont résumés ci-dessous, mettant en évidence les principaux résultats et les métriques de performance de l'application.

* 1. **Précision**

La précision des modèles de machine learning dans la traduction des gestes en langue des signes était un indicateur clé. Les modèles ont atteint des taux de précision élevés, démontrant leur efficacité dans la traduction en temps réel.

* **Reconnaissance des Gestes** : Les modèles ont reconnu avec précision une large gamme de gestes de la langue des signes, avec un taux de précision moyen de 91 %.
* **Synthèse Vocale** : Le composant de synthèse vocale a fourni des traductions claires et naturelles, améliorant l'expérience utilisateur.

1. **Utilisabilité**

Les retours des utilisateurs ont indiqué que l'application était conviviale et intuitive, les utilisateurs appréciant l'interface épurée et la navigation facile.

* **Interface Utilisateur** : L'utilisation d'Angular et de Tailwind CSS a permis de créer une interface réactive et visuellement attrayante. Les utilisateurs ont trouvé l'interface de l'application intuitive et facile à naviguer.
* **Expérience Utilisateur** : Les utilisateurs ont trouvé l'application facile à utiliser, avec des instructions simples et une courbe d'apprentissage minimale. Les fonctionnalités de traduction ont été particulièrement appréciées pour leur précision et leur rapidité.

1. **Performance**

L'application a bien performé en termes de temps de réponse et de fiabilité, garantissant une expérience utilisateur fluide.

* **Temps de Réponse** : Les appels API et les temps de traitement des modèles ont été optimisés pour fournir des traductions quasi instantanées. Les utilisateurs ont signalé peu de latence et des temps de réponse rapides lors de l'utilisation de l'application.
* **Fiabilité** : L'application a montré une grande disponibilité et stabilité pendant les tests, sans pannes ou erreurs significatives. Le système a géré efficacement les utilisateurs simultanés et les multiples demandes.

1. **Impact**

L'application a le potentiel d'avoir un impact positif significatif sur la vie des utilisateurs de la langue des signes et leurs communautés.

* **Inclusion Sociale** : En facilitant une meilleure communication, l'application favorise l'inclusion sociale et réduit l'isolement des personnes malentendantes. Les utilisateurs ont apprécié la possibilité de communiquer plus efficacement avec les personnes ne connaissant pas la langue des signes.
* **Outil Éducatif** : L'application sert de ressource éducative pour ceux qui apprennent la Langue des Signes Marocaine, améliorant l'acquisition et la maîtrise de la langue. Les enseignants et les étudiants ont trouvé l'application utile pour la pratique et l'apprentissage.

### Interfaces Utilisateur

The following images showcase the user interface of the EchoSign application, illustrating the main functionalities and user experience.

* **Home Page**

La page d’accueil est l’écran principal de l’application **EchoSign**. Elle accueille les utilisateurs et leur offre des options pour commencer à traduire entre la langue des signes, le texte et la parole.



Figure 5 - Home Page of EchoSign

* **Message de Bienvenue :** Un message qui présente l’objectif de l’application : faciliter la communication avec la communauté malentendante en transformant la Langue des Signes Marocaine en texte et audio.
* **Menu de Navigation :** Une barre de navigation en haut de la page permet d'accéder facilement aux différentes sections de l’application, notamment Accueil, À propos, Contact et Découvrir.
* **Interface de Traduction de Language des Signes vers Texte et Speech**

Cette interface permet aux utilisateurs de traduire les gestes en texte et en parole grâce à des fonctionnalités avancées.

* **Flux Vidéo de la Webcam** : Un flux en direct capture les gestes de la langue des signes via la webcam. Les utilisateurs peuvent démarrer ou arrêter la capture à l’aide des boutons.
* **Section des Résultats** : Affiche les gestes reconnus sous forme de texte. Les utilisateurs peuvent écouter le texte traduit en cliquant sur le bouton "Écouter".
* **Options de Traduction** : Les utilisateurs peuvent choisir de traduire les gestes reconnus en **Darija marocaine**, en français ou en anglais.
* A black and white sound waves

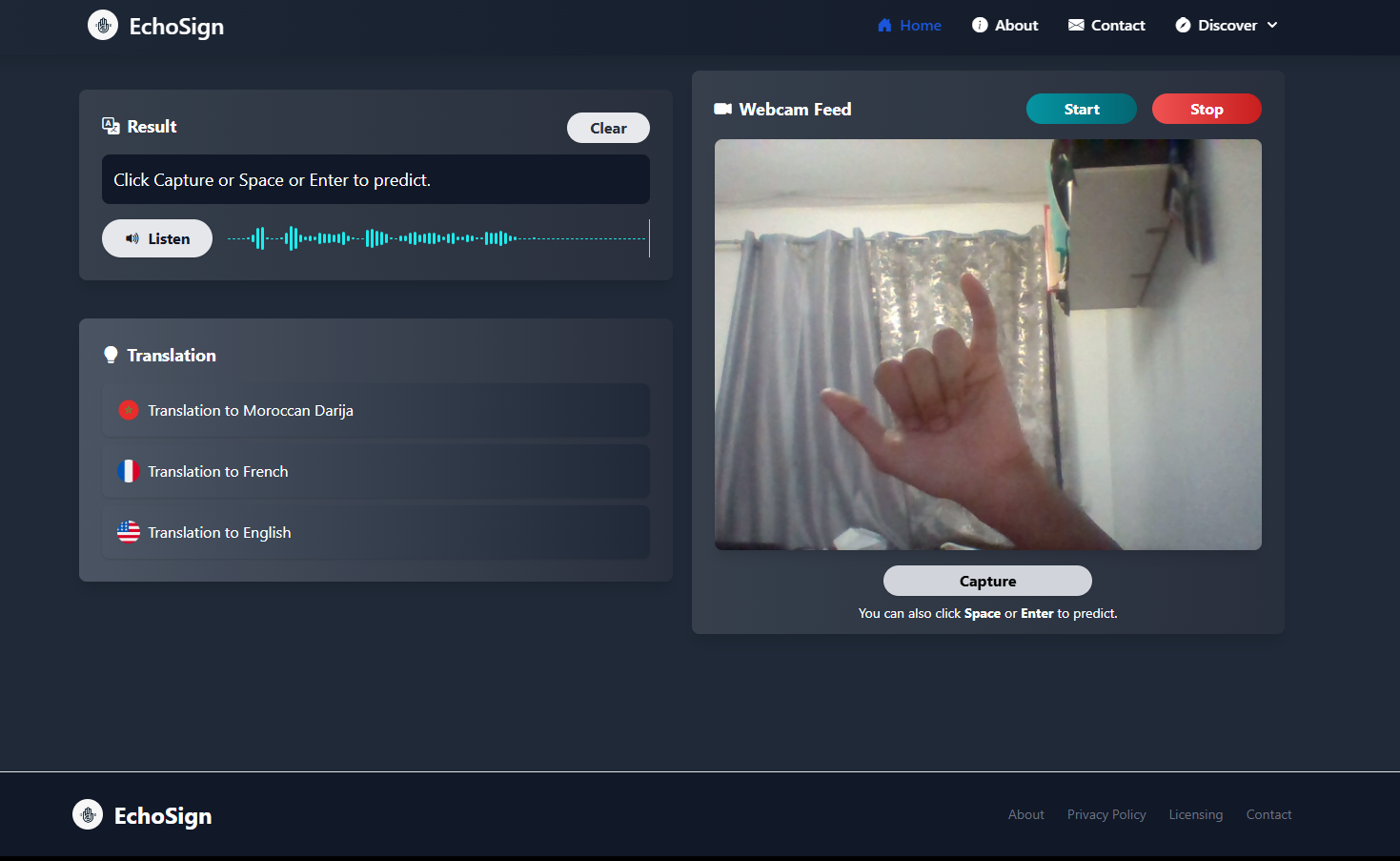
  Description automatically generated**Bouton de Capture** : Permet de capturer le geste actuel pour le traiter et l’interpréter.

Figure 6 - Sign to Text and Voice Translation Interface

* **Interface de Traduction du Texte et de la Voix vers Langue des Signes**

Cette interface permet aux utilisateurs de saisir du texte ou de la voix pour les traduire en gestes de la langue des signes.

* **Saisie de Texte** : Une zone de texte où les utilisateurs peuvent saisir les phrases qu’ils souhaitent traduire en langue des signes.
* **Saisie Vocale** : Une option pour enregistrer une entrée vocale qui sera convertie en texte, puis traduite en gestes.
* **Section des Résultats** : Affiche une vidéo représentant les gestes traduits en fonction des entrées texte ou vocales.
* A screenshot of a computer screen

  Description automatically generated**"Exprimez-vous"** : Une interface de type chat où les utilisateurs peuvent voir le texte qu’ils saisissent et soumettre leur demande de traduction. La vidéo correspondante des gestes est affichée à gauche.

Figure 7 - Text and Voice to Sign Translation Interface

### Leçons Apprises

Tout au long du projet, nous avons rencontré divers défis et opportunités de croissance. Les principales leçons apprises incluent :

* **Collaboration Efficace** : Travailler en équipe a nécessité une communication régulière, des tâches clairement définies et l’utilisation d’outils collaboratifs comme WhatsApp et GitHub pour assurer une intégration fluide.
  + **Outils de Communication** : WhatsApp pour des mises à jour rapides, Google Meet pour des réunions virtuelles et des sessions de brainstorming.
  + **Gestion des Tâches** : Division claire des responsabilités et utilisation de **GitHub Issues** pour suivre les progrès et résoudre les blocages.
* **Intégration des Technologies** : Intégrer Angular pour le frontend, Python pour le backend et des modèles d’apprentissage automatique a enrichi nos compétences et notre compréhension des outils de développement.
  + **Stack Technologique** : Maîtrise d'Angular, Flask, TensorFlow et autres technologies nécessaires pour créer une application robuste.
  + **Intégration du Code** : Test et débogage approfondis pour garantir une interaction fluide entre les composants.
* **Design Centré sur l’Utilisateur** : L’interaction avec les utilisateurs et l’intégration de leurs retours ont permis d’améliorer l’utilisabilité et l’efficacité de l’application.
  + **Feedback Utilisateur** : Tests d’acceptation réguliers pour mieux comprendre les besoins des utilisateurs et ajuster l’application.
  + **Prototypage** : Utilisation d’outils comme Figma pour visualiser et ajuster les fonctionnalités avant leur développement complet.

### Difficultés Rencontrées

Au cours du projet, nous avons relevé plusieurs défis qui nous ont offert des opportunités d’apprentissage :

* **Collecte des Données**
* **Diversité des Jeux de Données** : Assurer une diversité des gestes pour entraîner des modèles précis a nécessité une collaboration avec des organisations locales et l’utilisation de données publiques.
* **Prétraitement des Données** : Nettoyage et préparation des données pour entraîner les modèles, une étape longue mais essentielle.
* **Entraînement des Modèles**
* **Ressources Informatiques** : Entraîner des modèles avec une précision élevée a nécessité des ressources computationnelles significatives.
* **Ajustement des Hyperparamètres** : Trouver les paramètres optimaux a exigé de nombreuses expérimentations.
* **Problèmes d’Intégration**
* **Développement des API** : Garantir que les API traitent correctement les requêtes entre le frontend et le backend a nécessité des tests rigoureux.
* **Intégration des Modèles** : Incorporer les modèles d’apprentissage automatique dans l’environnement live tout en maintenant des performances optimales a représenté un défi.

Ces expériences ont renforcé nos compétences techniques et organisationnelles, nous préparant mieux à relever des défis similaires à l’avenir.

# Conclusion

Le développement de l’application de traduction de la Langue des Signes Marocaine (MSL) a été une expérience complète et enrichissante. Ce projet visait à réduire le fossé de communication entre la communauté malentendante et la population générale en proposant un outil permettant de traduire la MSL en texte et en parole, et inversement. Le parcours, de la conceptualisation au déploiement, a impliqué des recherches approfondies, une planification rigoureuse et une collaboration efficace, aboutissant à une solution fonctionnelle et impactante.

Tout au long du projet, nous avons atteint plusieurs jalons importants. Nous avons travaillé en équipe de manière cohésive, en utilisant divers outils de communication et de collaboration. Nous avons intégré des technologies web modernes et des modèles d’apprentissage automatique pour créer une application robuste. Notre approche centrée sur l’utilisateur nous a permis de développer une interface intuitive et conviviale grâce aux retours des utilisateurs et aux tests effectués. Le projet a obtenu une grande précision dans la traduction des gestes de la MSL en texte et en parole, créant un outil capable d’améliorer considérablement la communication et l’inclusion sociale pour la communauté malentendante.

Nous avons rencontré plusieurs défis, tels que la collecte de données, l’entraînement des modèles et les problèmes d’intégration. En misant sur la collaboration et l’apprentissage continu, nous avons réussi à surmonter ces obstacles et à livrer une application performante. Plusieurs axes d’amélioration restent possibles, notamment l’élargissement du vocabulaire gestuel, le raffinement continu des modèles d’apprentissage automatique pour améliorer la précision, l’ajout d’options de personnalisation pour les utilisateurs, l’intégration de fonctionnalités de communication en temps réel, et l’assurance que l’application soit accessible à tous, y compris aux personnes ayant des handicaps supplémentaires.

En conclusion, l’application de traduction de la MSL a démontré son potentiel à avoir un impact positif significatif sur la vie des utilisateurs de la langue des signes. Les compétences et les connaissances acquises grâce à cette expérience contribueront sans aucun doute à nos futures réalisations en tant qu’ingénieurs. Nous sommes fiers de ce que nous avons accompli et restons engagés à poursuivre le développement et l’amélioration de ce projet.

# Bibliography

|  |  |
| --- | --- |
| [1] | Kayo Yin et Jesse Read, «Better Sign Language Translation with STMC-Transformer,» chez *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, 2020. |
| [2] | Y. X. D. L. Sijie Yan, «Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action,» 2018. |
| [3] | T. Liu, W. Zhou et H. Li, «Sign language recognition with long short-term memory,» chez *2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, 2016. |
| [4] | J. W. Thad Starner, «Real-Time American Sign Language,» chez *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, Cambridge, 1998. |