



Université Mohammed V
Faculté des Sciences
Rabat



Master Science et Ingénierie de Données

Rapport de Stage PFA

Déploiement d'un modèle Deep Learning qui automatise la segmentation des dents 3D

Réaliser à : 3D SMART FACTORY

Période du : 4 Juillet au 8 Septembre 2023

Réalisé par :

EL BELGHITI Hamza, AKHMIM Abdelilah
ESSABIR Mohamed, EL QESSOUAR Tariq

Encadré par :

Mr. BERTIN GARDELLE Thierry
Mr. MOUNCIF Hamza
Mr. KASSIMI Amine

Année Universitaire : 2022-2023

Table des matières

I	Introduction Générale	1
1	Introduction	2
2	Présentation de l'organisme d'accueil	4
2.1	Introduction	4
2.2	Services	4
2.3	Fiche d'identité	5
3	Problématique et contexte du projet	6
3.1	Contexte du projet	6
3.2	Problématique	6
3.3	Objectif du projet	7
3.4	Conduite du projet	7
3.5	Diagramme de GANTT	10
II	Segmentation 3D	11
4	La notion du 3D	12
4.1	Introduction	12
4.2	Définition	12
4.3	Contenu des fichiers 3D	13
4.4	Représentation mathématique des formes 3D	13
4.4.1	Les données euclidiennes	13
4.4.2	Les données non euclidiennes	14
4.5	Les formats de fichier 3D	16
4.5.1	Fichiers VTK	16
4.5.2	Fichiers OBJ	16
5	La notion de segmentation	18
5.1	Définition	18

5.2	Types de segmentations	18
6	MeshSegNet	19
6.1	L'architecture de MeshSegNet	19
6.1.1	Les couches de MeshSegNet	20
6.1.2	Upsampling	26
6.1.3	Downsampling	27
III	Analyse et Conception	29
7	Analyse	30
7.1	Introduction	30
7.2	Etude des besoins	30
7.2.1	Besoins fonctionnels	30
7.2.2	Besoins non fonctionnels	31
7.3	Diagrammes UML	31
7.3.1	Diagramme de cas d'utilisation	32
7.3.2	Diagramme de séquence	32
7.4	Pipelines CI/CD	34
7.5	L'API REST	35
8	Conception	37
8.1	Logiciels et outils	37
8.2	Front End	41
8.3	Back End	44
IV	Réalisation	48
9	Back End	49
9.1	Architecture du Back End	49
9.2	Visualisation du fichier VTP générer avec Paraview	52
9.2.1	Sans post-processing	52
9.2.2	Avec post-processing	52
9.3	Alternative à Paraview	53

9.4	Deploiement	55
10	Front End	61
10.1	Design Figma	61
10.2	Structure du Front End	62
10.3	Architecture du Front End	63
10.4	Présentation des interfaces de l'application	63
10.4.1	Page d'accueil	64
10.4.2	Page Documentation	66
10.4.3	Page à propos	67
10.4.4	Page d'authentification	68
10.4.5	Page Segmentation	69
V	Conclusion	73
Bibliographie		75

Remerciement

Tout d'abord, je remercie Allah le tout puissant de m'avoir donné le courage et la patience nécessaires à mener ce travail à son terme. Pour la réalisation de ce projet, je tiens à exprimer mes plus vifs remerciements à :

Mr. Thierry Bertin Gardelle

L'encadrant du stage qui nous a soutenus tout au long du projet, nous tenons à exprimer notre gratitude pour sa disponibilité et les précieux conseils qu'il nous a prodigués lors de la réalisation de ce projet.

Mr. Amine Kassimi

Pour sa disponibilité constante tout au long du projet. À chaque fois que nous avions besoin de le solliciter, il était prêt à nous apporter son assistance précieuse à travers ses remarques éclairées et son soutien infaillible. Sa présence et son engagement ont grandement contribué à la réussite de notre projet.

Mr. Mouncif Hamza

Pour sa précieuse contribution tout au long de notre parcours, son expertise, ses conseils judicieux, sa présence et son dévouement ont joué un rôle essentiel dans la réalisation de notre projet et son encouragement permanent nous a donné la confiance nécessaire pour surmonter les obstacles rencontrés.

Enfin, un grand merci à mes coéquipiers qui ont travaillé main dans la main tout au long de ce projet. Votre collaboration, votre persévérance et votre esprit d'équipe ont été inestimables. Ensemble, nous avons surmonté les défis et accompli quelque chose dont nous pouvons tous être fiers. Merci d'avoir partagé cette aventure avec moi et d'avoir contribué à notre réussite collective.

Résumé

Dans le cadre de notre Projet de Fin d'Année, nous avons travaillé au sein de la société 3D Smart Factory pour créer une application web qui déploie un modèle de Deep Learning capable de segmenter automatiquement des objets 3D. Ce projet s'inscrit dans un contexte d'évolution rapide de l'impression 3D et de l'Intelligence artificielle, en particulier l'apprentissage profond, qui offre des possibilités d'amélioration du contenu 3D et d'application à la vision par ordinateur. Nous avons utilisé une technique basée sur des maillages nommée MeshSegNet, qui permet de segmenter les dents en 3D avec plus de précision et de rapidité, ce qui est très bénéfique pour l'industrie dentaire. Nous avons déployé le modèle en développant une API REST avec FastAPI sur le Cloud d'Amazon Web Services (AWS).

Mots clés : MeshSegNet, Segmentation Dentaire, 3D Deep Learning, Scan Intra-oral, Cloud Computing, Visualisation 3D

Abstract

As part of our Final Year Project, we worked with 3D Smart Factory to create a web application that deploys a Deep Learning model capable of automatically segmenting 3D objects. This project is set against a backdrop of rapid developments in 3D printing and Artificial Intelligence, in particular Deep Learning, which offers opportunities for improving 3D content and applying it to computer vision. We used a mesh-based technique called MeshSegNet, which enables teeth to be segmented in 3D with greater precision and speed, which is of great benefit to the dental industry. We deployed the model by developing a REST API with FastAPI on the Amazon Web Services (AWS) Cloud.

Key words : MeshSegNet, Dental Segmentation, 3D Deep Learning, Intra-oral Scanning, Cloud Computing, 3D Visualization

Liste des abréviations

Abréviation	Désignation
ML	Machine Learning
3D	Three-dimensional
GANTT	Generalized Activity Normalization Time Table
UML	Unified Modeling Language
CI	Continuous Integration
CD	Continuous Delivery
API	Application Programming Interface
REST	Representational State Transfer
JSON	JavaScript Object Notation
XML	Extensible Markup Language
AWS	Amazon Web Services
HTML	Hypertext Markup Language
VTK	Visualization Toolkit
VTP	Visualization Toolkit Polygonal Data
GLM	Graph Constrained Learning Module
FTM	Feature-Transformer Module
BN	Batch Normalisation
ReLU	Rectified Linear Unit
MLP	Multi-Layer Perceptron
GMP	Global Max Pooling
KNN	k-Nearest Neighbors

FIGURE 1 – Liste des abréviations

Première partie

Introduction Générale

CHAPITRE 1

Introduction

Avant l'avènement de l'impression 3D, les interactions homme-objets et homme-images étaient sévèrement restreintes. La technologie en deux dimensions, avec ses images sur surfaces planes, souffrait du manque de profondeur naturelle, entravant la perception des textures et distances par l'œil. Pour une meilleure visualisation, une projection en trois dimensions s'imposait.

Parmi les domaines bénéficiant de l'évolution de l'impression 3D, le secteur médical en ressort comme le plus transformé. Les techniques d'impression ont permis de produire des instruments chirurgicaux, de l'électronique médicale et de réparer des parties corporelles endommagées, telles que les os et les dents. La conception assistée par ordinateur (CAO) a connu une révolution, avec une meilleure résolution et une taille réduite. Cela facilite la pose directe de dispositifs dans la bouche des patients, améliorant le travail des professionnels de la santé et accélérant les résultats. La dentisterie numérique en découle, visant à améliorer la vitesse, la précision et l'efficacité économique des traitements dentaires.

Cette innovation repose sur des scanners 3D capturant des images intra-orales du patient. En couplant ces scanners à l'impression 3D, des modèles tridimensionnels des mâchoires supérieure, inférieure, voire des deux, sont obtenus. Ces modèles contiennent des détails précis tels que distance, dimensions, surface, couleurs et textures.

Avec l'automatisation cruciale dans de multiples domaines, de nouvelles techniques doivent être développées pour traiter ces données nouvelles. Chaque projet requiert des données spécifiques selon les objectifs et l'étude. Des techniques variées sont introduites pour résoudre les problèmes des données 3D, étant donné les variations dans la représentation des objets tridimensionnels.

À l'ère de l'intelligence artificielle, la convergence avec l'impression 3D était inévitable. Parmi les modèles issus de l'apprentissage en profondeur, MeshSegNet se démarque. Créé pour automatiser la segmentation et l'annotation dentaire, il remplace des méthodes manuelles fastidieuses en obtenant des résultats supérieurs. Basée sur des maillages, cette technique s'avère efficace pour résoudre les problèmes liés aux petites zones dentaires aux formes et apparences variées.

Dans ce rapport, nous nous intéressons particulièrement au déploiement du modèle entraîner sur la technique MeshSegNet, en utilisant différentes techniques et technologies web pour la segmentation des fichiers 3D, les représenter de manière réaliste et les afficher en temps réel.

La structure de ce rapport s'articule autour de 4 chapitres :

- Dans le premier chapitre, nous présentons le cadre général du projet.
- Le deuxième chapitre se focalise sur la segmentation et la notion du 3D.
- Dans le troisième chapitre (Analyse et conception) on aborde en détail l'analyse et la conception du projet, en mettant l'accent sur l'utilisation des différents diagrammes UML, la méthode REST et les différentes technologies utilisées.
- Dans le dernier chapitre (Réalisation) nous présentons le déploiement du modèle sur le cloud ainsi que la mise en œuvre de notre site web.

Enfin, nous terminerons avec une conclusion générale qui récapitule les résultats et les accomplissements du projet dans leur intégralité. Nous mettrons en avant les compétences et les connaissances que nous avons acquises grâce à ce projet, ainsi que les difficultés que nous avons rencontrées et les résolutions que nous avons trouvées. Pour finir, nous soulignerons l'importance de cette expérience pour notre croissance personnelle et professionnelle.

CHAPITRE 2

Présentation de l'organisme d'accueil

2.1 Introduction

3D SMART FACTORY est un établissement privé basé à Mohammedia qui s'est fixé pour mission de service public, d'intérêt collectif et sociétal. Fondée en 2018, elle est gérée par Mr Bertin Thierry Maurice Leon.



FIGURE 2.1 – Logo de 3D Smart Factory

La 3D SMART FACTORY offre un environnement propice aux start-ups, dans le but d'aider les jeunes entrepreneurs à développer et concrétiser leurs idées. Elle propose plusieurs structures de soutien et de formation dès les premières phases de réflexion ou de création.

2.2 Services

La société 3D SMART FACTORY offre plusieurs services à ces clients comme :

- **Encadrement :** Une équipe experte qui offre du soutien et de l'encadrement dans tous les aspects liés à la réalisation des projets, de l'idée au succès (Gestion de projet, Coaching et Plan de travail).
- **Financement :** 3D Smart Factory aide les porteurs de projets à financer leurs idées, avec une analyse approfondie de leurs besoins (Paramètres financiers, Paramètres techniques et Paramètres réglementaires).

- **Encouragement :** La motivation pour nous est un concept incontournable, afin de régler le niveau d'engagement des entrepreneurs (Attractivité du travail, Perspectives d'évolution et Développement personnel).

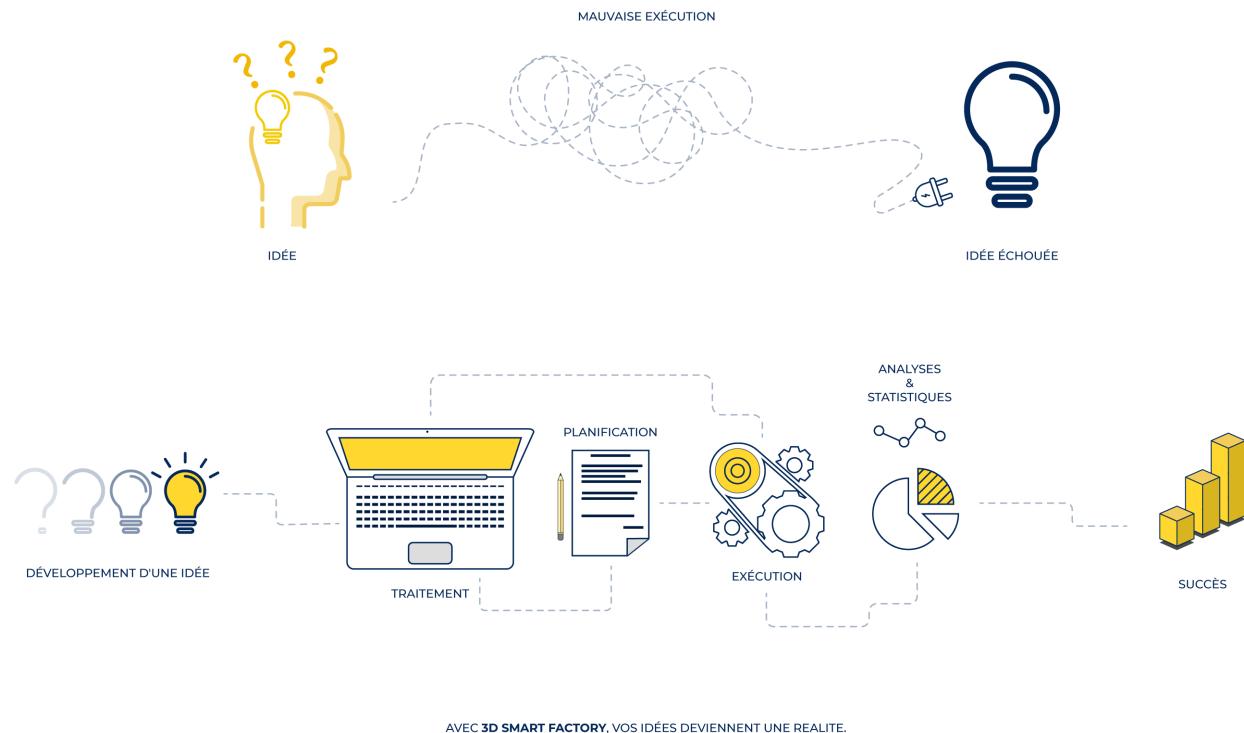


FIGURE 2.2 – Services de 3D Smart Factory

2.3 Fiche d'identité

Le tableau suivant représente la fiche d'identité de l'entreprise 3D Smart Factory :

Raison sociale	3D Smart Factory
Forme juridique	Société à Responsabilité Limitée
Date de création	2018
Capital	100000 DHS
Activité	Recherche et innovation
Site web	3D Smart Factory (csit.ma)
Siège sociale	75 Lottissement La Gare Etg 01 Mohammédia

FIGURE 2.3 – La fiche d'identité de la 3D Smart Factory

CHAPITRE 3

Problématique et contexte du projet

3.1 Contexte du projet

3D SMART FACTORY nous a donné la mission de déployer un modèle Deep Learning (MeshSegNet) pré-entraîné et de créer une application web pour le rendre accessible aux utilisateurs du domaine de la dentisterie médicale. Les progrès réalisés dans le domaine de la Conception Assistée par Ordinateur (CAO) ont ouvert la voie à la mise à disposition d'outils plus accessibles aux professionnels de la santé, en particulier dans le domaine dentaire, pour faciliter l'identification des problèmes rencontrés par les patients.

3.2 Problématique

L'étiquetage des dents pour des raisons liées à leur apparence inhabituelle chez certains patients, ainsi que les formes non uniformes des parties environnantes comme les tissus gingivaux, et les difficultés à capturer les zones profondes de la bouche, telles que les molaires postérieures, à l'aide de scanners intra-oraux, présentent des défis considérables. De plus, la répétition des tâches manuelles de segmentation et d'annotation s'avère non seulement épuisante, mais également dépendante de l'expertise humaine pour l'utilisation des logiciels existants. Ainsi, dans le but de simplifier le processus pour le domaine de la dentisterie médicale qui emploie ces données à des fins louables, il est impératif de développer des méthodes d'étiquetage automatique des dents, améliorant ainsi l'efficacité et la précision de la planification des traitements orthodontiques.

En première étape, la recherche d'une solution substantielle consiste à utiliser un modèle de Deep Learning nommé MeshSegNet pré-entraîné fourni par l'entreprise et le déployer pour pouvoir l'utiliser après avoir développer une interface utilisateur pour interagir les fichiers 3D.

3.3 Objectif du projet

Dans ce projet, notre but est de déployer le modèle MeshSegNet afin d'accomplir la segmentation automatisée des dents en 3D. L'objectif est de mettre en place un système de segmentation de bout en bout qui soit conscient des erreurs, capable d'autocorrection et totalement automatisé. Les modèles 3D renferment des centaines de milliers de faces maillées, et notre tâche consiste à associer chaque face maillée à une dent ou à des tissus gingivaux spécifiques et les étiqueter en conséquence. Pour cela, nous adopterons l'approche MeshSegNet. La précision de la segmentation réalisée par notre système doit surpasser toutes les méthodes existantes.

Par la suite, nous mettrons en place une interface pour interagir avec le modèle, tout en configurant l'infrastructure nécessaire pour son exécution. Enfin, nous déployerons le modèle pour le rendre accessible et opérationnel en production, offrant ainsi la capacité de segmenter les dents représentées sous forme de maillages.

3.4 Conduite du projet

En tant qu'équipe de 5 personnes, nous avons opter pour utiliser la méthodologie SCRUM. Scrum est de nos jours l'approche agile la plus plébiscitée par les équipes de développeurs car elle promeut les valeurs du fameux Agile Manifesto : la collaboration avec le client, l'acceptation du changement, l'interaction avec les personnes et des logiciels opérationnels.

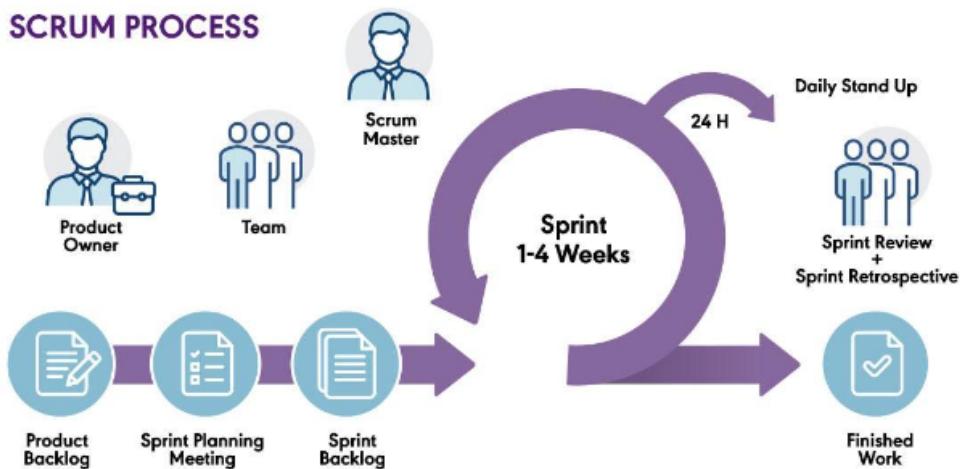


FIGURE 3.1 – Architecture du SCRUM

Scrum surpassé les autres méthodologies de développement agile à bien des égards. C'est actuellement le Framework le plus utilisé et le plus fiable dans l'industrie du logiciel. Certains des avantages bien connus de Scrum sont les suivants :

→ **Facilement évolutif** : Les processus Scrum sont itératifs et gérés dans des périodes de travail spécifique, ce qui permet à l'équipe de se concentrer sur des fonctionnalités définies pour chaque période. Cela se traduit non seulement par de meilleurs livrables qui correspondent aux besoins de l'utilisateur, mais permet également aux équipes de faire évoluer les modules en termes de fonctionnalité, de conception, de portée et de caractéristiques de manière ordonnée, transparente et simple.

→ **Conformité aux attentes** : Le client établit ses attentes en indiquant la valeur qu'apporte chaque exigence du projet, l'équipe les estime et le Product Owner établit sa priorité en fonction de ces informations. Le Product Owner vérifie régulièrement que les exigences ont été respectées lors des démonstrations de sprint et fournit des commentaires à l'équipe.

→ **Flexible aux changements** : La méthodologie est destinée à s'adapter aux changements d'exigences induits par les besoins des clients ou l'évolution du marché qui accompagnent les projets complexes.

→ **Réduction des risques** : La capacité à éliminer efficacement les risques à l'avance en réalisant d'abord les fonctionnalités les plus importantes et en connaissant la vitesse à laquelle l'équipe avance dans le projet.

→ **Meilleure qualité** : La méthode de travail, ainsi que l'exigence d'obtenir une version fonctionnelle après chaque itération, contribuent à la production de logiciels de meilleure qualité.

L'équipe SCRUM est composée de trois entités : le Product Owner, l'équipe de développement et le Scrum Master. Ces équipes doivent être auto organisées et transversales. Ils peuvent choisir la meilleure approche pour effectuer le travail lorsqu'ils s'auto-organisent et ne comptent pas sur des personnes extérieures pour les diriger.

Les sprints sont des cycles de travail qui sont utilisés pour créer des incrément de produits logiciels ou de services livrables. Les sprints sont également appelés itérations. Les sprints sont toujours courts, généralement entre 1 et 4 semaines.

Le Product Backlog est l'élément central pour gérer toutes les exigences connues d'un projet Scrum. Il comprend toutes les exigences du client et les résultats de travail requis pour exécuter et mener à bien un projet. Les exigences fonctionnelles et non fonctionnelles, ainsi que d'autres caractéristiques liées à l'expérience utilisateur et à la conception de l'interface utilisateur, sont considérées comme des exigences. Les réunions de sprint (sprint meetings).

→ **Sprint Planning :** Le but du Sprint Planning est de définir ce qui sera fait dans le Sprint et comment cela sera fait. Cette réunion se tient au début de chaque Sprint et définit comment le projet sera abordé en fonction des étapes et des délais du Product Backlog.

→ **Daily Scrum :** Chaque début de journée, notre équipe de développement se réunit pendant 5-15 minutes pour examiner l'état d'avancement vers l'objectif du Sprint. On décrit le travail effectué pendant la journée précédente, les problèmes et les blocages rencontrés, on demande de l'aide en cas de besoin.

→ **Sprint Review :** Cette réunion a lieu généralement le dernier jour du Sprint et permet à l'équipe Scrum de montrer l'incrément terminé aux parties prenantes. En plus d'inspecter les fonctionnalités de travail produites pendant le Sprint, Le Product Owner nous donne des commentaires et des remarques utiles qui peuvent être incorporés dans le Product Backlog et utilisés pour guider le travail pour les futurs sprints.

Pendant notre période de stage chez 3D SMART FACTORY, les sprints avaient une durée d'une semaine. Le product owner était Mr. BERTIN Thierry, le Scrum Master était Mr. MOUNCIF Hamza et pour l'équipe de développement on était quatres personnes : EL BELGHITI Hamza (FSR), AKHMIM Abdelilah (ENSAJ), ESSABIR Mohamed (ENSAJ) et EL QESSOUAR Tariq (ENSETM).

3.5 Diagramme de GANTT

Le diagramme de Gantt est un outil utilisé en ordonnancement et en gestion de projet et permettant de visualiser dans le temps les diverses tâches composant un projet. Il s'agit d'une représentation d'un graphe connexe, valué et orienté, qui permet de représenter graphiquement l'avancement du projet.

Voici notre diagramme de Gantt permettant de visualiser les différentes tâches qu'on a réalisé pendant la durée du stage.

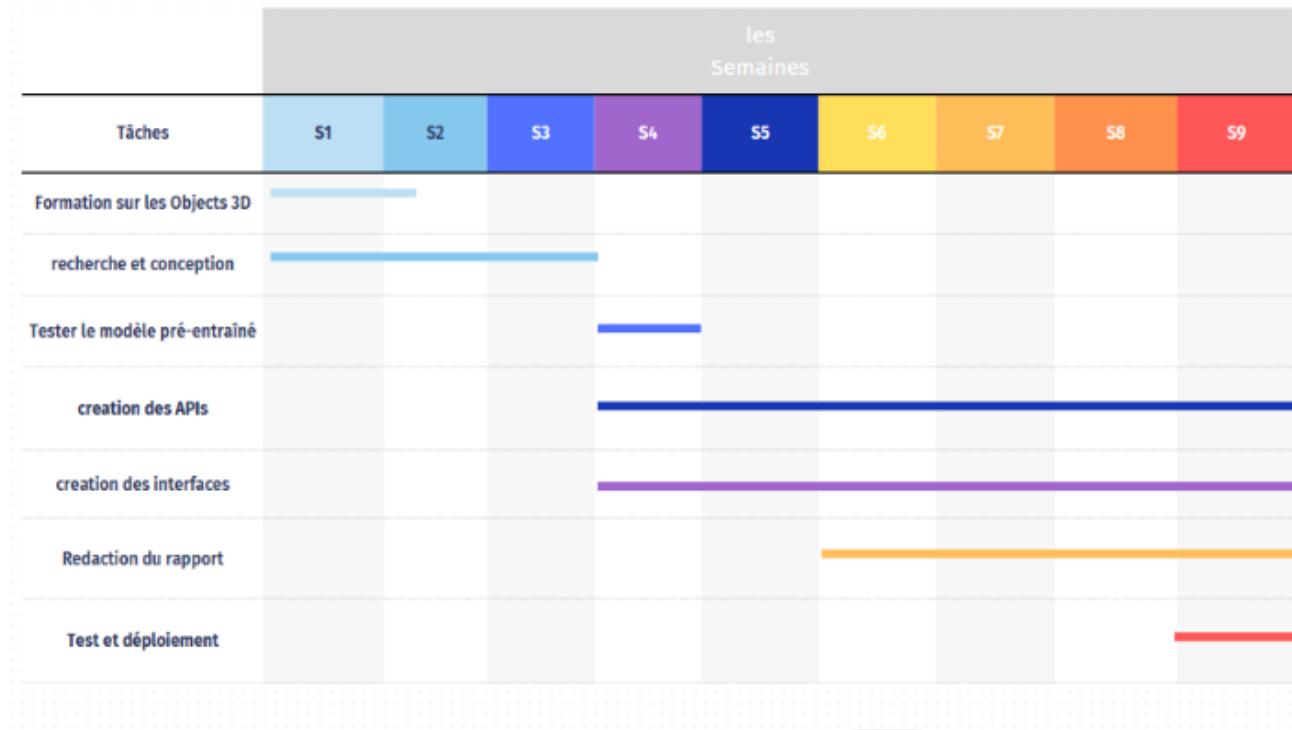


FIGURE 3.2 – Notre Diagramme de Gantt

Ce chapitre a été le point de départ de la présentation de notre projet, car il décrit son contexte général en présentant l'organisme d'accueil, le cadre du projet, les objectifs généraux à atteindre et la planification de sa mise en œuvre.

Deuxième partie

Segmentation 3D

CHAPITRE 4

La notion du 3D

4.1 Introduction

Dans ce chapitre, nous allons présenter une perspective mathématique et informatique du concept en 3D, dans le but d'acquérir une compréhension fondamentale de la perception du monde en trois dimension, puis on va entamer le vif de notre sujet : la notion de segmentation des modèles 3D, les types de segmentations et la technique MeshSegNet et ses différentes couches.

4.2 Définition

Trois dimensions, souvent abrégées en 3D, sont des termes qui décrivent l'espace qui nous entoure. Au quotidien, nous observons de multiples formes en trois dimensions, telles que le cube, le parallélépipède rectangle, le cône et le cylindre. Ces dimensions permettent de définir un objet en spécifiant trois axes : x, y et z, correspondent respectivement à la longueur, à la hauteur et à la profondeur.

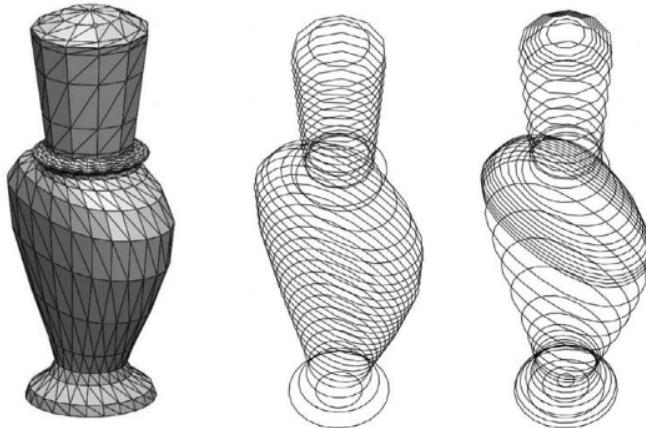


FIGURE 4.1 – Objet présenté en trois dimensions x,y,z

4.3 Contenu des fichiers 3D

Tout d'abord, nous examinons le contenu des données en trois dimensions (3D). Ce contenu 3D peut être regroupé en trois catégories principales : informations sur la géométrie, sur l'apparence, sur la scène et sur l'animation. Par la suite, nous détaillons chacune de ces catégories :

- **Géométrie** : La géométrie ou maillage polygonal est un ensemble de sommets, d'arêtes et de faces qui forment un objet 3D. Il peut s'agir d'une voiture, d'une arme, d'un environnement, d'un personnage ou de tout autre élément visuel d'un jeu.
- **Apparence** : L'apparence ou la texture, se rapporte aux propriétés visuelles d'un objet 3D telles que la couleur, la rugosité, la brillance, la transparence, etc. Les textures sont souvent utilisées pour créer des surfaces plus réalistes en ajoutant des motifs, des images ou des effets spéciaux.
- **Scène** : La scène dans un fichier 3D fait référence à la disposition d'un modèle en ce qui concerne la caméra, les lumières et d'autres modèles 3D. La scène est un élément important d'un fichier 3D car elle permet de définir l'environnement dans lequel les objets 3D sont affichés.
- **Animation** : Les animations sont utilisées pour donner vie à l'objet en le faisant bouger ou en le déformant. Les animations peuvent inclure des mouvements de translation, de rotation et de mise à l'échelle des objets, ainsi que des déformations de la géométrie.

4.4 Représentation mathématique des formes 3D

En imagerie en deux dimensions (2D), nous disposons d'une représentation unique avec une structure de données régulière (tableaux de pixels). Lorsque nous examinons les différentes représentations des données en trois dimensions (3D), nous pouvons les classer en deux grandes catégories : les données structurées euclidiennes et les données non euclidiennes.

4.4.1 Les données euclidiennes

Ces représentations sont qualifiées d'euclidiennes car elles sont basées sur une structure de géométrie euclidienne. Dans cette géométrie, les distances et les angles sont mesurés se-

lon les principes de la géométrie euclidienne, où la distance entre deux points correspond à la longueur de la ligne droite les reliant, et les angles sont déterminés par les intersections de lignes droites.

Représentation par voxel (volumique) : Le voxel est un pixel dans un espace tridimensionnel, il est similaire aux pixels d'une image en deux dimensions ; il peut être considéré comme une unité cubique de base 3D pouvant être utilisée pour représenter des modèles 3D. Les voxels sont utilisés pour modéliser les données 3D en décrivant comment l'objet 3D est distribué à travers les trois dimensions de la scène.

Représentation Multi View : connue sous le nom de représentation à plusieurs vues, est une méthode utilisée pour représenter des objets 3D en utilisant plusieurs vues ou images 2D de cet objet prises sous différents angles ou perspectives.

Les données 3D peuvent être présentées comme une combinaison de plusieurs images 2D capturées pour l'objet 3D à partir de différents points de vue. La représentation des données 3D de cette manière permet d'apprendre des ensembles de caractéristiques multiples pour réduire l'effet de bruit, l'incomplétude, d'occlusion et d'illumination des données capturées. L'apprentissage de données 3D à partir d'images 2D du même objet vise à apprendre une fonction modélisant chaque vue séparément, puis à optimiser conjointement toutes les fonctions pour représenter le même objet puis optimiser conjointement toutes les fonctions pour représenter la forme 3D entière et permettre la généralisation à d'autres formes 3D.

4.4.2 Les données non euclidiennes

Le deuxième type de représentations de données 3D est celui des données non euclidiennes, ces représentations ne se basent pas entièrement sur une géométrie euclidienne il est par exemple théoriquement possible d'avoir plusieurs points au même endroit dans ces représentations.

Nuage de Point (Point Cloud) : Les représentations basées sur des points décrivent les objets 3D en utilisant un ensemble de points dans l'espace 3D chaque point peut avoir des attributs tels que la position, la couleur ou la densité. La plupart des techniques d'apprentis-

sage s'efforcent de capturer les caractéristiques globales d'un objet pour effectuer des tâches complexes telles que la reconnaissance, la correspondance, l'appariement ou la récupération.

Les maillages Polygones : Un maillage polygone est une représentation géométrique d'un objet 3D qui est composé de sommets, d'arêtes et de faces polygonales. Les maillages polygonaux sont largement utilisés en modélisation 3D pour représenter des objets de formes complexes, ils peuvent être subdivisés en maillages triangulaires et en maillages quadrilatéraux, selon la forme de leurs faces. La figure ci-dessous montre la structure d'un maillage polygonal :

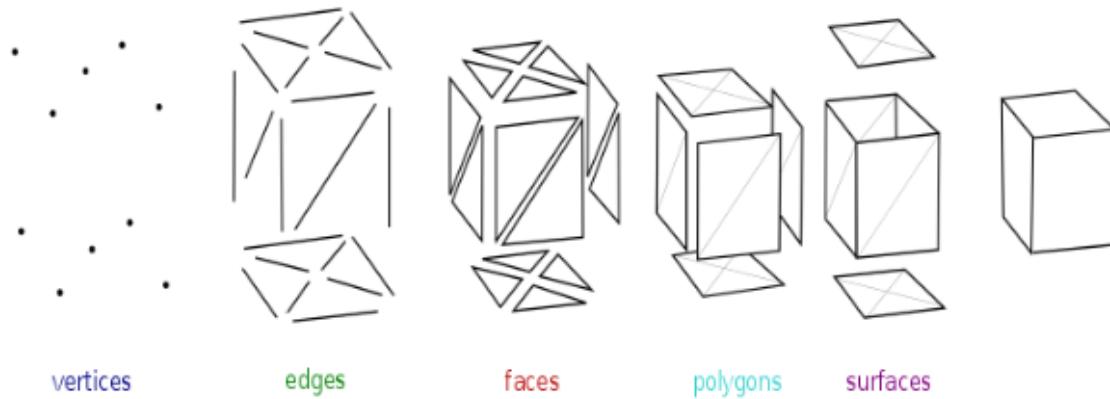


FIGURE 4.2 – Structure de maillage polygonal

Une structure de données destinée à représenter un maillage doit pouvoir stocker différents types d'éléments, tels que des sommets, des arêtes et des faces.

- **Sommet (Vertex) :** Les sommets, également appelés "vertices" en anglais, sont des points dans le plan ou dans l'espace tridimensionnel. Ils forment les éléments fondamentaux d'un maillage polygonal. Chaque sommet représente une position dans l'espace à travers les coordonnées x, y et z. De plus, ils peuvent être dotés de propriétés supplémentaires telles que la couleur et les normales pour l'affichage.

- **Arête (Edge) :** Les arêtes, aussi connues sous le nom d'"edges" en anglais, sont les liaisons entre deux sommets dans l'espace. Elles se situent sur la frontière d'une face et connectent deux sommets.

- **Face :** Les faces sont des polygones dans le plan ou dans l'espace, dont les sommets et les arêtes font partie du maillage. Les faces peuvent également posséder des propriétés comme la couleur ou la transparence pour l'affichage.

4.5 Les formats de fichier 3D

Un format de fichier 3D est simplement un type de fichier qui renferme des informations relatives aux modèles en trois dimensions (3D). Ces informations incluent la géométrie (forme) du modèle, son apparence (couleur, texture et/ou matériaux), ainsi que la scène (positionnement des sources lumineuses, de la caméra et d'autres objets autour du modèle 3D).

4.5.1 Fichiers VTK

Le Visualization Toolkit (VTK) est un système logiciel libre pour l'infographie 3D, la modélisation, le traitement d'images, le rendu de volume, la visualisation scientifique et le tracé 2D. Il prend en charge une grande variété d'algorithmes de visualisation et de techniques de modélisation avancées, et il tire parti du traitement en parallèle de la mémoire filetée et distribuée pour la vitesse et l'évolutivité, respectivement.

VTK fournit également certains de ses propres formats de fichiers. La principale raison de la création d'un autre format de fichier de données est d'offrir un schéma de représentation des données cohérent pour une variété de types d'ensembles de données, et de fournir une méthode simple pour communiquer les données entre les logiciels. Il existe deux styles différents de formats de fichiers disponibles dans VTK : **formats hérités simples et formats de fichiers XML**.

Les formats simples sont les anciens formats qui sont facile à lire et à écrire à la main ou par programmation. Les fichiers basés sur XML prennent en charge l'accès aléatoire et la compression de données portables.

4.5.2 Fichiers OBJ

Le format de fichier OBJ a été développé par WaveFront Technologies et avait initialement une utilisation dans le domaine de la conception graphique, servant de format d'échange. Cependant, OBJ est un format de fichier en texte brut (ASCII) qui contient des informations concernant la géométrie 3D, les matériaux et les textures. Les fichiers OBJ peuvent également contenir des informations sur les normales des faces, les coordonnées de texture et les données relatives aux matériaux.

The "v" identifies this element as a vertex

The "f" identifies this element as a face

```
v 1.000000 0.000000 0.000000
v 0.000000 1.000000 0.000000
v 0.000000 0.000000 1.000000
v -1.000000 0.000000 0.000000
v 0.000000 -1.000000 0.000000
v 0.000000 0.000000 -1.000000

f 1 2 3
f 2 4 3
f 4 5 3
f 5 1 3
f 2 1 6
f 4 2 6
f 5 4 6
f 1 5 6
```

Vertex coordinates (x,y,z)

Definition of faces from vertex indices (1st vertex is 1)

FIGURE 4.3 – Structure d'un fichier OBJ

La notion de segmentation

5.1 Définition

La segmentation d'un objet 3D consiste à le décomposer en plusieurs sous objets élémentaires cette décomposition pourra faciliter les opérations de traitement de cet objet; en effet la segmentation 3D est devenu actuellement un outil de pré-traitement indispensable aux applications liées à l'utilisation des objets 3D (reconnaissance de forme, compression, la réalité augmentée...).

5.2 Types de segmentations

→ **La segmentation sémantique :** La segmentation sémantique vise à prédire les étiquettes de classe des objets, telles que table et chaise comme illustré sur la figure 4.1 (f).

→ **La segmentation d'instance :** Les méthodes de segmentation d'instances permettent en outre de distinguer différentes instances d'une même classe, par exemple table une/deux et chaise une/deux comme illustré sur la figure 4.1 (g)

→ **La segmentation des parties :** La segmentation des parties vise à décomposer les instances en différentes composantes, comme les accoudoirs, les pieds et le dossier d'une même chaise comme illustré sur la figure 4.1 (h)

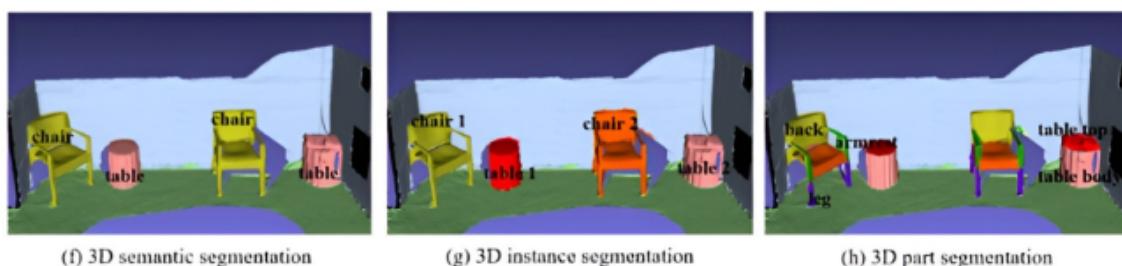


FIGURE 5.1 – Types de segmentations

CHAPITRE 6

MeshSegNet

Il existe de nombreuses techniques de segmentation 3D basées sur différentes représentations, à savoir les maillages, les nuages de points et les voxels. Chaque représentation offre ses propres avantages et défis, et les méthodes de segmentation varient en fonction de la nature de la donnée 3D utilisée. Dans notre projet on a choisi la technique MeshSegNet qui utilise des réseaux de neurones convolutifs profonds pour identifier et classifier les différentes parties de l'objet. Cette technique est particulièrement utile dans la modélisation 3D, la réalité virtuelle et la robotique, où la segmentation précise des différentes parties d'un objet est essentielle pour la planification et l'exécution de tâches complexes.

6.1 L'architecture de MeshSegNet

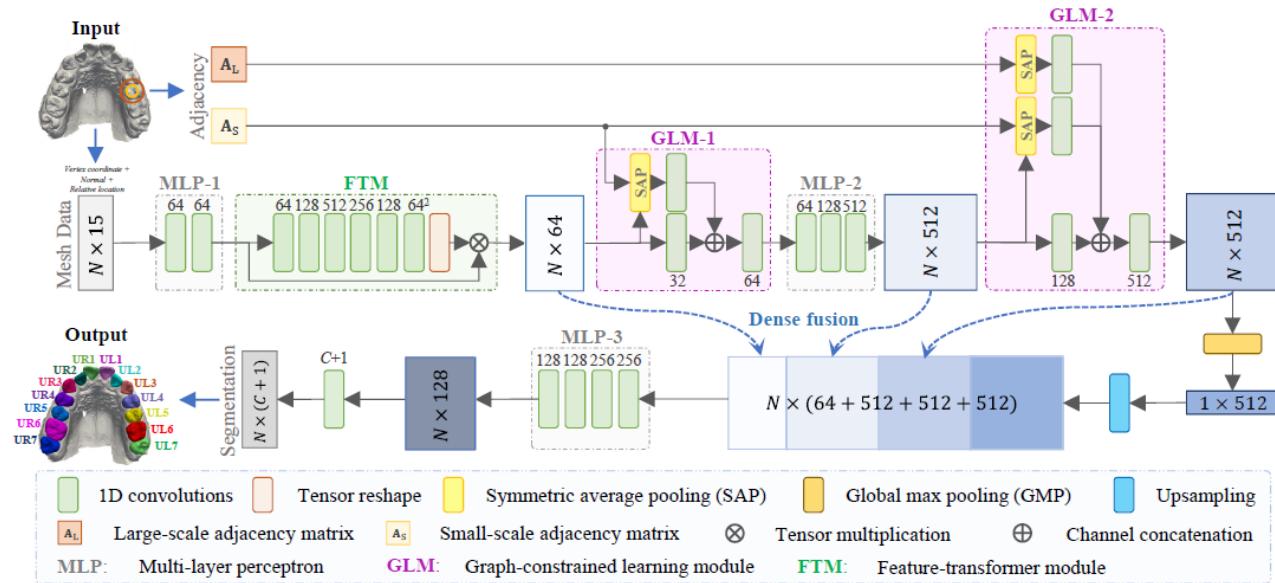


FIGURE 6.1 – Architecture de MeshSegNet

MeshSegNet est un modèle de réseau neuronal profond conçu de manière intégrale pour extraire directement les caractéristiques géométriques avancées à partir des données non traitées d'un maillage dentaire. Son objectif est de faciliter la segmentation automatique des dents.

Le modèle MeshSegNet est présenté comme un réseau neuronal profond multi-échelle capable d'apprendre des caractéristiques géométriques avancées pour la segmentation complète des dents sur des surfaces dentaires 3D. Il se compose de plusieurs modules d'apprentissage (GLM) qui sont intégrés le long du chemin avant pour extraire de manière hiérarchique des caractéristiques contextuelles locales multi-échelles à partir de données brutes de surface dentaire. Chaque GLM est composé de plusieurs sous-modules, notamment des couches convolutionnelles, des couches de normalisation par lots et des fonctions d'activation. Les couches de convolutions sont utilisées pour extraire les caractéristiques des données d'entrée, tandis que les couches de normalisation par lots aident à normaliser les cartes de caractéristiques et à améliorer la stabilité de l'entraînement. Les fonctions d'activation introduisent la non-linéarité dans le réseau et aident à capturer les relations complexes entre les caractéristiques.

6.1.1 Les couches de MeshSegNet

Les couches convolutionnelles : sont utilisées pour extraire des caractéristiques à partir des données brutes de surface dentaire. Ces couches appliquent une opération de convolution aux données d'entrée pour produire des cartes de caractéristiques qui représentent différentes caractéristiques de la surface dentaire.

Dans MeshSegNet, la convolution utilisée est la convolution 1D. Convs 1D désigne les couches convolutionnelles unidimensionnelles. Le nombre de canaux dans chaque couche convulsive détermine le nombre de filtres appliqués aux données d'entrée. La sortie de chaque couche convulsive est ensuite passée par la fonction d'activation ReLU.

Kernel : La convolution est définie par un noyau (kernel) d'image. Le noyau image n'est rien de plus qu'une petite matrice. La plupart du temps, une matrice noyau 3x3 est très courante.

Dans la figure ci-dessous, la matrice verte est l'image originale et la matrice mobile jaune est appelée noyau (kernel), qui est utilisé pour apprendre les différentes caractéristiques de l'image originale. Le noyau se déplace d'abord horizontalement, puis décale vers le bas et se déplace de nouveau horizontalement. La somme du produit point de la valeur de pixel d'image et de la valeur de pixel du noyau donne la matrice de sortie. Initialement, la valeur du noyau s'initialise de manière aléatoire et c'est un paramètre d'apprentissage

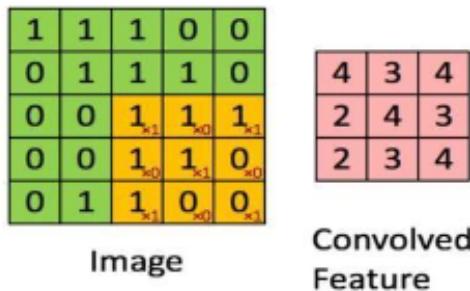


FIGURE 6.2 – L'image Kernel

Padding : La convolution pose deux problèmes :

- Chaque fois après l'opération de convolution, la taille originale de l'image rétrécit, comme nous l'avons vu dans l'exemple ci-dessus six par six jusqu'à quatre par quatre et dans la tâche de classification d'image il y a plusieurs couches de convolution, donc après opération de convolution multiple, notre image originale deviendra vraiment petite mais nous ne voulons pas que l'image rétrécit à chaque fois.
- Le deuxième problème est que, lorsque le noyau se déplace sur les images originales, il touche le bord de l'image moins de fois et touche le milieu de l'image plus de fois et il se chevauche également au milieu.

Ainsi, les caractéristiques de coin de n'importe quelle image ou sur les bords ne sont pas utilisées beaucoup dans la sortie. Ainsi, afin de résoudre ces deux problèmes, un nouveau concept est introduit appelé padding. Padding préserve la taille de l'image originale. Le padding (rembourrage) est une opération d'augmentation de la taille des données d'entrée. Pour les données à 1 dimension, le tableau est ajouté/préfixé avec une constante. En n-dimensions, on entoure n-dim hypercube avec la constante. Dans la plupart des cas, cette constante est égale à zéro et appelée zero-padding.

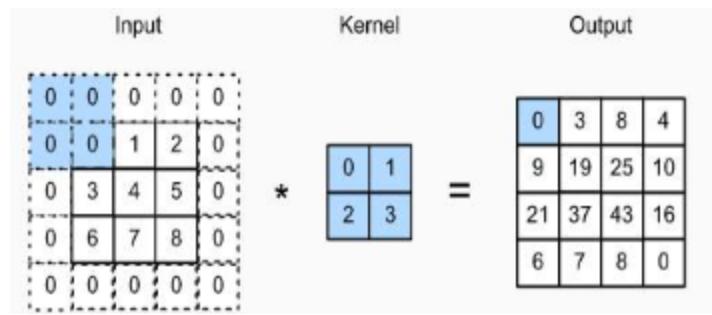


FIGURE 6.3 – Image Kernel rembourrée et convoluée avec 2*2

Le stride : Le stride est un composant des réseaux neuronaux convolutifs adaptés à la compression d’images et un paramètre du filtre du réseau neuronal qui modifie la quantité de mouvement sur l’image. Stride est le nombre de décalages de pixels sur la matrice d’entrée. Par exemple, si le pas d’un réseau neuronal est défini sur le 1, le filtre se déplace d’un pixel, ou unité, à la fois. La taille du filtre affecte le volume de sortie codé, c’est pourquoi le stride est souvent défini sur un nombre entier, plutôt que sur une fraction ou une décimale.

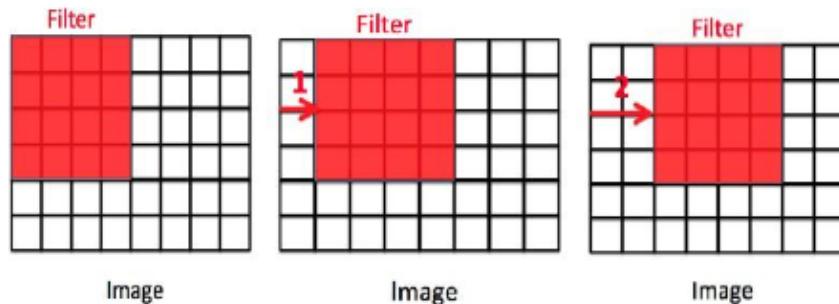


FIGURE 6.4 – Image de gauche : stride = 0 - Du milieu : stride = 1 - De droite : stride= 2

Le poids : Dans les couches conventionnelles, les poids sont représentés comme le facteur multiplicatif des filtres.

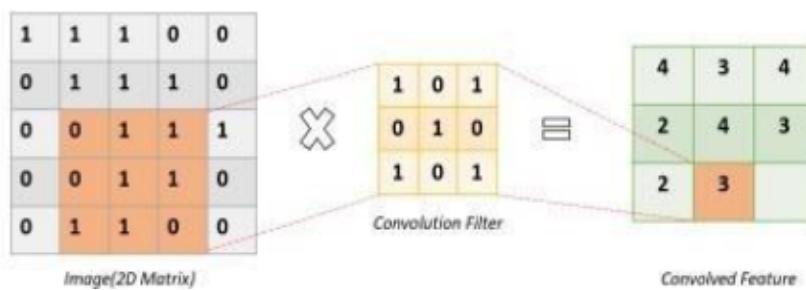


FIGURE 6.5 – Exemple de convolution avec poids/filter 3X3

Supposant qu'on dispose d'une matrice 2D d'entrée et un filtre de convolution. Chaque élément de la matrice dans le filtre de convolution correspond aux poids qui sont formés. Ces poids auront un impact sur les caractéristiques de convolution extraites. Sur la base des caractéristiques résultantes, nous obtenons alors les sorties prédites et nous pouvons utiliser la rétropropagation pour entraîner les poids dans le filtre de convolution.

Le bias : La fonction d'activation dans les réseaux neuronaux prend une entrée x multipliée par un poids w . Le Bias permet de décaler la fonction d'activation en ajoutant une constante (c'est-à-dire le biais donné) à l'entrée. Le bias dans les réseaux neuronaux peut être considéré comme analogue au rôle d'une constante dans une fonction linéaire, la ligne étant effectivement transposée par la valeur constante.

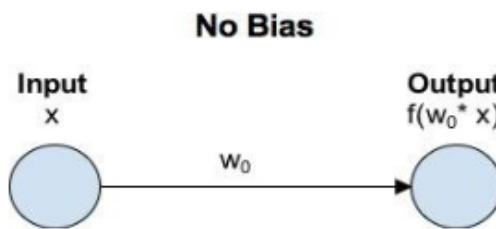


FIGURE 6.6 – Sans Bias

Pour sans biais, l'entrée de la fonction d'activation est x multipliée par le poids de connexion.

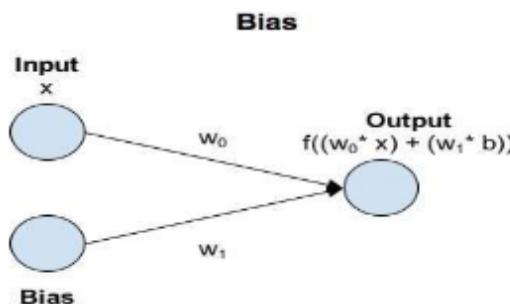


FIGURE 6.7 – Avec Bias

Pour avec biais, l'entrée à la fonction d'activation est x multipliée par le poids de connexion w_0 plus le biais multiplié par le poids de connexion pour le biais w_1 . Cela a pour effet de déplacer la fonction d'activation par une quantité constante ($b * w_1$). Si le biais est vrai, les valeurs de ces poids sont échantillonnées.

Bias = (Tensor) - le biais apprenable du module de forme (out Channel).

Les couches de normalisation par lots : sont utilisées pour normaliser les cartes de caractéristiques produites par les couches convolutionnelles. Cela permet de stabiliser l'apprentissage et d'accélérer la convergence vers une solution optimale.

Dans MeshSegNet, les couches de normalisation par lots sont insérées après chaque couche convective dans les MLP pour normaliser les cartes de caractéristiques et réduire le décalage des variables internes. En normalisant les cartes de caractéristiques, la normalisation par lots permet de réduire le décalage de variable interne et d'améliorer la stabilité de l'entraînement. Cela permet également des taux d'apprentissage plus élevés et une convergence plus rapide pendant l'entraînement.

Les fonctions d'activation sont utilisées pour introduire de la non-linéarité dans le modèle. Elles sont appliquées après chaque couche convolutionnelle et de normalisation par lots pour introduire de la non-linéarité dans les cartes de caractéristiques.

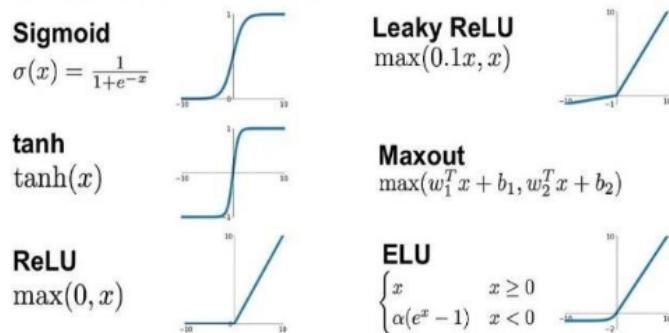


FIGURE 6.8 – Différentes fonctions d'activation et leurs graphiques

La fonction d'activation Relu : ReLU (Rectified Linear Unit) est une fonction d'activation non linéaire couramment utilisée dans les réseaux de neurones. ReLU est défini comme suit :

$$Relu(x) = \begin{cases} 0, & \text{Si } x < 0 \\ x, & \text{Si } x \geq 0 \end{cases}$$

FIGURE 6.9 – Fonction d'activation ReLU

où x (caractéristique) est l'entrée de la fonction.

Si l'entrée est négative, la sortie sera 0, et si l'entrée est positive, la sortie sera la valeur d'entrée. Le principal avantage de la fonction ReLU par rapport aux autres fonctions d'activation est qu'elle n'active pas tous les neurones en même temps.

Les perceptrons multicouches (Multi-Layer Perceptron) : sont utilisés pour extraire des caractéristiques géométriques à partir des données d'entrée. Ils sont composés de plusieurs couches entièrement connectées et de fonctions d'activation. Le perceptron possède une fonction d'activation Relu qui impose un seuil.

Dans MeshSegNet, chaque MLP se compose de plusieurs couches entièrement connectées, également appelées couches denses, qui appliquent des transformations linéaires aux données d'entrée et le font passer par des fonctions d'activation ReLU pour permettre une meilleure optimisation à l'aide de la descente de gradient stochastique, un calcul plus efficace et invariant à l'échelle.

MLP / Couches	C-1 (C_{in}, C_{out})	C-2 (C_{in}, C_{out})	C-3 (C_{in}, C_{out})	C-4 (C_{in}, C_{out})
MPL-1	15, 64	64, 64		
MPL-2	64, 64	64, 128	128, 512	
MPL-3	256, 256	256, 256	256, 128	128, 128

FIGURE 6.10 – Couches de convolution dans chaque MLP

où $C - i$: Couche i

Le MLP-1 est utilisé pour extraire des caractéristiques des données d'entrée. MLP-1 contient des couches entièrement connectées (64, 64) et produit une caractéristique spatiale initiale de longueur 64 (Nx64).

Ensuite, le MLP-2 est utilisé pour extraire des caractéristiques multi-échelles des données d'entrée et des caractéristiques extraites par GLM-1. MLP-2 contient des couches entièrement connectées (64, 128, 512) et produit une caractéristique spatiale initiale de longueur 512 (Nx512). Enfin, il y a le MLP-3 qui est utilisé pour extraire des caractéristiques hautement discriminantes à partir des caractéristiques fusionnées provenant de l'étape Upsample MLP-3 contient des couches entièrement connectées (256, 256, 128, 128) et produit une caractéristique spatiale initiale de longueur 128 (Nx128).

Les caractéristiques résultantes sont ensuite combinées en utilisant des connexions de saut denses pour produire une matrice de caractéristiques finale qui est utilisée pour l'étiquetage des dents.

6.1.2 Upsampling

Dans MeshSegNet, le suréchantillonnage ou Upsampling est effectué en utilisant la méthode d'interpolation des K plus proches voisins ou K-Nearest Neighbors (KNN). Cette méthode est utilisée pour augmenter la résolution spatiale des cartes de caractéristiques et produire des cartes de segmentation avec la même résolution que la structure de données de maillage d'entrée.

Plus précisément, après la fusion dense de caractéristiques locales à globales de FTM, GLM-1, GLM-2 et GMP, la matrice de caractéristiques résultante est traitée par MLP-3 pour produire une matrice de caractéristiques $N \times 128$. Cette matrice de caractéristiques est ensuite échantillonnée à l'aide de l'algorithme KNN pour produire une nouvelle matrice de caractéristiques avec le même nombre de cellules que la structure de données du maillage d'entrée.

Le fonctionnement de KNN : En trouvant les voisins k les plus proches de chaque cellule dans la structure de données de maillage d'entrée et en utilisant leurs caractéristiques correspondantes pour ajouter un nouveau vecteur de caractéristique pour cette cellule. Où k détermine le nombre de cellules voisines utilisées.

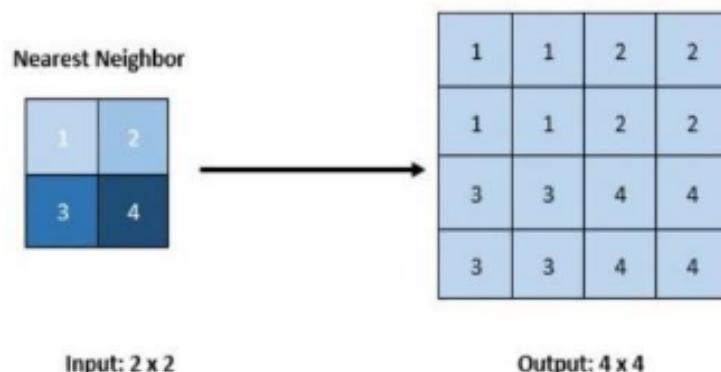


FIGURE 6.11 – KNN Upsampling

En utilisant KNN, MeshSegNet peut produire la segmentation avec une haute résolution spatiale qui reflètent avec précision les détails géométriques de la structure de données de maillage d'entrée.

6.1.3 Downsampling

Le GMP (global max pooling) est un type d'opération de pooling dans MeshSegNet qui lui permet de capturer les caractéristiques géométriques globales de la structure de données de maillage d'entrée et de les utiliser pour la segmentation, GMP est appliqué à la sortie de GLM-2 (N^512) pour produire des fonctionnalités holistiques invariantes par translation qui codent l'information sémantique de toute la surface d'entrée.

Le pooling fonctionne en réduisant la taille spatiale de la représentation et en créant aussi une forme d'invariance par translation. Par rapport à d'autres types d'opérations de pooling, telles que average-pooling ou max-pooling avec une taille de kernel (noyau) fixe, GMP est plus flexible et peut traiter des entrées avec des résolutions spatiales variables. Cela le rend bien adapté pour traiter des structures de données de maillage avec une connectivité irrégulière et un nombre variable de cellules.

Max pooling est simplement une règle pour prendre le maximum d'une région et il aide à procéder avec les caractéristiques les plus importantes. Tandis que Average-pooling conserve beaucoup d'informations sur les éléments moins importants d'un bloc ou d'un pool.



FIGURE 6.12 – Max-pooling Vs. Average-pooling

Max-pooling : On applique le max-pooling sur une entrée de 4x4. Cette entrée sera en premier lieu divisée en quatre régions. Donc, la sortie est 2x2, chacune des sorties sera juste le maximum de la région ombrée correspondante.

Average-pooling : De même, on applique average-pooling sur une entrée de 4x4. Cette entrée sera en premier lieu divisée en quatre régions. Donc, la sortie est 2×2, chacune des sorties sera juste la moyenne de la région ombrée correspondante.

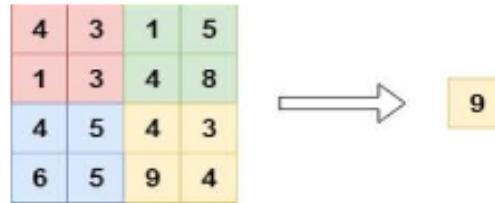


FIGURE 6.13 – Global Max-pooling

Global max-pooling : Lorsqu'on applique le global max-pooling, on passe par le max-pooling. Ainsi, on aura une sortie de 2x2 qu'on divise sur quatre régions. D'où, la sortie de 1×1, correspond au maximum parmi les quatre régions.

Pour plus d'informations sur MeshSegNet, voici l'article scientifique correspondant :
<https://arxiv.org/pdf/2109.11941.pdf>

Dans ce chapitre, nous avons tout d'abord abordé la notion de dimensionnalité 3D, ainsi que la représentation mathématique des formes en 3D, qu'elles soient euclidiennes ou non euclidiennes. De plus, nous avons examiné en détail les formats de fichiers 3D, en mettant en avant le format de fichier de type obj et vtk. Ensuite nous avons présenté la segmentation d'un objet 3D. Nous avons cité les trois types de segmentations. Nous avons également vu la technique MeshSegNet et ses différentes couches.

Troisième partie

Analyse et Conception

Analyse

7.1 Introduction

Dans ce chapitre, nous entamons la partie conceptuelle qui est considérée comme une étape critique et nécessaire avant de se lancer dans le développement du projet. Tout d'abord, nous utiliserons la modélisation UML pour présenter des diagrammes de cas d'utilisation et quelques diagrammes de séquence pour une meilleure compréhension. Ensuite, nous présenterons l'architecture REST et son intégration dans une application web qui est un choix courant pour de nombreux projets, finalement, nous décrivons les technologies utilisées.

7.2 Etude des besoins

En informatique, une bonne spécification des besoins est très importante dans la réalisation de tout projet. Elle représente le travail le plus délicat et le plus significatif, elle repose sur une bonne méthode qui n'est autre que les questions que doit se poser tout développeur au début de ses travaux. On peut retenir que cette phase consiste à mieux comprendre le contexte du système à concevoir. Il s'agit-là de déterminer les fonctionnalités, les acteurs et les cas d'utilisations. On distingue deux principaux besoins :

- Les besoins fonctionnels.
- Et les besoins non fonctionnels.

7.2.1 Besoins fonctionnels

Les besoins fonctionnels représentent les principales fonctions du système. Ces besoins proviennent de l'utilisateur du système.

Notre application devrait permettre à l'utilisateur de :

- Charger un fichier .obj au niveau de l'application.
- Visualiser le fichier dans la scène destinée à cet effet.
- Segmenter le fichier une fois que l'utilisateur clique sur le bouton de segmentation.
- Télécharger le fichier sous format .vtp, et d'avoir un aperçu de ce dernier au niveau de la scène.
- Visualiser les fichiers .vtp déjà segmenter.

7.2.2 Besoins non fonctionnels

A part les besoins fonctionnels de notre application, notre système doit répondre aux critères suivants :

- Avoir une interface stylé, fluide et simple à utiliser.
- Performance et rapidité : le temps de traitement et de réponse doit être rapide afin d'optimiser l'utilisation de notre application.
- Fiabilité des données : Les informations doivent être claires et précises.
- Maintenance et réutilisabilité : Notre code source de l'application doit être assez compréhensible et modulaire pour faciliter sa maintenance technique.
- Disponibilité.

7.3 Diagrammes UML

UML est un langage standard conçu pour l'écriture de plans d'élaboration de logiciels il peut être utilisé pour visualiser, spécifier, construire et documenter les artefacts d'un système à forte composante logicielle. C'est un langage très expressif qui couvre toutes perspectives nécessaires au développement puis au déploiement des systèmes.

7.3.1 Diagramme de cas d'utilisation

Diagramme de cas d'utilisation est un diagramme UML, qui permet de représenter le fonctionnement du système vis-à-vis de l'utilisateur : c'est donc une vue du système dans son environnement extérieur. Voici le diagramme de cas d'utilisation :

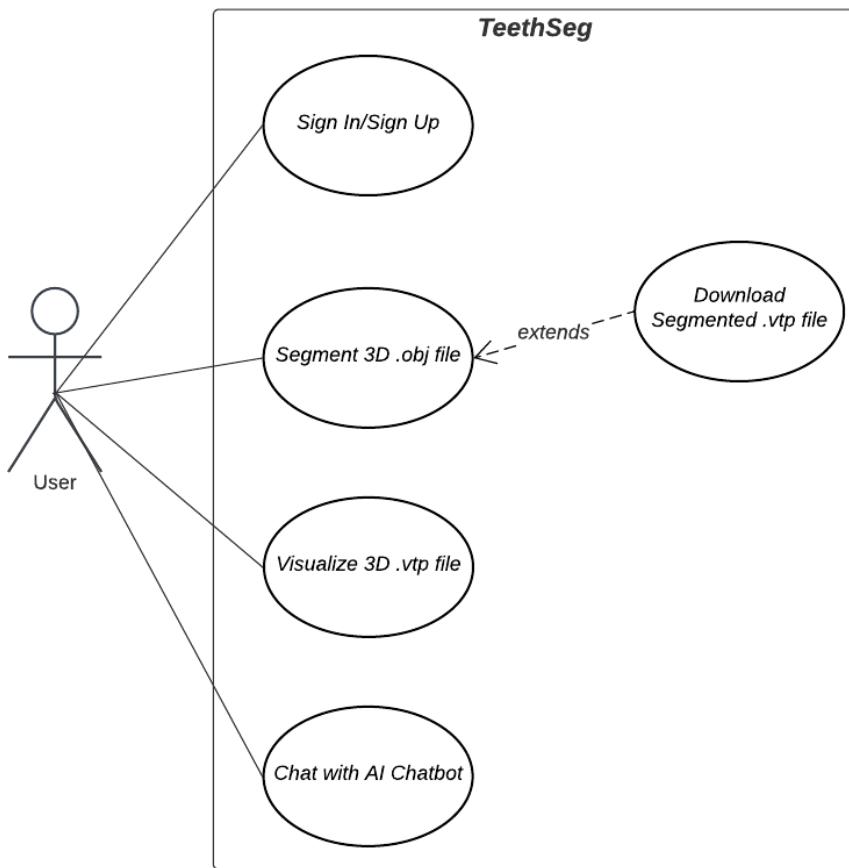


FIGURE 7.1 – Diagramme de cas d'utilisation

7.3.2 Diagramme de séquence

Le diagramme de séquence est la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation UML. Les figures ci-dessous représentent les diagrammes que nous avons effectués. Le diagramme de séquence des différents formats de fichiers :

Diagramme de séquence de segmentation d'objets 3D

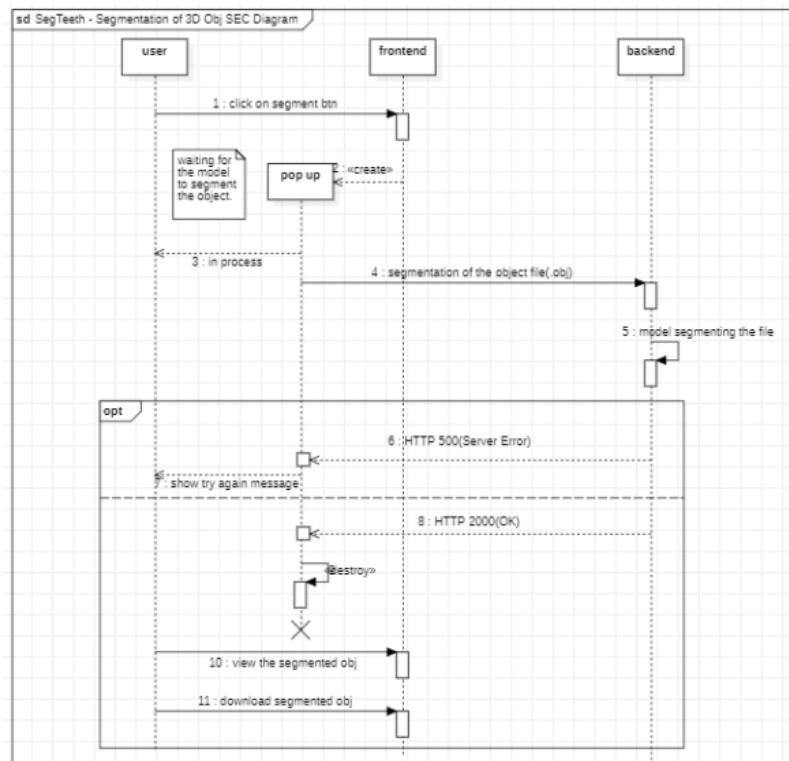


FIGURE 7.2 – Diagramme de séquence de segmentation d'objets 3D

Diagramme de séquence de téléchargement des fichiers vtp 3D

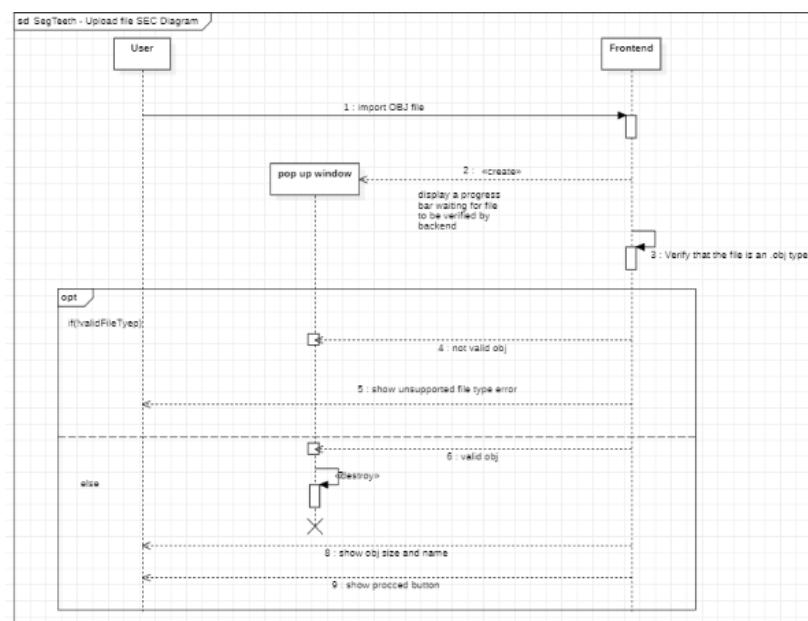


FIGURE 7.3 – Diagramme de séquence de téléchargement des fichiers vtp 3D

7.4 Pipelines CI/CD

Un pipeline CI/CD est un flux de travail automatisé qui facilite le processus de livraison du modèle à la production. Il comprend généralement les étapes suivantes :

1. Source : Les modifications de code sont envoyées vers un dépôt de code source pour déclencher le processus de pipeline CI/CD.

2. Build : Le code source est construit avec ses dépendances en paquets et exécutables.

3. Test : Des tests automatisés sont exécutés pour valider que le code fonctionne comme prévu; cela sert de filet de sécurité pour s'assurer que les bogues du code sont identifiés rapidement avant d'être déployés en production.

4. Déploiement : Le modèle est prêt à être déployé en production lorsqu'une instance exécutable du modèle a réussi tous les tests automatisés. CI/CD dans MLOps : Les modèles de ML se dégradent souvent en précision lorsqu'ils sont déployés dans des scénarios réels parce qu'ils ne s'adaptent pas aux changements dans les données. Pour maintenir la performance des modèles ML en production, vous devez surveiller activement la performance de votre modèle, le recycler avec des données plus récentes, et expérimenter continuellement avec différentes approches dans le cycle de développement ML.

Chaque étape de ce processus itératif est souvent manuelle et entraîne des frais généraux importants, car les ingénieurs ML doivent souvent refaire l'intégralité du pipeline de formation sur les modèles et produire les nouveaux modèles pour s'adapter aux changements de code et de données. Un flux de travail CI/CD pour les pipelines ML peut être décrit avec les deux concepts :

- Intégration continue, qui consiste en une construction et des tests automatisés.
- Livraison continue, qui consiste en un déploiement automatisé du pipeline pour la formation continue et la prestation de modèles de ML.

Intégration continue (CI) : Au cours du processus d'intégration continue, le pipeline ML et ses composants sont créés, testés et conditionnés pour être livrés lorsque des modifications sont apportées au référentiel de code source. Le processus CI du pipeline se compose de trois étapes : développement, construction et test. Au cours du processus de développement, vous expérimitez de manière itérative de nouveaux algorithmes ML et approches de modélisation où les étapes d'expérimentation sont orchestrées et suivies. Vous pouvez ensuite sélectionner un modèle approprié et transférer le code source du pipeline ML vers le référentiel de code source.

Livraison continue (CD) : Au cours du processus de livraison continue (CD) du pipeline, le pipeline CI/CD déploie continuellement de nouvelles implémentations de pipeline de ML qui, à leur tour, effectuent une formation continue (CT) des modèles de ML pour la prédiction. Lors de la TC du modèle de ML dans le processus de CD de pipeline, le pipeline ML déployé est automatiquement déclenché pour le recyclage du modèle en production en fonction des déclencheurs de l'environnement de pipeline ML en direct. Le modèle ML formé est ensuite déployé automatiquement en tant que service de prédiction de modèle dans le cadre du processus de modèle CD.

7.5 L'API REST

Une API REST (également appelée API RESTful) est une interface de programmation d'application (API ou API web) qui respecte les contraintes du style d'architecture REST et permet d'interagir avec les services web RESTful. L'architecture REST (Representational State Transfer) a été créée par l'informaticien Roy Fielding.

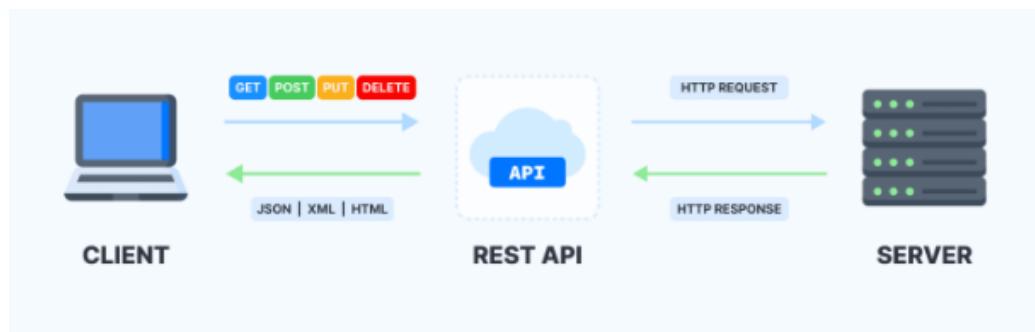


FIGURE 7.4 – Web Service Rest

Les APIs REST fonctionnent avec des requêtes au format HTTP. Il existe quatre types principaux de requêtes, on les appelle les requêtes “CRUD” : Create, Read, Update et Delete.

Requête	Méthode HTTP	Action demandée et effet
Create	POST	Ajouter une nouvelle donnée à la base de données du serveur
Read	GET	Récupérer une information auprès du serveur. Cette donnée sera retournée à un format standardisé (HTML, JSON, etc.)
Update	PUT	Effectuer une mise à jour dans la base de données du serveur
Delete	DELETE	Effacer une donnée sur le serveur

FIGURE 7.5 – Web Service Rest

Les avantages d'API REST :

Nous avons choisi l'API REST car elle offre plusieurs avantages :

→ **Polyvalence** : REST n'est pas limité à XML, mais peut renvoyer des formats XML, JSON, HTML, PYTHON, PHP ou texte en fonction de ce que le client demande.

→ **Flexibilité** : La conception d'API REST se distingue par son grand niveau de flexibilité. Puisque les données ne sont pas liées à des méthodes et à des ressources, REST a la capacité de traiter plusieurs types d'appels, de renvoyer différents formats de données et même de changer de structure avec la mise en œuvre correcte de systèmes hypermédia.

→ **Client-serveur** : L'API REST fonctionne sur un modèle client-serveur composé de clients, de serveurs et de ressources. En effet, elle joue un rôle d'intermédiaire entre le client et le serveur. Elle reçoit les requêtes du client, les transmet au serveur, récupère les réponses données par ce dernier et les renvoie au client. Pour émettre des requêtes, le client effectue des appels HTTP.

CHAPITRE 8

Conception

8.1 Logiciels et outils

Voici les différents logiciels, langages de programmation et librairies utilisés pour développer ce projet sont :

Trello

Trello est une application de gestion de projet en ligne qui utilise un système de tableau virtuel pour organiser des tâches et des projets. Les tâches sont représentées sous forme de cartes que vous pouvez déplacer entre différentes colonnes pour suivre leur progression. Trello facilite la collaboration entre les membres d'une équipe en permettant le partage de tableaux et de cartes, ainsi que la possibilité d'ajouter des commentaires et des dates d'échéance. C'est un outil visuel et intuitif pour la gestion de tâches et de projets.



FIGURE 8.1 – Logo de Trello

Voici un screenshot de notre utilisation de ce service par notre équipe :

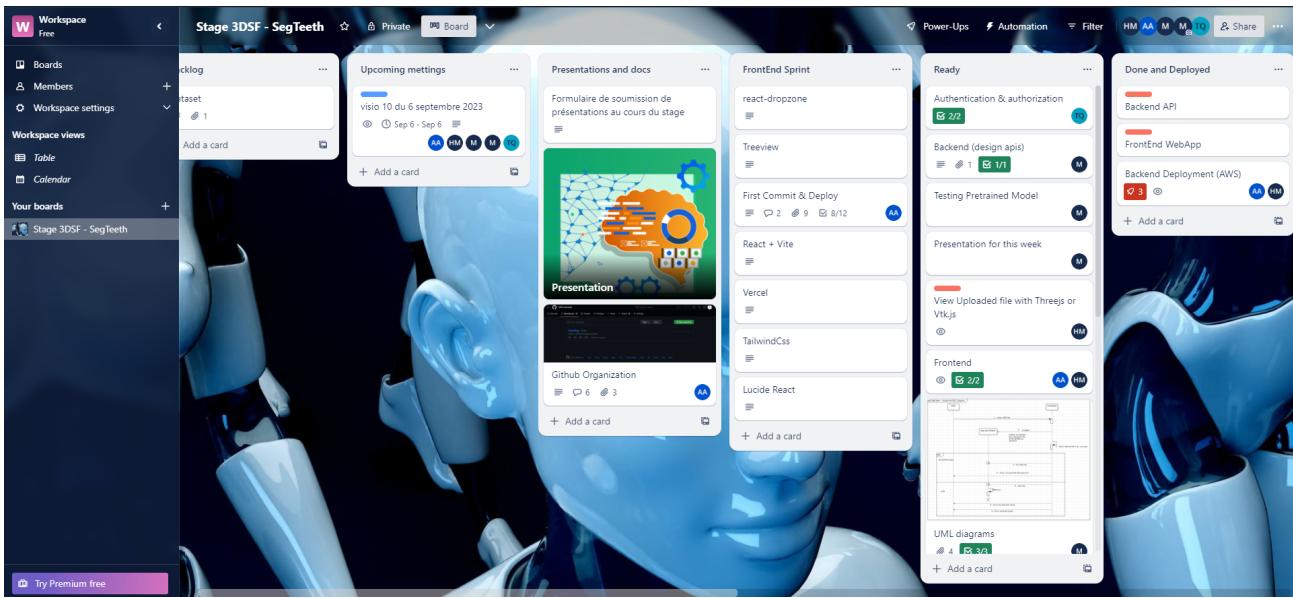


FIGURE 8.2 – Gestion du projet sur Trello

Figma

Figma est une plateforme de conception d’interfaces utilisateur (UI) et d’expérience utilisateur (UX) basée sur le cloud. C’est un outil populaire utilisé par les concepteurs, les développeurs et les équipes de conception pour créer, collaborer et prototyper des designs d’interfaces utilisateur interactives pour des applications, des sites web et d’autres produits numériques. Figma offre une interface de conception conviviale qui permet aux utilisateurs de créer des maquettes, des wireframes et des prototypes interactifs. On a utilisé Figma pour faire le design initial de l’application car elle facilite la collaboration à distance et la gestion des versions.



FIGURE 8.3 – Logo de Figma

StarUML

StarUML est un logiciel de modélisation UML (Unified Modeling Language) largement utilisé pour concevoir et créer des diagrammes de modèles logiciels. Il permet aux développeurs et aux ingénieurs de créer des représentations visuelles des structures, des interactions et des comportements des systèmes logiciels.



FIGURE 8.4 – Logo de StarUML

Visual Studio Code

Visual Studio Code (VSCode) est un éditeur de code source largement utilisé développé par Microsoft. Il est connu pour sa légèreté, ses options de personnalisation étendues et son support pour divers langages de programmation et technologies. VSCode propose des fonctionnalités telles que la mise en évidence du code, l'autocomplétion, les capacités de débogage, l'intégration de contrôle de version et un éco système d'extensions qui permettent aux développeurs d'améliorer leur fonctionnalité en fonction de leurs besoins spécifiques.

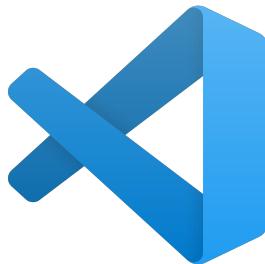


FIGURE 8.5 – Logo de VSCode

Google Colaboratory

Google Colab ou Colaboratory est un service cloud, offert par Google, basé sur Jupyter Notebook et destiné à la formation et à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entraîner des modèles de Machine Learning directement dans le cloud, elle vous permet d'écrire, d'exécuter et de partager du code Python dans votre navigateur.



FIGURE 8.6 – Logo de Google Colab

Github

GitHub est une plateforme de développement collaboratif basée sur le système de contrôle de version Git. Elle permet aux développeurs de travailler ensemble sur des projets, de partager du code source, de suivre les modifications, de gérer les problèmes et les demandes de fusion, et de collaborer de manière transparente sur des projets logiciels. En utilisant GitHub, vous pouvez héberger vos dépôts de code source, collaborer avec d'autres développeurs en facilitant le travail en équipe, examiner les modifications apportées au code, signaler et résoudre des problèmes, et gérer le cycle de vie de développement d'un projet.

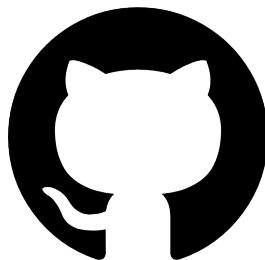


FIGURE 8.7 – Logo de Github

8.2 Front End

Vite

Vite est un outil front-end JavaScript qui vise à améliorer la rapidité de développement en offrant un serveur de développement rapide et une compilation optimisée pour la production. Il prend la suite d'une grande famille dans laquelle on peut évoquer Grunt, Gulp, et dernièrement Webpack. Vite utilise la fonctionnalité ES Module Import de JavaScript pour charger les modules de manière asynchrone, ce qui permet une expérience de développement plus rapide et une compilation (étape build) plus petite pour la production. Vite est également conçu pour être facile à utiliser et à configurer, ce qui en fait un choix populaire pour les développeurs de front-end.



FIGURE 8.8 – Logo de Vite

ReactJS

ReactJS est une bibliothèque JavaScript open-source pour la construction d'interfaces utilisateur interactives et réactives. Elle permet de créer des composants réutilisables qui gèrent leur propre état et s'actualisent automatiquement en réponse aux changements. Utilisée pour le développement web, React facilite la création d'applications réactives et efficaces en rendant la gestion de l'interface utilisateur plus modulaire et conviviale.

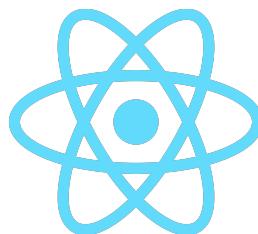


FIGURE 8.9 – Logo de ReactJS

TailwindCSS

Tailwind CSS est un framework permettant aux développeurs de personnaliser totalement et simplement le design de leur application ou de leur site web. Avec ce framework CSS, il est possible de créer un design d'interface au sein même du fichier HTML. Cette façon de programmer n'interfère pas avec les pratiques recommandées par le W3C comme celle de séparer le HTML des feuilles de style CSS.



FIGURE 8.10 – Logo de TailwindCSS

ThreeJS

Three.js est une bibliothèque JavaScript légère et puissante qui facilite la création et la manipulation d'objets et de scènes 3D interactives dans un navigateur web. La simplicité d'utilisation de Three.js, combinée à sa flexibilité et à sa grande communauté de développeurs, en fait un choix populaire pour les projets nécessitant des graphismes 3D dans un navigateur web. Elle est compatible avec la plupart des navigateurs modernes et permet de créer des expériences visuelles impressionnantes directement sur le web, sans nécessiter de plugins ou de logiciels tiers.

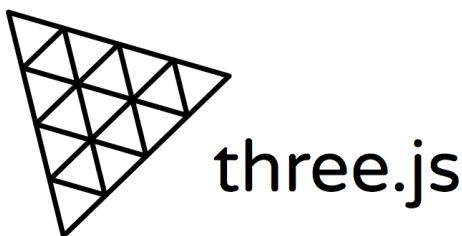


FIGURE 8.11 – Logo de ThreeJS

Vtk JS

Vtk.js est une bibliothèque JavaScript destinée à la visualisation et à la manipulation de données en 3D. Elle permet de créer des visualisations interactives, des rendus et des analyses de données scientifiques complexes directement dans un navigateur web. vtk.js est basée sur le kit de développement Visualization Toolkit (VTK) et offre des fonctionnalités avancées pour la représentation visuelle et l'exploration de données tridimensionnelles. C'est une solution puissante pour les applications qui nécessitent des visualisations 3D dans un environnement web.



FIGURE 8.12 – Logo de Vtk JS

Firebase

Firebase est une plateforme de développement d'applications mobiles et web proposée par Google. Elle offre des outils complets pour la gestion de bases de données en temps réel, l'authentification des utilisateurs, l'hébergement de sites web et bien plus encore. Firebase permet aux développeurs de créer rapidement des applications robustes sans se soucier de l'infrastructure sous-jacente. C'est un outil populaire pour le développement d'applications interactives et évolutives. On a utilisé Firebase pour intégrer l'authentification à notre application web.



FIGURE 8.13 – Logo de Firebase

Vercel

Vercel est une plateforme cloud pour le déploiement rapide et la mise en ligne d'applications web. Elle facilite le déploiement, l'hébergement et la gestion d'applications front-end et full-stack. Vercel offre des fonctionnalités telles que le déploiement automatisé depuis des référentiels Git, des prévisualisations pour les push-requests, la gestion de domaines personnalisés et des performances optimisées grâce à la mise en cache mondiale des contenus. C'est un choix populaire pour les développeurs cherchant à déployer rapidement des applications modernes et performantes en ligne. On a utilisé Vercel pour le déploiement du front end.



FIGURE 8.14 – Logo de Vercel

8.3 Back End

Pytorch

Pytorch est un Framework Python de calcul scientifique basé sur Torch, et développé par Meta. Pytorch a été utilisé dans le développement du modèle MeshSegNet. Il a deux objectifs principaux :

- Un remplacement de NumPy pour utiliser la puissance des GPU et d'autres accélérateurs.
- Une bibliothèque de différenciation automatique Autograd qui est utile pour entraîner des réseaux de neurones.



FIGURE 8.15 – Logo de Pytorch

FastAPI

FastAPI est un framework Web moderne et hautes performances conçu pour construire des applications API en utilisant Python. Il offre une syntaxe asynchrone qui permet une exécution rapide et efficace des requêtes. Grâce à son intégration native avec des bibliothèques telles que Pydantic pour la validation des données et Swagger UI pour la génération automatique de la documentation, FastAPI facilite le développement d'API robustes, bien documentées et faciles à maintenir. Grâce à sa vitesse et à son évolutivité, FastAPI est souvent préféré pour les applications où les performances sont essentielles, comme les services Web à fort trafic, les microservices etc...



FIGURE 8.16 – Logo de FastAPI

Postman

Postman est un outil populaire utilisé par les développeurs pour tester, documenter et collaborer sur les API (Interfaces de Programmation Applicative). Il fournit une interface conviviale qui permet aux développeurs de créer, envoyer et gérer des requêtes HTTP vers des services web, ainsi que de visualiser et d'analyser les réponses reçues.



FIGURE 8.17 – Logo de Postman

Paraview

Paraview est une application de visualisation et d'analyse de données scientifiques, principalement utilisée pour visualiser des données en 3D provenant de simulations numériques, de modèles scientifiques et de données expérimentales. Elle est couramment utilisée dans les domaines de la physique, de l'ingénierie, de la géophysique et d'autres domaines scientifiques pour comprendre visuellement et analyser des données complexes.



FIGURE 8.18 – Logo de Paraview

Docker

Docker est une plateforme open-source pour la virtualisation légère de conteneurs. Elle permet d'emballer des applications et leurs dépendances dans des conteneurs isolés, garantissant la portabilité et la cohérence lors du déploiement sur différentes plateformes. Docker simplifie la gestion des environnements de développement, des tests et du déploiement en fournissant une solution efficace pour encapsuler des applications dans des unités autonomes. On a utilisé Docker pour déployer une image Docker à AWS ECR pour construire une fonction Lambda.

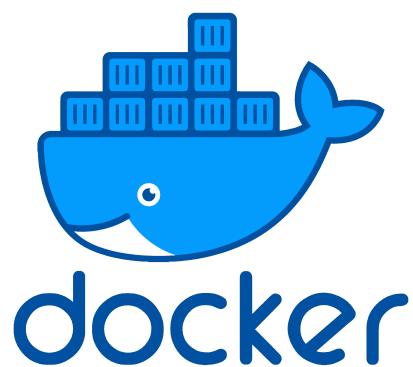


FIGURE 8.19 – Logo de Docker

AWS Cloud

Amazon Web Services (AWS) est la principale plateforme de services cloud au monde, proposée par Amazon. Elle offre une vaste gamme de services informatiques, de stockage, de bases de données, d'analyse, de machine learning, de sécurité et bien plus encore. Grâce à AWS, les entreprises peuvent facilement provisionner des ressources informatiques à la demande, évoluer rapidement en fonction de leurs besoins et bénéficier d'une infrastructure cloud hautement évolutive et sécurisée. Plus précisément, on a utilisé 3 services d'AWS :

- **Amazon Elastic Container Registry (Amazon ECR)** est un registre de conteneurs entièrement géré qui offre un hébergement haute performance, afin que vous puissiez déployer de manière fiable des images d'applications et des artefacts n'importe où.
- **AWS Lambda** est un service de calcul d'événement sans serveur qui vous permet d'exécuter du code pour presque tout type d'application ou de service de backend, sans vous soucier de l'allocation ou de la gestion des serveurs. Vous pouvez activer Lambda pour plus de 200 services AWS et applications SaaS (logiciel en tant que service). En plus, vous ne payez que pour ce que vous utilisez.
- **Amazon API Gateway** est un service AWS permettant de créer, de publier, de maintenir, de surveiller et de sécuriser les API REST, HTTP et WebSocket à n'importe quelle échelle. Les développeurs d'API peuvent créer des API qui accèdent à AWS ou à d'autres services Web, ainsi qu'aux données stockées dans le cloud AWS. En tant que développeur d'API API Gateway, vous pouvez créer des API à utiliser dans vos propres applications client. Vous pouvez également mettre vos API à la disposition des développeurs d'applications tiers.



FIGURE 8.20 – Logo de AWS

Quatrième partie

Réalisation

Back End

Dans cet premier chapitre on va présenter notre back end ; l'API créer en utilisant FastAPI, les routes et les méthodes disponibles, puis on va le tester localement et on va expliquer les étapes de déploiement sur le cloud d'AWS.

9.1 Architecture du Back End

Voici les fichiers et dossiers que contient notre Back End :

- model/ : Stocke les modèles MeshSegNet pré-entraînés.
- temp/ : Stockage temporaire des fichiers pendant le traitement.
- output/ : Stocke les fichiers générés par le modèle pour les réponses API.
- config.py : Configuration et chargement du modèle.
- helper.py : Fonctions utilitaires pour les API.
- main.py : Définit les points de terminaison de l'API.
- meshsegnet.py : Encapsule l'architecture et les méthodes de MeshSegNet.
- model.py : Contient les fonctions *predict* et *predict_alpha* pour la segmentation 3D.
- setup.py : Installe les packages requis et configure l'espace de travail.
- install-requirements.ps1 : Configure le projet sous Windows.
- install-requirements.sh : Configure le projet sur Linux/macOS.
- Dockerfile : Image Docker utilisé pour le déploiement sur le Cloud.
- Requirements.txt : Contient les dépendances pour le fonctionnement backend.
- README : Fournit des informations sur l'installation, la configuration et le projet.
- venv : Environnement isolé pour les projets Python.
- Licence : Projet sous licence MIT.

The screenshot shows a GitHub Copilot interface with a dark theme. On the left, there's a sidebar showing the project structure:

```

model
  MeshSegNet_Max_15_classes_7...
  Mesh_Segmentation_MeshSeg...
.gitignore
Dockerfile
LICENSE
README.md
config.py
helpers.py
install-requirements.ps1
install-requirements.sh
main.py
meshsegnet.py
model.py
requirements.txt
setup.py

```

The main area displays the `main.py` file content:

```

1  from helpers import create_temp_file, delete_temp_file
2  from fastapi import FastAPI, UploadFile, File
3  from model import predict, predict_alpha
4  from config import AppConfig
5
6  import json
7  import vedo
8  import os
9
10
11  app = FastAPI()
12
13  # Configure the application using AppConfig
14  config = AppConfig(app)
15
16  @app.get("/")
17  def read_root():
18      return {"message": "TeethSeg MeshSegNet API by 3DSF Interns!"}
19
20  @app.post("/api/v1/predict")
21  async def predict_and_send(file: UploadFile = File(...)):
22
23      if file.filename == "":
24          return {
25              "message": "File Required"
26      }

```

FIGURE 9.1 – Structure du dossier du Back End

Notre REST API a 3 routes possibles :

- **GET "/"** : Donne des informations générales sur l'API et les routes disponibles.
- **POST "/api/v1/predict"** : Accepte comme entrée un fichier d'extension .obj et retourne un fichier d'extension .vtp et son nom. Ce route API fait la segmentation du fichier d'entrée .obj en utilisant MeshSegNet sans post processing et retourne un fichier .vtp segmenter. (Cela ne donnera pas un bon résultat)
- **POST "/api/v1/predict/post_processing"** : Accepte comme entrée un fichier d'extension .obj et retourne un fichier d'extension .vtp et son nom. Ce route API fait la segmentation du fichier d'entrée .obj en utilisant MeshSegNet avec post processing et retourne un fichier .vtp segmenter. (Cela donnera un résultat plus précis)

Exécutons notre serveur FastpApi en direct localement :

```

hamza@DESKTOP-SDAQCQ5 MINGW64 /d/Bureau/3DSF/backend/SegTeethApi (aws_lambda)
$ source venv/Scripts/activate
(venv)
hamza@DESKTOP-SDAQCQ5 MINGW64 /d/Bureau/3DSF/backend/SegTeethApi (aws_lambda)
$ uvicorn main:app --reload
←[32mINFO←[0m:     Will watch for changes in these directories: ['D:\\Bureau\\3DSF\\backend\\SegTeethApi']
←[32mINFO←[0m:     Uvicorn running on ←[1mhttp://127.0.0.1:8000←[0m (Press CTRL+C to quit)
←[32mINFO←[0m:     Started reloader process [←[36m←[1m17384←[0m] using ←[36m←[1mWatchFiles←[0m
INFO:     Started server process [2600]
INFO:     Waiting for application startup.
INFO:     Application startup complete.

```

FIGURE 9.2 – Démarrage du serveur local FastAPI sur `http://127.0.0.1:8000`

Testons maintenant notre API en utilisant Postman :

The screenshot shows a Postman interface with a GET request to 'localhost:8000'. The 'Body' tab is selected, showing the response body:

```

1 "message": "TeethSeg MeshSegNet API by 3DSF Interns! Routes possible: '/' General Informations about API! '/api/v1/predict/' Segmentation without post processing. '/api/v1/predict/post_processing' Segmentation with post-processing."

```

FIGURE 9.3 – Test du root route

Testons la route de segmentation avec post processing :

The screenshot shows a POST request to 'localhost:8000/api/v1/predict/post_processing'. The 'Body' tab is selected, showing the request body:

Key	Value	Description	Bulk Edit
file	Sample_7.obj		
Key	Value	Description	

The response status is 200 OK with the following JSON body:

```

1 {
2   "statusCode": 200,
3   "headers": {
4     "Content-Type": "application/json"
5   },
6   "body": {
7     "filename": "\\"Sample_7_d_predicted_refined.vtp\\", "prediction_file": "<?xml version='1.0'?\>\n<VTKFile type='PolyData' version='0.\n1'>\n<byte_order='LittleEndian'>\n<header_type='UInt32'>\n<compressor='vtkZlibDataCompressor'>\n<PolyData>\n<Piece\nNumberofPoints='5074'\nNumberofVerts='0'\nNumberofLines='0'\nNumberofPolys='9999'\n>\n<PointData>\n<Normals>\n<DataArray\nname='Normals'\nformat='appended'\nRangeMin='0.9999999307711111'\nRangeMax='1.\n000000979'\noffset='0'\n/>\n<PointData>\n<CellData>\n<Normals>\n<DataArray\nname='Normals'\nformat='appended'\nRangeMin='0.9999998853411111'\nRangeMax='1.'\nNumberofComponents='3'\n>\n</CellData>\n</Normals>\n</DataArray>\n</Normals>\n</PointData>\n</Piece>\n</PolyData>\n</VTKFile>\n"
}

```

FIGURE 9.4 – Test du route de segmentation avec post processing

Dans la réponse json qu'on reçoit on peut voir le nom du fichier de sortie vtp dans le champ body.filename et le contenu de ce fichier sous format xml dans le champ body.predictionfile . Passons maintenant à la visualisation du fichier générer avec le logiciel Paraview qui est une application logicielle open-source largement utilisée pour la visualisation et l'analyse de données en 3D.

9.2 Visualisation du fichier VTP générer avec Paraview

9.2.1 Sans post-processing

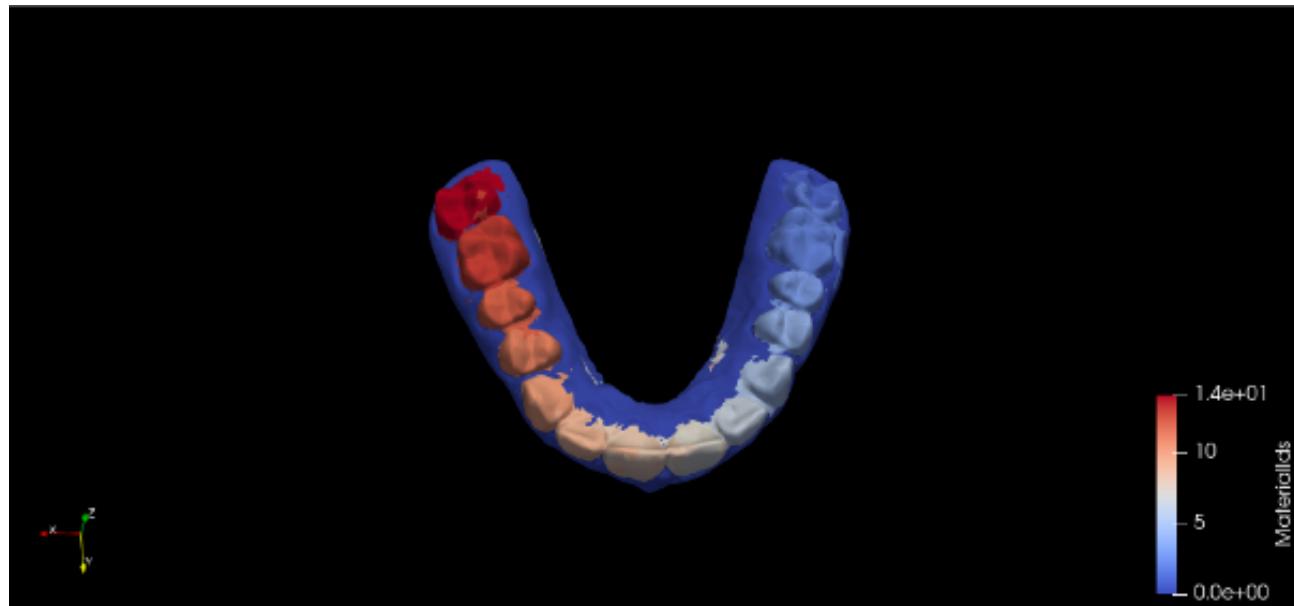


FIGURE 9.5 – Résultat de la segmentation sans post-processing

9.2.2 Avec post-processing

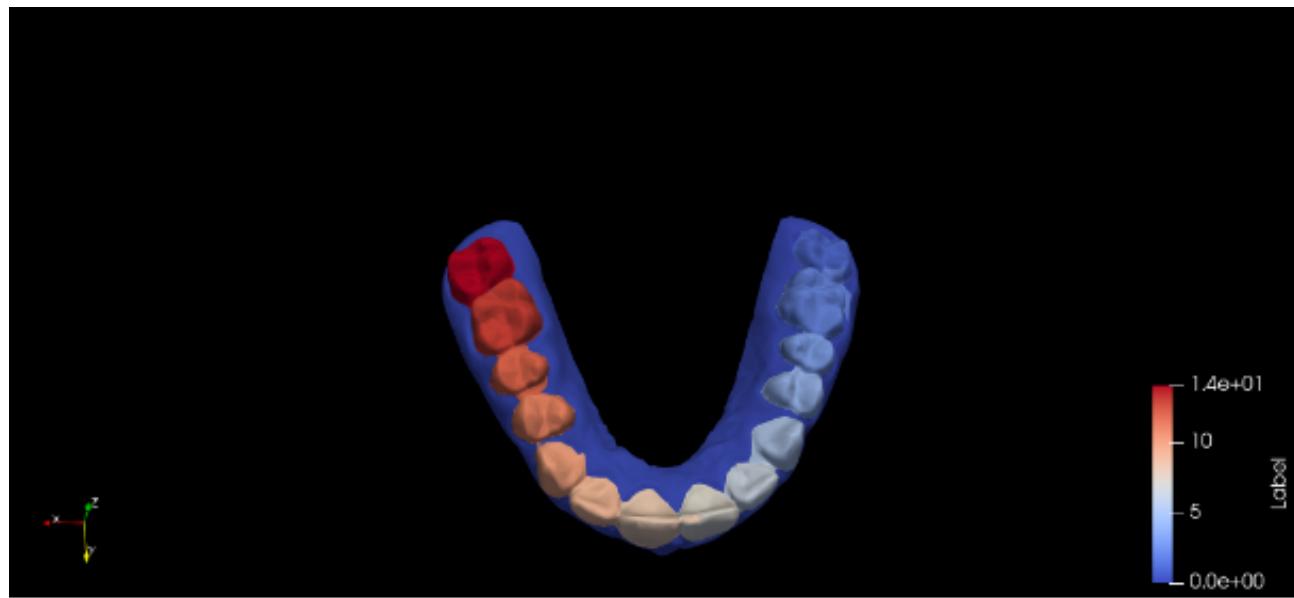


FIGURE 9.6 – Résultat de la segmentation avec post-processing

Comme on peut constater, l'ajout du post processing donne un résultat de segmentation plus précis. Le post processing est mis en place grâce à la fonction **`cut_from_graph(edges, unaries, pairwise)`** de la librairie Pygco.

```

279     refine_labels = cut_from_graph(edges, unaries, pairwise)
280     refine_labels = refine_labels.reshape([-1, 1])
281
282     # output refined result
283     mesh3 = mesh_d.clone()
284     mesh3.celldata['Label'] = refine_labels
285
286     # vedo.write(mesh3, os.path.join(output_path, '{}_d_predicted_refined.vtp'.format(i_sample[:-4])))
287     output_file_name_d_refined = '{}_d_predicted_refined'.format(filename[:-4])
288     vedo.write(mesh3, os.path.join(output_path, output_file_name_d_refined + ".vtp"))
289
290     # upsampling
291     print('\tUpsampling...')
292     if mesh.ncells > 50000:
293         target_num = 50000 # set max number of cells
294         ratio = target_num/mesh.ncells # calculate ratio
295         mesh.decimate(fraction=ratio)
296         print('Original contains too many cells, simplify to {} cells'.format(mesh.ncells))
297

```

FIGURE 9.7 – Post processing dans le code

9.3 Alternative à Paraview

Pendant le développement de notre application web TeethSeg, nous avions besoin d'un outil pour la visualisation et le test de nos API directement depuis le navigateur. Ainsi, nous avons développé une petite application en JavaScript pur pour tester nos API de segmentation (avec ou sans post-traitement), plutôt que d'utiliser Paraview.

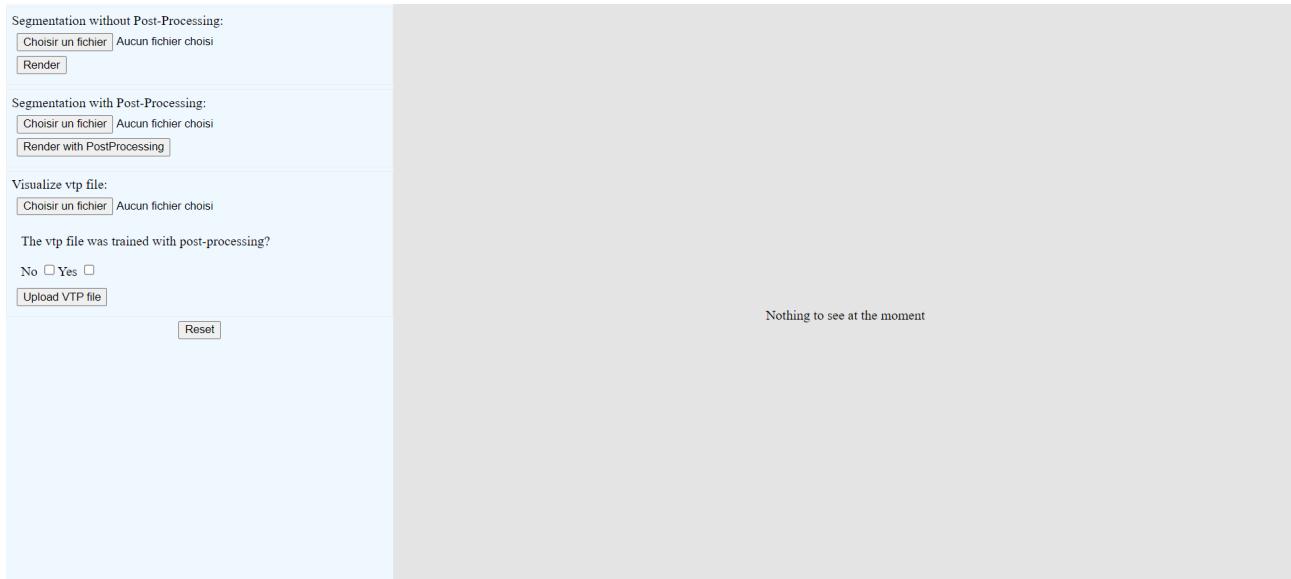


FIGURE 9.8 – Notre outil de visualisation avant TeethSeg



FIGURE 9.9 – Visualisation du fichier vtp segmenter sans post-processing



FIGURE 9.10 – Visualisation du fichier vtp segmenter avec post-processing

9.4 Deploiement

Pour le déploiement, on a proposé deux options : soit le déploiement local sur l'un des serveurs de la société, soit sur l'un des services du Cloud comme Render, Google Cloud, Azure et AWS.

Pour le déploiement sur le cloud, on a considéré deux options, la première est Render et la deuxième AWS. Pour Render, on a effectué le déploiement qui a été plutôt simple à réaliser mais le coût énorme qui fallait pour mettre le back end en marche à freiner notre progrès. Notre back end a besoin de minimum 4gb de RAM pour réaliser la segmentation. Comparons maintenant les coûts des deux plateformes, celui de Render contre celui d'AWS.

```

START RequestId: 5f048932-7e7f-43af-9485-98a904ac1ea1 Version: $LATEST
Predicting Sample filename: Sample_7.obj
Downsampling...
Predicting...
Refining by pygco...
Upsampling...
Sample filename: Sample_7.obj completed
Computing time: 15.34 sec
END RequestId: 5f048932-7e7f-43af-9485-98a904ac1ea1
REPORT RequestId: 5f048932-7e7f-43af-9485-98a904ac1ea1 Duration: 15420.11 ms Billed Duration: 15421 ms Memory Size: 4000 MB Max Memory Used: 3965 MB

```

FIGURE 9.11 – Utilisation de 4GB RAM pour une segmentation

The screenshot shows the Render pricing page. At the top, it says "From \$0 USD / month". It lists several service features with checkmarks: "Web services with HTTP/2 and full TLS", "Private services", "Background workers", "Node, Python, Go, Rust, Ruby, and Elixir", "Custom Docker containers", and "SSD disks for \$0.25/GB per month". Below this is a table with columns for "Instance Type", "Pricing", "RAM", and "CPU". The table has five rows: "Free" (\$0/month with limits, 512 MB, 0.1 CPU), "Starter" (\$7/month, 512 MB, 0.5 CPU), "Standard" (\$25/month, 2 GB, 1 CPU), and "Pro" (\$85/month, 4 GB, 2 CPU).

Instance Type	Pricing	RAM	CPU
Free	\$0/month <small>with limits</small>	512 MB	0.1
Starter	\$7/month	512 MB	0.5
Standard	\$25/month	2 GB	1
Pro	\$85/month	4 GB	2

FIGURE 9.12 – Prix de 4 Gb RAM dans Render est 85\$/mois



FIGURE 9.13 – Prix de 4 Gb RAM dans AWS est 1.3\$ pour 1000 segmentations

On a donc opté pour le choix de déploiement sur la plateforme AWS. Voici donc l'architecture du Back End adopté sur le cloud AWS. On a utilisé trois services : **ECR**, **LAMBDA** et **API GATEWAY**.

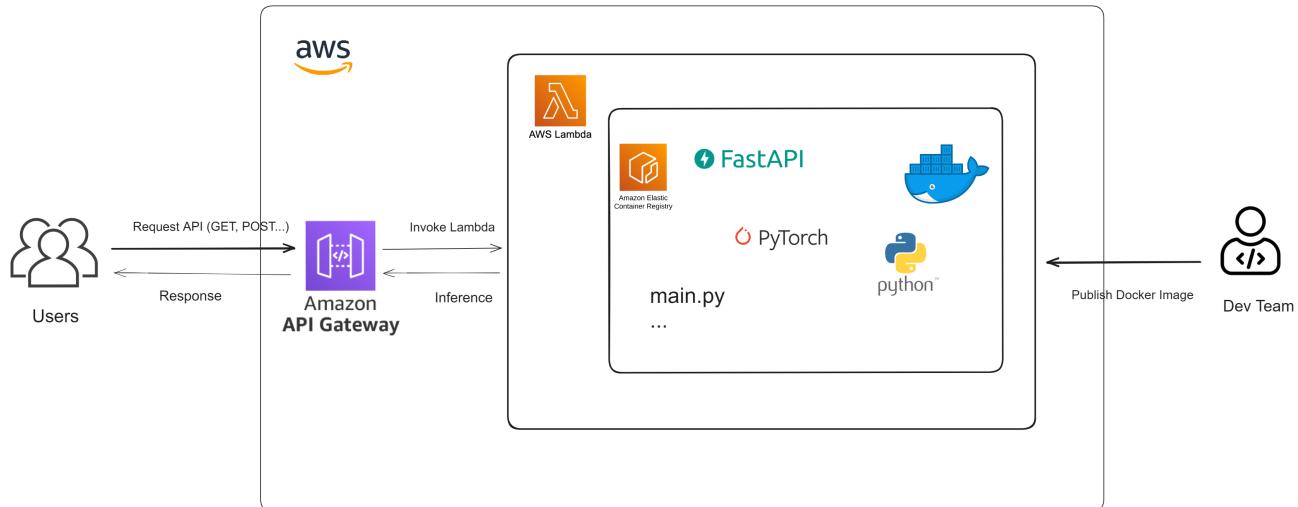


FIGURE 9.14 – Architecture sur AWS Cloud

Pour le déploiement sur AWS, nous avons suivi ces étapes :

1. Création d'un utilisateur IAM avec ces autorisations :

The screenshot shows the 'Permissions' tab in the AWS IAM console. Under 'Permissions policies (3)', three AWS managed policies are listed:

Policy name	Type	Attached via
AmazonEC2ContainerRegistryFullAccess	AWS managed	Directly
AWSLambda_FullAccess	AWS managed	Directly
IAMFullAccess	AWS managed	Directly

FIGURE 9.15 – Crédit d'un utilisateur IAM dans le compte AWS racine

2. Connection à l'utilisateur IAM, puis création d'un référentiel ECR et l'upload de l'image Docker depuis le Dockerfile vers le référentiel ECR nouvellement créé.

The screenshot shows the 'Repositories' tab in the AWS ECR console. The 'teethsegapi' repository is selected, displaying one image entry:

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
latest	image	27 août 2023, 00:06:06 (UTC+01)	5836.95	Copy URI	sha256:f229022894573aa38112e8dc4122b...	-	-

FIGURE 9.16 – Upload de l'image Docker au référentiel ECR

3. Crédit d'une nouvelle fonction Lambda à l'aide de l'image dans ECR.

The screenshot shows the 'Function overview' tab in the AWS Lambda console for the 'teethsegapi' function. It displays the following details:

- Function name:** teethsegapi
- Description:** -
- Last modified:** 2 days ago
- Function ARN:** arn:aws:lambda:us-east-1:099072753431:function:teethsegapi
- Function URL:** <https://zts5x7dqs4nujbg4hzrrotvw40cxlv.lambda-url.us-east-1.on.aws/>

The 'Image' tab is selected, showing the deployment information:

- Image URI:** 099072753431.dkr.ecr.us-east-1.amazonaws.com/teethsegapi@sha256:f229022894573aa38112e8dc4122b6e562aa22be669fc6504eb450e4142c5a9d
- Architecture:** x86_64

FIGURE 9.17 – La fonction Lambda contenant l'API

4. Configuration de la fonction Lambda en augmentant la mémoire allouée à 4096 Mo (4 Go) et du délai d'attente à 1 minute (normalement, notre prédiction est effectuée dans les 15/20 secondes et le l'utilisation Maximale de la mémoire est de 4 Go).

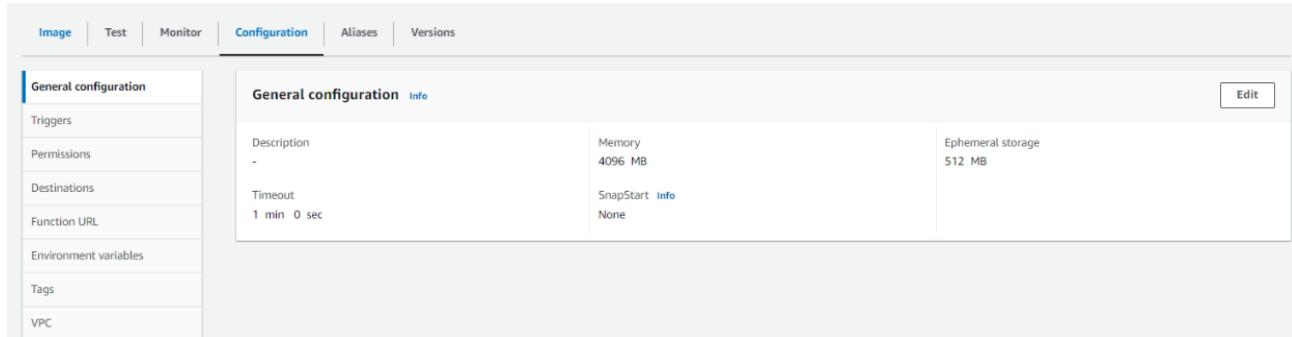


FIGURE 9.18 – Configuration de la fonction Lambda

5. Ajout du service API Gateway que nous allons utiliser pour communiquer avec notre fonction Lambda.

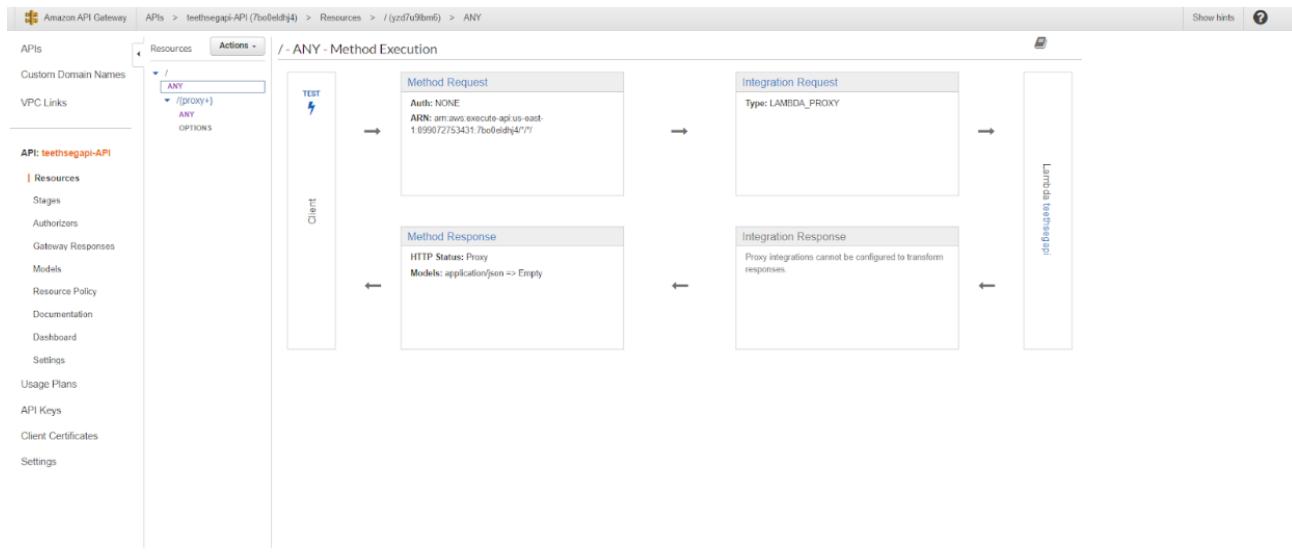


FIGURE 9.19 – Ajout du API Gateway

6. Déploiement de l'API Gateway et génération du URL de l'API.

prod Stage Editor

Invoke URL: <https://7bo0eldhj4.execute-api.us-east-1.amazonaws.com/prod>

[Delete Stage](#) [Configure Tags](#)

Settings Logs/Tracing Stage Variables SDK Generation Export Deployment History Documentation History Canary

Cache Settings

Enable API cache

Default Method Throttling

Choose the default throttling level for the methods in this stage. Each method in this stage will respect these rate and burst settings. Your current account level throttling rate is 10000 requests per second with a burst of 5000 requests. [Read more about API Gateway throttling](#)

Enable throttling ⓘ

Rate: 10000 requests per second

Burst: 5000 requests

Web Application Firewall (WAF) [Learn more](#).

Select the Web ACL to be applied to this stage.

Web ACL: None [Create Web ACL](#)

Client Certificate

Select the client certificate that API Gateway will use to call your integration endpoints in this stage.

Certificate: None

[Save Changes](#)

FIGURE 9.20 – Deploiement de l'API publiquement

7. Testons maintenant notre url publique de l'API. Faisons une requête GET au route "/".

https://7bo0eldhj4.execute-api.us-east-1.amazonaws.com/prod/

GET https://7bo0eldhj4.execute-api.us-east-1.amazonaws.com/prod/

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

None form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

Body Cookies Headers (12) Test Results

Pretty Raw Preview Visualize JSON

Status: 200 OK Time: 6.84 s Size: 683 B Save as Example

```

1   "message": "TeethSeg MeshSegNet API by 3DSF Interns! Routee possible: '/api/v1/predict/post_processing' Prediction with post-processing."
2
3
  
```

FIGURE 9.21 – Test du route "/"

8. Testons une requête POST au route "/api/v1/predict/post-processing" avec comme entrée un fichier de scan intra-dentaire d'extension .obj

POST https://7bo0eldh4.execute-api.us-east-1.amazonaws.com/prod/api/v1/predict/post_processing

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Body (1) none form-data x-www-form-urlencoded raw binary GraphQL

Key	Value	Description
<input checked="" type="checkbox"/> file	Sample_4.obj	
Key	Value	Description

Body Cookies Headers (12) Test Results

Status: 200 OK Time: 24.50 s Size: 380.89 KB Save as Example

Pretty Raw Preview Visualize JSON

```
1
2   "statusCode": 200,
3   "headers": {
4     "Content-Type": "application/json"
5   },
6   "body": "{\"filename\": \"Sample_4_d_predicted_refined.vtp\", \"prediction_file\": \"<?xml version=\\\\\"1.0\\\\\"?>\\n<VTKFile type=\\\\\"PolyData\\\\\" version=\\\\\"0.1\\\\\" byte_order=\\\\\"LittleEndian\\\\\" header_type=\\\\\"UInt32\\\\\" compressor=\\\\\"vtkZLibDataCompressor\\\\\">\\n<Piece NumberofPoints=\\\\\"5071\\\\\" NumberofVerts=\\\\\"0\\\\\" NumberofLines=\\\\\"0\\\\\" NumberofFPoints=\\\\\"0\\\\\" NumberofFStrips=\\\\\"0\\\\\" NumberofPolys=\\\\\"9999\\\\\">\\n<PointData Normals=\\\\\"Normals\\\\\">\\n<DataArray RangeMin=\\\\\"0.99999992592\\\\\" RangeMax=\\\\\"1.0000000956\\\\\" type=\\\\\"Float32\\\\\" Name=\\\\\"Normals\\\\\" NumberofComponents=\\\\\"3\\\\\" format=\\\\\"appended\\\\\" offset=\\\\\"0\\\\\">\\n</PointData>\\n<CellData Normals=\\\\\"Normals\\\\\">\\n<DataArray RangeMin=\\\\\"0.999999999179\\\\\" RangeMax=\\\\\"1.00000000182\\\\\" type=\\\\\"Float32\\\\\" Name=\\\\\"Normals\\\\\" NumberofComponents=\\\\\"3\\\\\" format=\\\\\"appended\\\\\" offset=\\\\\"74866\\\\\">\\n<DataArray type=\\\\\"Int32\\\\\" Name=\\\\\"Label\\\\\" format=\\\\\"appended\\\\\">\\n
```

FIGURE 9.22 – Test du route ”/api/v1/predict/post_processing”

CHAPITRE 10

Front End

Dans cette section on va présenter notre front end, les étapes qui ont mené à sa création en commençant par l'étape du design et prototypage sur Figma, à son développement avec Vite et React et enfin à son déploiement avec Vercel.

10.1 Design Figma

La partie front end a été notre première tâche pendant cette période de stage, comme ressources de travail on n'avait qu'une ancienne maquette de site web réalisé par un ancien stagiaire PFE, qu'on a du re-désigner avec les composants de Tailwindcss qu'on a décidé d'utiliser dans notre application web. Voici le design sur Figma de la page "Start" qui est censé être la page principale qui va contenir les principales fonctionnalités de notre site web.

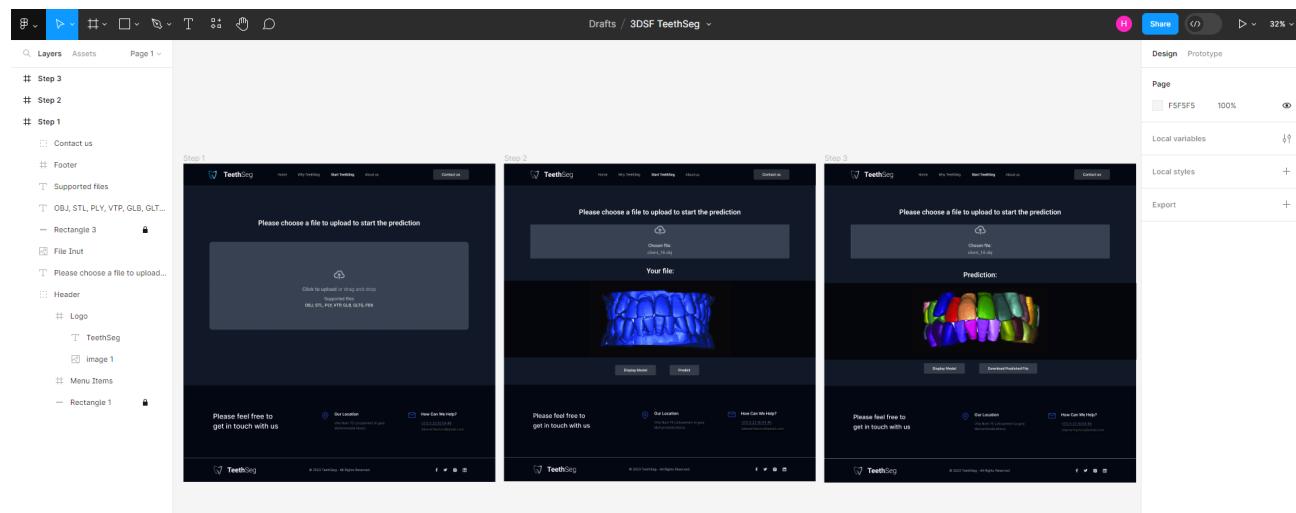


FIGURE 10.1 – Design initiale sur Figma

Ensuite, au fur et à mesure de notre avancement dans le projet, et la compréhension de nouveaux concepts dans la visualisation 3D sur le web avec ThreeJs et Vtk Js on a modifié et adapté le look de notre application web pour être en harmonie avec ces librairies et leur style.

10.2 Structure du Front End

- node_modules/ : Répertoire des modules externes du projet.
- public/ : Dossier contenant des fichiers statiques accessibles par navigateur.
- → superviseurs/ : Contient des images de superviseurs.
- → team/ : Contient des images des contributeurs de notre groupe.
- src/ : Le répertoire source principal de notre application.
- → assets/ : Les ressources statiques de votre application ; images et polices.
- → → tech/ : Contient des images liées aux technologies utilisées.
- → composants/ : Les composants réutilisables de notre application.
- → → shared/ : Composants partagés dans toute l'application.
- → → ui/ : Composants UI spécifiques à notre application.
- → config/ : Fichiers de configuration de l'application.
- → data/ : Données statiques ou fichiers JSON utilisés dans notre application.
- → helpers/ : Fonctions utilitaires utilisées dans diverses parties de l'application.
- → lib/ : Bibliothèques ou modules personnalisés utilisés dans l'application.
- → routes/ : Fichiers représentant la structure de routage entre les pages.
- → → services/ : Services organisés en fonction des différentes sections de l'application.
- → → about/ : Sous-dossiers correspondant à la section à propos.
- → → contact/ : Sous-dossiers correspondant à la section contact.
- → → footer/ : Sous-dossiers correspondant à la partie pied de page des pages.
- → → header/ : Sous-dossiers correspondant à la partie en-tête des pages.
- → → main/ : Sous-dossiers correspondant à la fonctionnalité principale.
- → styles/ : Fichiers CSS et préprocesseurs pour le style.
- .gitignore : Les fichiers et dossiers a ignoré par Git.
- .env : Stocke les variables d'environnement.
- Licence : Projet sous licence MIT.

10.3 Architecture du Front End

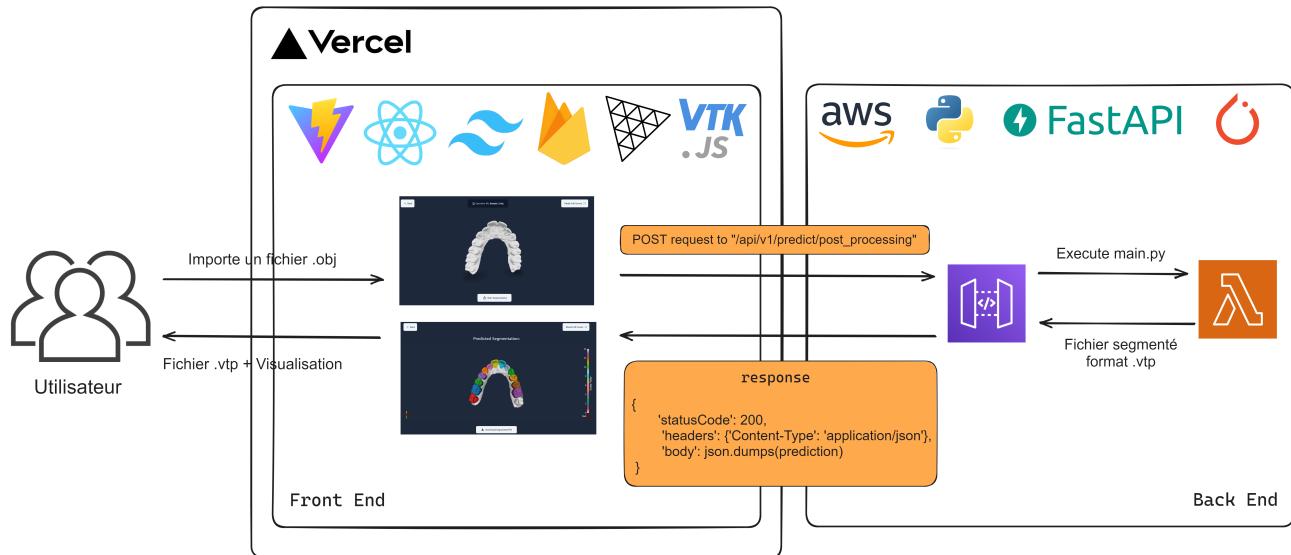


FIGURE 10.2 – Architecture du Front End et sa communication avec le Back End

10.4 Présentation des interfaces de l'application

L'application offre deux fonctionnalités principales :

- 1. Segmentation OBJ :** Segmentation d'un fichier OBJ, en cliquant sur "Segmenter le fichier OBJ", puis sur "Démarrer la segmentation" pour segmenter le fichier. Visualisation et téléchargement du fichier résultat de la segmentation au format VTP.
- 2. Visualisation VTP :** Visualisation d'un fichier VTP pré-segmenter en cliquant sur "Visualiser le fichier VTP".

La conception des interfaces de l'application est une étape très importante puisque toutes les interactions avec le cœur de l'application passent à travers ces interfaces, on doit alors guider l'utilisateur de la meilleure des manières. Cette partie présentant un scénario d'utilisation de l'application par quelques interfaces, peut être considérée comme un guide d'utilisation de l'application.

10.4.1 Page d'accueil

La page d'accueil contient des informations sur la nature du site web, une vidéo démonstrative ainsi qu'un formulaire de contact qui renvoie les messages des utilisateurs à l'email de la société. Le site web est disponible dans les deux modes : dark et light.

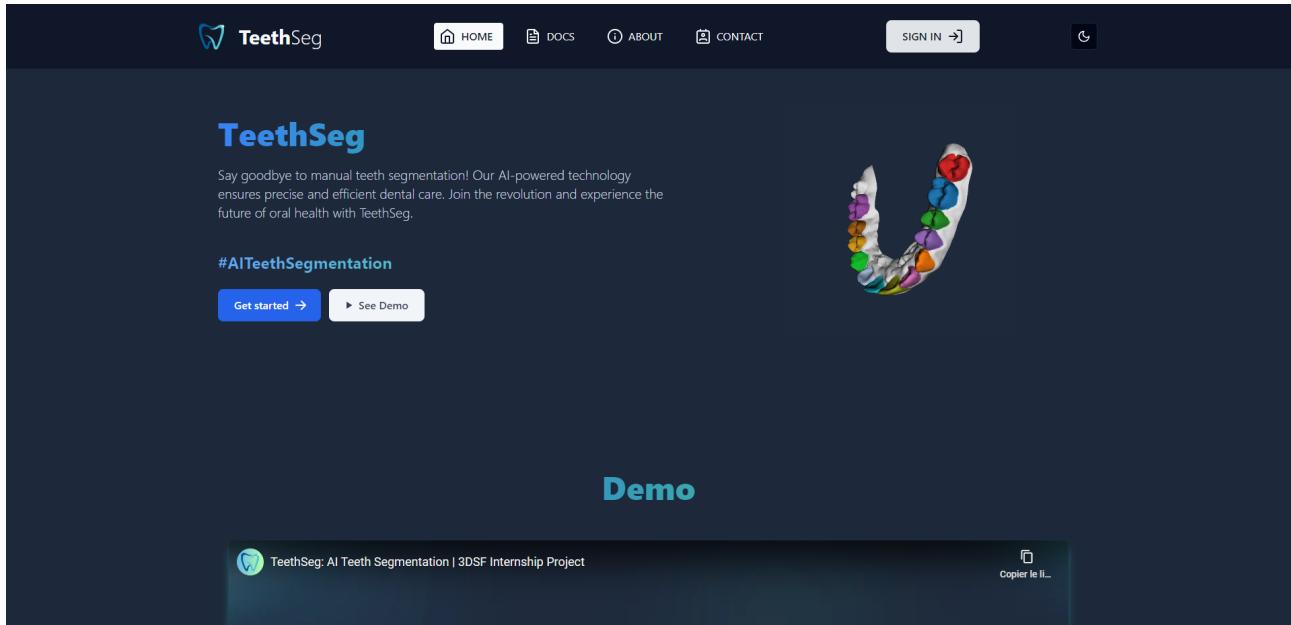


FIGURE 10.3 – Hero de la page d'accueil TeethSeg mode Dark

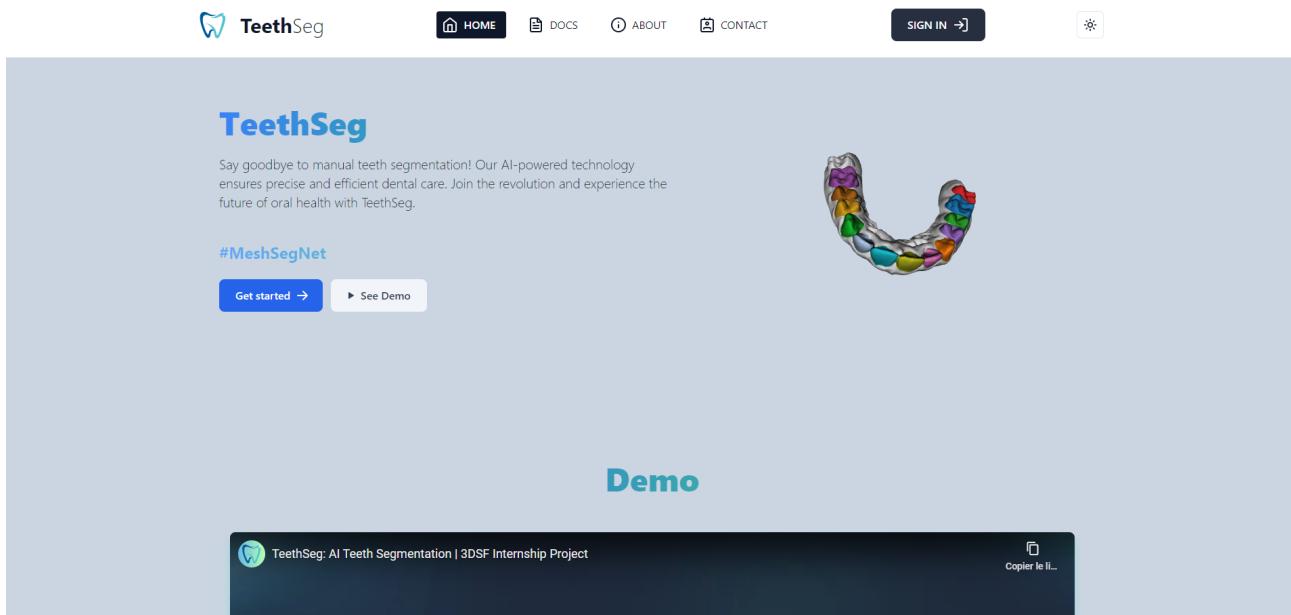


FIGURE 10.4 – Hero de la page d'accueil TeethSeg mode Light

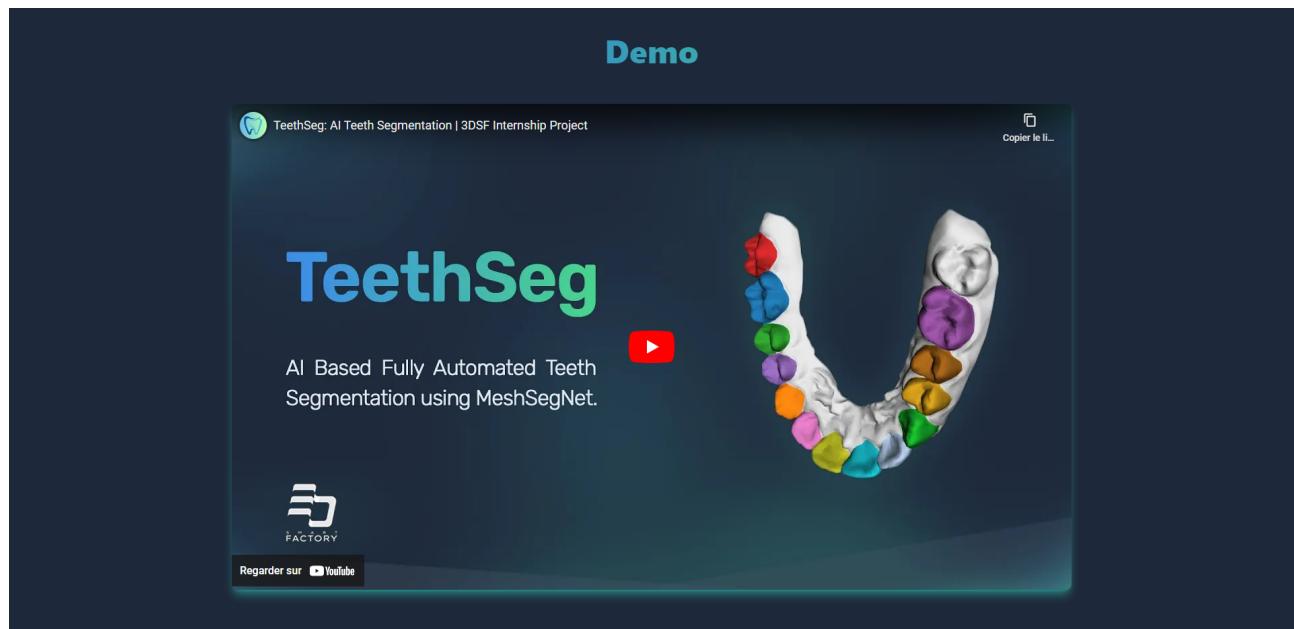


FIGURE 10.5 – Video Demo de l'utilisation de l'application

The screenshot shows the TeethSeg landing page. At the top center is the heading "Why TeethSeg". Below it is a paragraph about the importance of 3D tooth segmentation in digital orthodontics. Three circular icons represent features: "Best AI Model" (blue), "Fast Response" (yellow), and "Easy & Simple UI" (green). Below each icon is a brief description. At the bottom is a section titled "FAQs" with the heading "Any Questions ?" and two expandable FAQ cards: "What is TeethSeg?" and "What technologies power TeethSeg's front end?".

FIGURE 10.6 – Fonctionnalités et FAQ

The screenshot shows a dark-themed contact form titled "Contact Us". It includes fields for "Your email" (name@teethseg.com), "Subject" (Let us know how we can help you), and "Your message" (Leave a comment...). A blue "Send Message" button is at the bottom. Below the form, the TeethSeg logo is on the left, followed by copyright information: "© 2023 TeethSeg - All Rights Reserved by 3DSF Interns". On the right are social media icons for Facebook, Instagram, LinkedIn, and Twitter.

FIGURE 10.7 – Formulaire de Contact et Footer

10.4.2 Page Documentation

Cette page contient la documentation sur le back et front end du projet : description, structure, méthode d'installation, de configuration et d'usage etc...

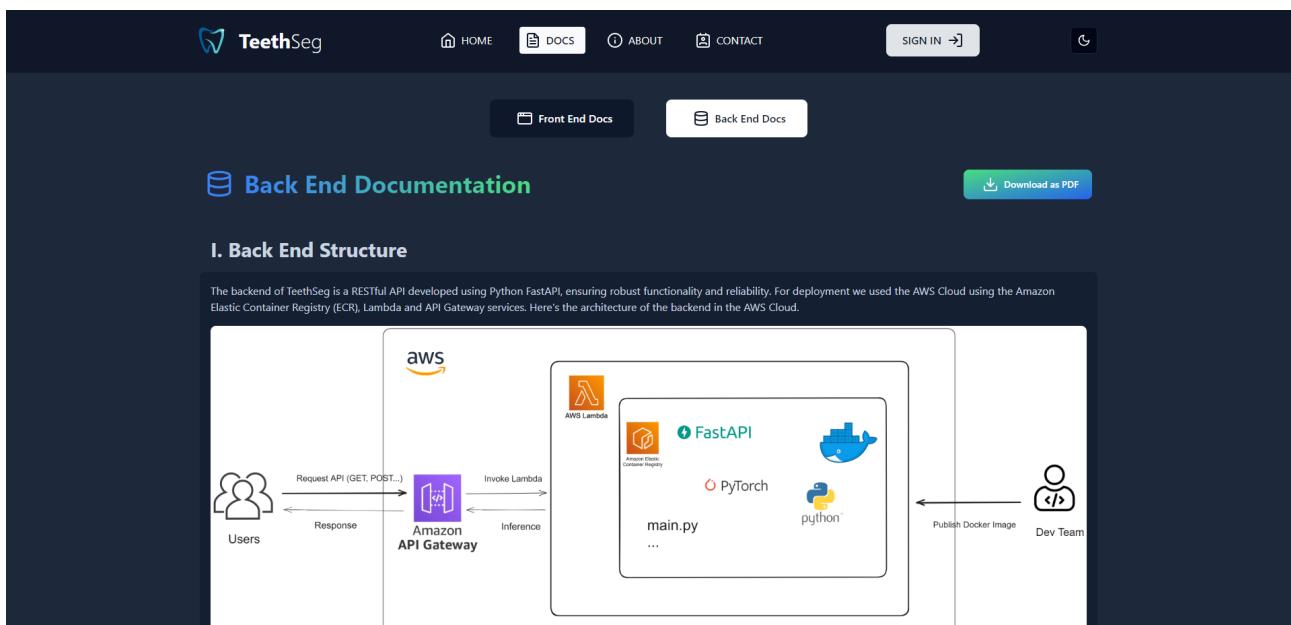


FIGURE 10.8 – Page de la documentation du projet

10.4.3 Page à propos

Cette page contient les membres de l'équipe qui ont travaillé sur ce projet : les noms, écoles, filière ainsi que les liens Github et LinkedIn de chacun. En bas, on trouve aussi des informations sur les encadrants, et les technologies avec lesquelles on a réalisé cette application web.

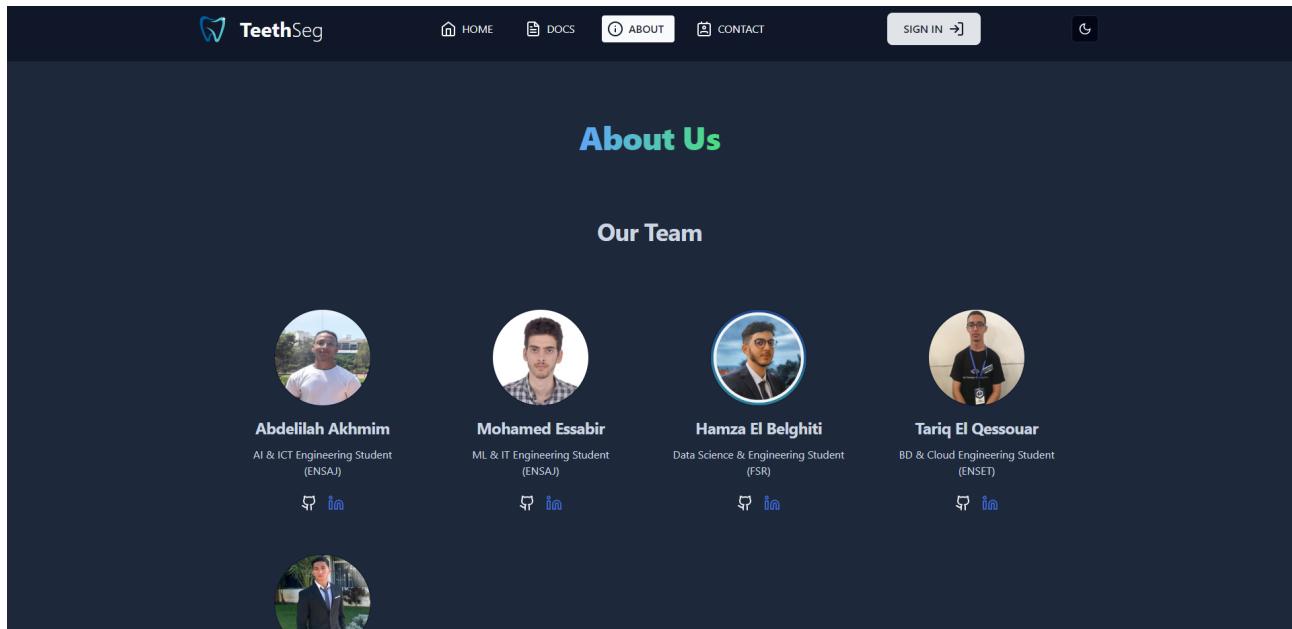


FIGURE 10.9 – Page à propos (Equipe de stagiaires)

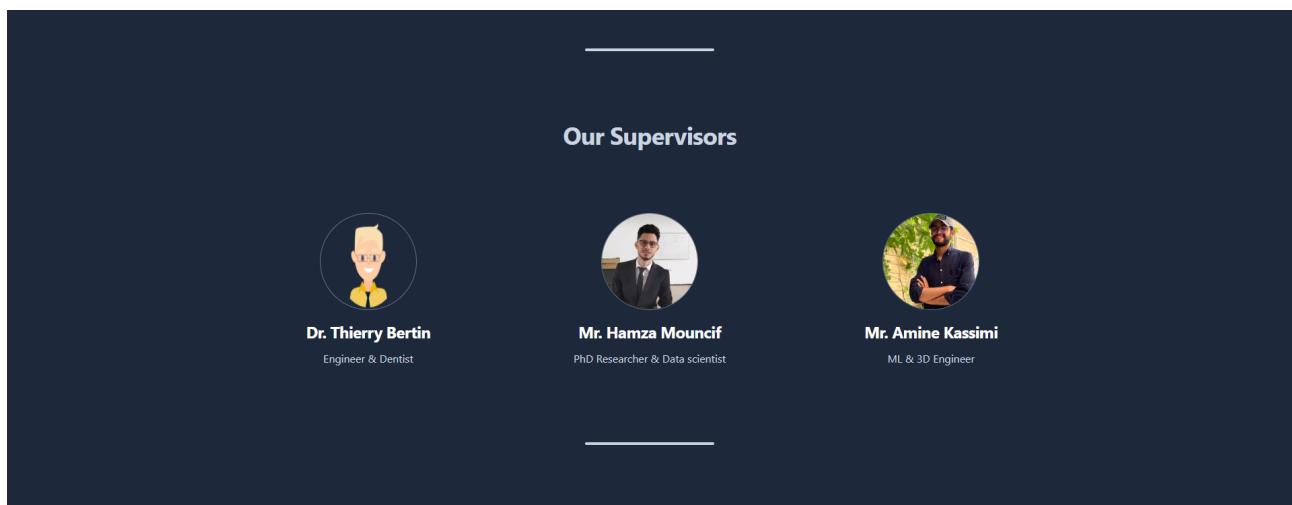


FIGURE 10.10 – Page à propos (Encadrants)



FIGURE 10.11 – Page à propos (Technologies utilisés)

10.4.4 Page d'authentification

Pour pouvoir accéder à la page "Start" qui permet de segmenter et visualiser les objets vtp, il faut obligatoirement s'inscrire et se connecter soit via Google ou Github, soit par la méthode classique avec un email et un mot de passe, cela est géré par Firebase.

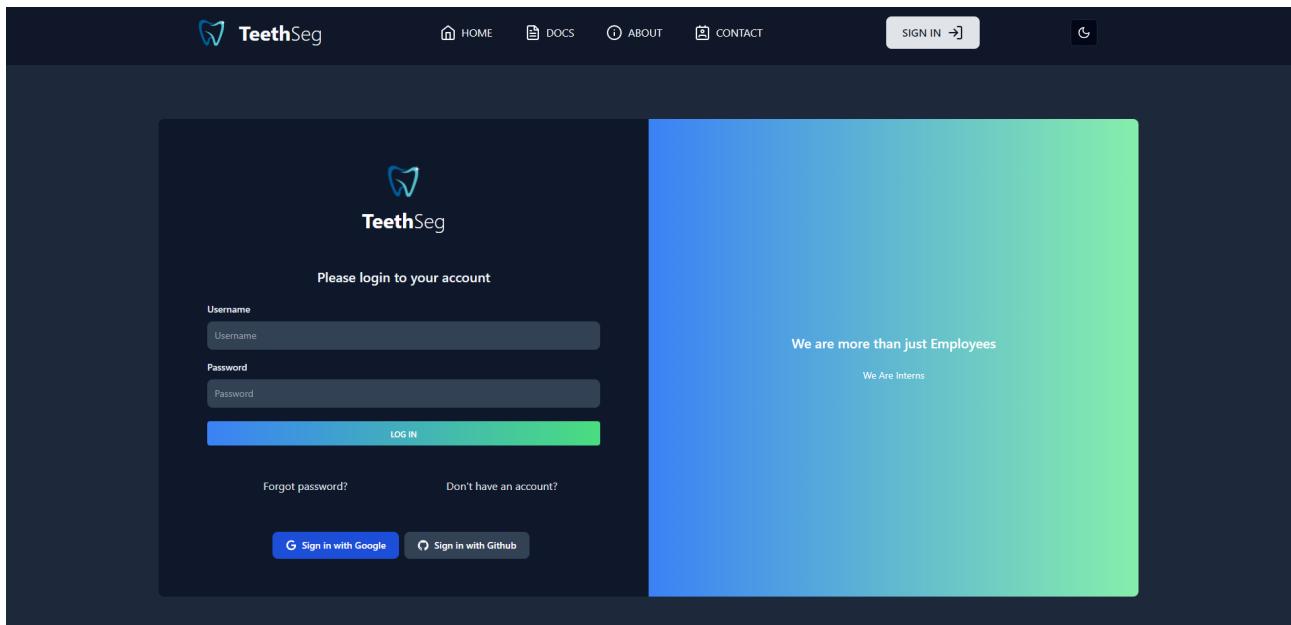


FIGURE 10.12 – Page d'authentification

10.4.5 Page Segmentation

Après authentification, on a accès à la page Start, où on peut choisir soit de segmenter un fichier obj soit de visualiser un fichier vtp. On a aussi la possibilité de discuter avec un chatbot créé avec GPT d'OpenAI et pré entraîné avec les informations sur notre application.

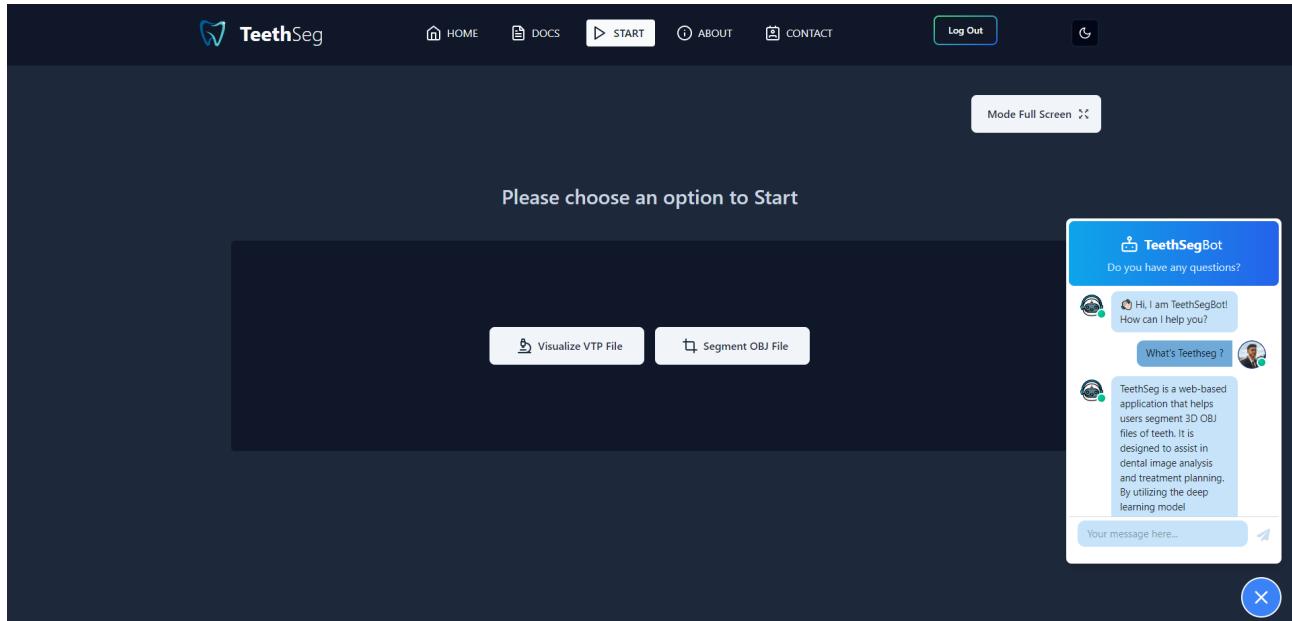


FIGURE 10.13 – Page Start avec un AI chatbot assistant

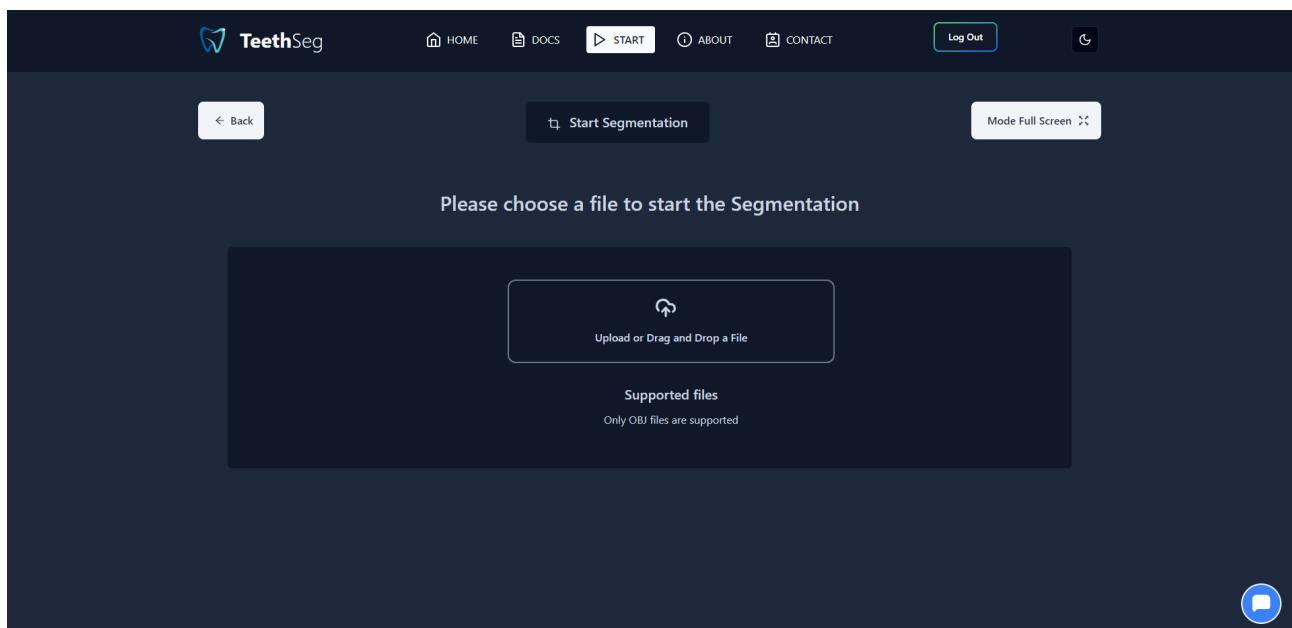


FIGURE 10.14 – Page de segmentation (Etape 1)

Après avoir importé un fichier obj, on peut visualiser l'objet et on peut cliquer sur "Start Segmentation" pour commencer la segmentation. Le fichier obj sera considerer comme l'entrée dans la requête POST à l'url de l'API : "/api/v1/predict/post_processing", après un écran de chargement qui dure 15 à 20 secondes le temps que la segmentation finisse, on a notre fichier segmenter prêt à être visualiser et telecharger.

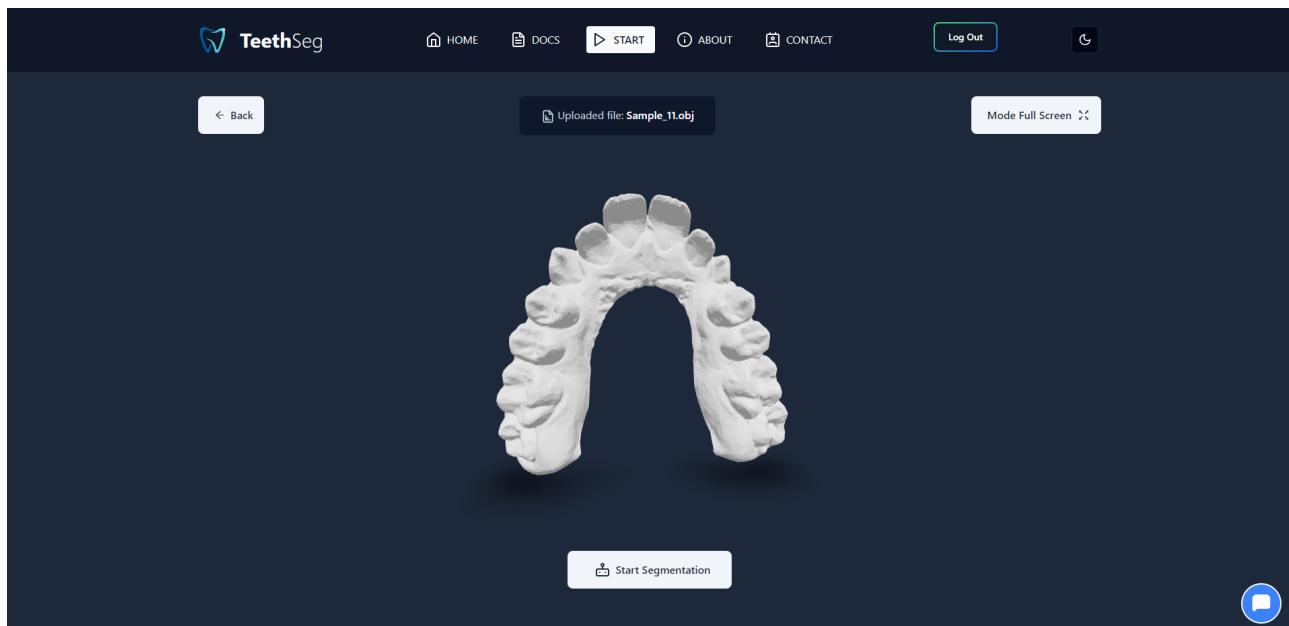


FIGURE 10.15 – Visualisation du fichier OBJ importer (Etape 2)

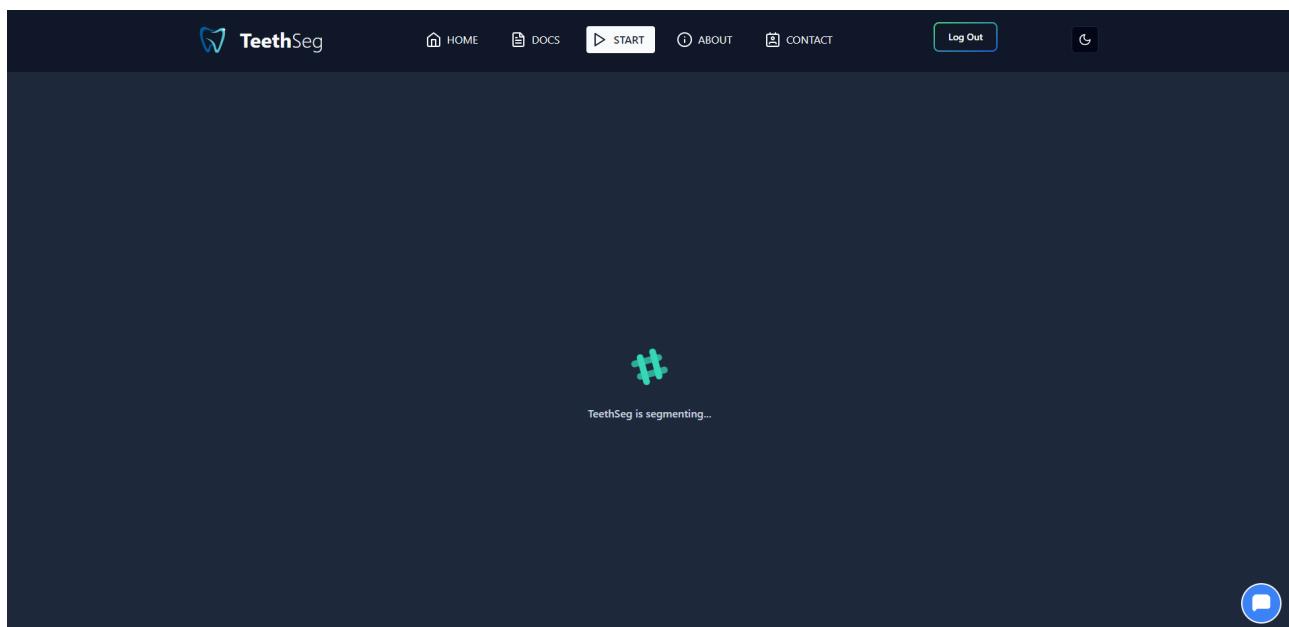


FIGURE 10.16 – Ecran de chargement (Etape 3)

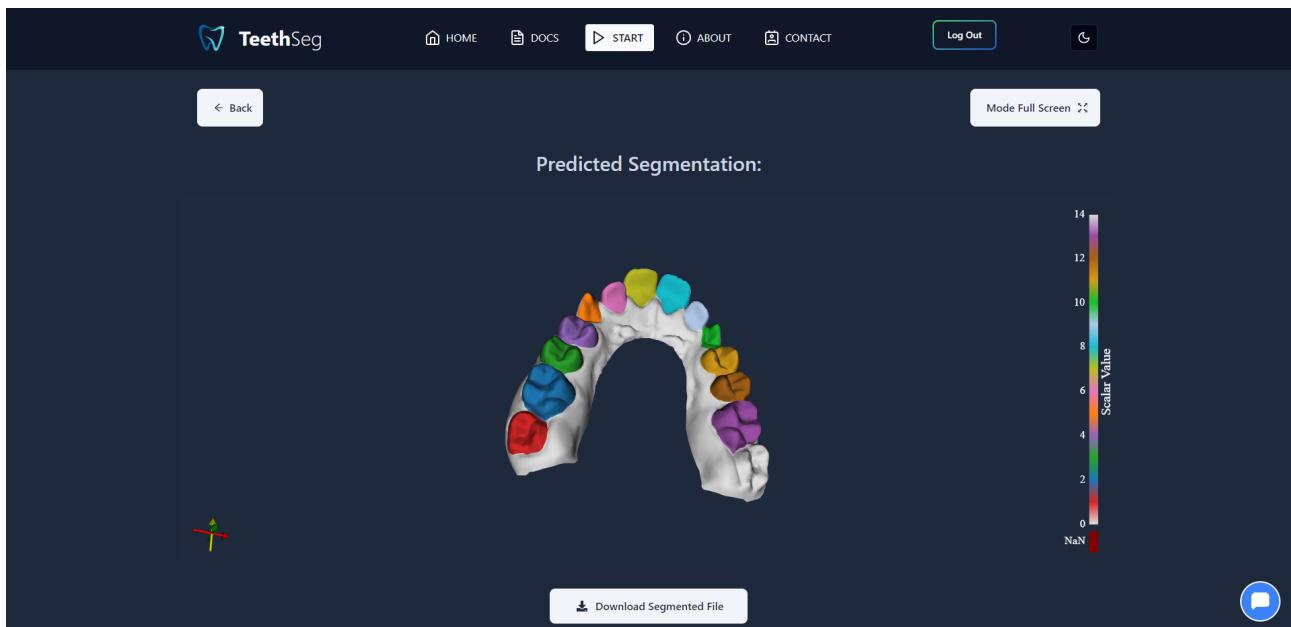


FIGURE 10.17 – Visualisation du fichier VTP après segmentation (Etape 4)

Maintenant qu'on a téléchargé le fichier vtp segmenter on peut cliquer sur "Back" pour revenir à la page Start puis on choisit "Visualize VTP file", on importe notre fichier vtp et on peut visualiser le fichier. Les deux visualisations sont mises en œuvre grâce à la librairie Vtk.js qui permet une visualisation dynamique et personnalisée.

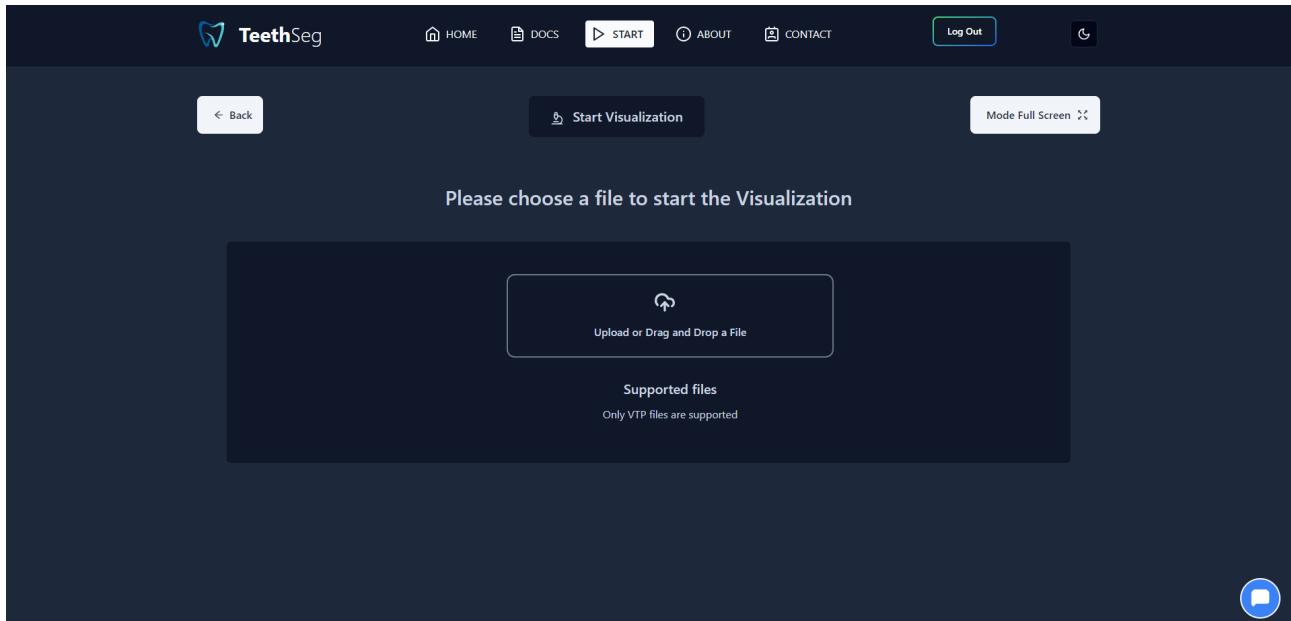


FIGURE 10.18 – Importation du fichier VTP segmenter (Etape 1)

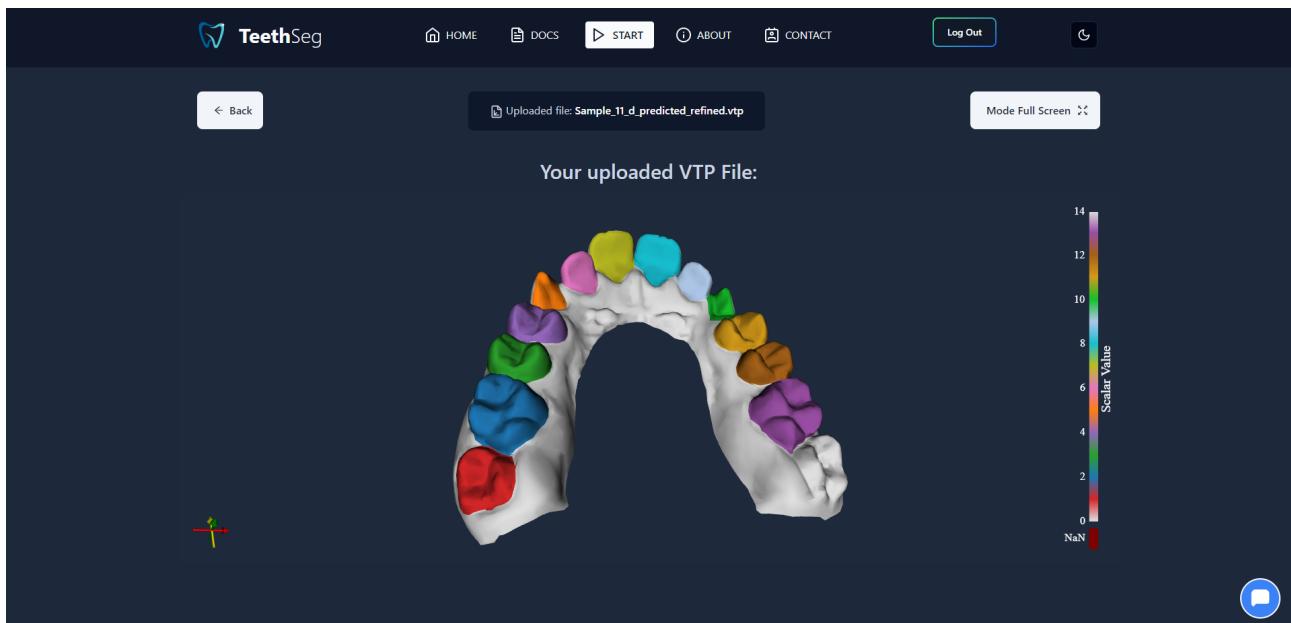


FIGURE 10.19 – Visualisation du fichier VTP segmenter (Etape 2)

Finalement, on a déployé notre application web sur Vercel sous le nom de TeethSeg.

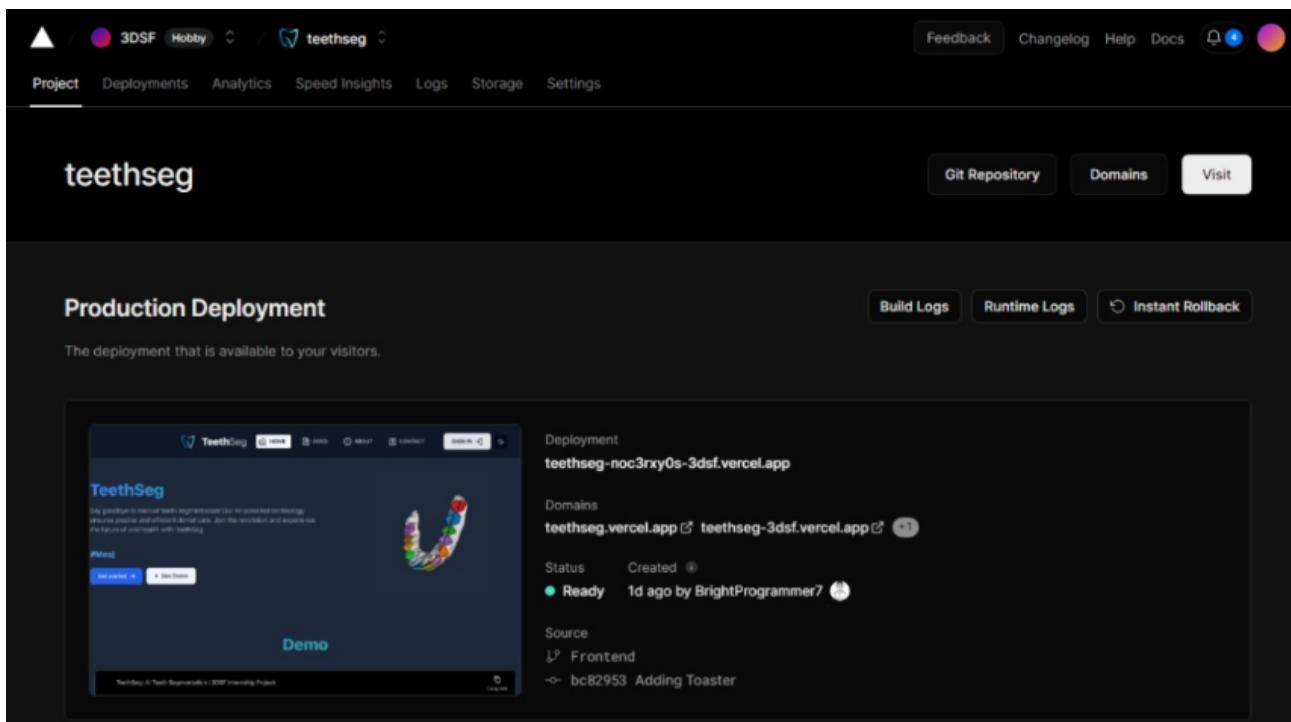


FIGURE 10.20 – Deploiement automatique à Vercel après chaque push au Github

Pour tester le site par vous même vous pouvez visiter le lien suivant : teethseg.vercel.app

Cinquième partie

Conclusion

Ce projet de fin d'année a marqué une étape cruciale dans la matérialisation des résultats issus de nos travaux de recherche et de développement. En combinant nos connaissances en Machine Learning et en développement web, notre initiative avait pour objectif de rendre notre modèle de Machine Learning accessible et exploitable grâce à une interface utilisateur conviviale et intuitive.

Dans un premier temps, nous avons développé notre back end grâce au modèle Mesh-SegNet, puis on a étudié les différentes possibilités de déploiement. On a choisi de déployer notre back end sur le cloud plus précisément AWS. Ensuite nous avons intégré notre back end avec le front grâce à une API. L'application web résultante permet aux utilisateurs de bénéficier des fonctionnalités du modèle de manière transparente, sans qu'ils aient à se soucier des détails techniques sous-jacents. Le déploiement de cette plateforme a exigé la mise en place d'une infrastructure solide, prenant en considération des aspects tels que la scalabilité, la sécurité et la disponibilité.

Par ailleurs, maintenir une veille constante des meilleures pratiques en développement web ainsi que des évolutions dans le domaine de l'IA demeure crucial. Cette démarche garantit une expérience utilisateur fluide et réactive à mesure que les technologies continuent de progresser. En sondant les défis spécifiques liés au déploiement d'un modèle de Machine Learning au sein d'une application web, ce projet de fin d'année apporte une contribution significative au secteur de l'intelligence artificielle et de l'ingénierie logicielle.

Notre travail a permis de diversifier notre perspective quant à l'exploitation d'un nouveau type de données pour l'apprentissage profond. Nous avons eu l'opportunité de découvrir de nouvelles techniques et concepts d'apprentissage automatique appliqués aux images 3D. Malgré un départ avec des connaissances limitées dans ce domaine au début de notre stage, l'expérience acquise constitue un atout de poids pour notre trajectoire professionnelle future. Pour conclure, ce projet ouvre la voie à de nouvelles perspectives tout en laissant entrevoir des explorations à venir.

Bibliographie

- [1] [09/07/2023], <https://3dsmartfactory.csit.ma/>, **Site Officiel de 3D SMART FACTORY**
- [2] [14/07/2023], <https://arxiv.org/pdf/2109.11941.pdf>, **Article Scientifique de MeshSegNet**, Tai-Hsien Wu, Chunfeng Lian, Sanghee Lee, Matthew Pastewait, Christian Piers, Jie Liu, Fang Wang, Li Wang, Chiung-Ying Chiu, WENCHI WANG, Christina Jackson, Wei-Lun Chao, Dinggang Shen, Ching-Chang Ko, **Computer Vision and Pattern Recognition**
- [3] [14/07/2023], <https://medium.com/stanford-cs224w/deep-learning-on-3d-meshes-9608a5b33c98>, **Deep Learning on 3D Meshes**, Anya Fries
- [4] [15/07/2023], <https://www.scrum.org/>, **Site Officiel de SCRUM**
- [5] [16/07/2023], <https://trello.com/>, **Trello**
- [6] [18/07/2023], <https://vitejs.dev/guide/>, **Vite Documentation**
- [7] [20/07/2023], <https://examples.vtk.org/site/VTKFileFormats/>, **Fichier VTK format**
- [8] [22/07/2023], https://en.wikipedia.org/wiki/Polygon_mesh, **Polygon Mesh**
- [9] [17/08/2023], <https://fastapi.tiangolo.com/tutorial/first-steps/>, **Création d'un REST API avec FastAPI**
- [10] [19/08/2023], <https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning>, **MLOps CI/CD**
- [11] [21/08/2023], <https://www.redhat.com/fr/topics/api/what-is-a-rest-api>, **REST API**
- [12] [20/08/2023], <https://staruml.io/>, **StarUML**
- [13] [21/08/2023], <https://colab.research.google.com/>, **Google COLAB**
- [14] [21/08/2023], <https://neptune.ai/blog/how-to-use-google-colab-for-deep-learning-complete-tutorial>, **Entraînement d'un modèle de deep learning sur Google Colab**
- [15] [22/08/2023], https://kitware.github.io/vtk-js/docs/vtk_react.html, **Documentation Vtk.js React**
- [16] [22/08/2023], <https://docs.pmnd.rs/react-three-fiber/getting-started/introduction>, **React Three Fiber Documentation**
- [17] [22/08/2023], <https://blog.bitsrc.io/tutorial-build-a-chatbot-with-react-and-openai-2c183c50991e>, **Chatbot avec React et OpenAI**

- [18] [22/08/2023], <https://threejs.org/docs/#examples/en/loaders/OBJLoader>, **OBJ Loader ThreeJs Documentation**
- [19] [22/08/2023], <https://www.lucidchart.com/pages/fr/exemple/diagramme-de-gantt-en-ligne>, **Diagramme De GANTT en ligne**
- [20] [22/08/2023], <https://www.simplilearn.com/tutorials/docker-tutorial>, **Guide d'utilisation de Docker**
- [21] [23/08/2023], <https://blog.searce.com/fastapi-container-app-deployment-using-aws-lambda-and-api-gateway-6721904531d0>, **Deploiement d'un FastAPI api sur AWS**
- [22] [25/08/2023], <https://vercel.com/docs/frameworks/vite>, **Deploiement d'un projet Vite sur Vercel**